

## ABSTRACT

Title of dissertation: MACHINE LEARNING OF FACIAL  
ATTRIBUTES USING EXPLAINABLE,  
SECURE, AND GENERATIVE  
ADVERSARIAL NETWORKS

Pouya Samangouei  
Doctor of Philosophy, 2018

Dissertation directed by: Professor Rama Chellappa  
Department of Electrical and  
Computer Engineering

"Attributes" are referred to abstractions that humans use to group entities and phenomena that have a common characteristic. In machine learning (ML), attributes are fundamental because they bridge the semantic gap between humans and ML systems. Thus, researchers have been using this concept to transform complicated ML systems into interactive ones. However, training the attribute detectors which are central to attribute-based ML systems can still be challenging. It might be infeasible to gather attribute labels for rare combinations to cover all the corner cases, which can result in weak detectors. Also, it is not clear how to fill in the semantic gap with attribute detectors themselves. Finally, it is not obvious how to interpret the detectors' outputs in the presence of adversarial noise.

First, we investigate the effectiveness of attributes for bridging the semantic gap in complicated ML systems. We turn a system that does continuous authentication of human faces on mobile phones into an interactive attribute-based one. We

employ deep multi-task learning in conjunction with multi-view classification using facial parts to tackle this problem. We show how the proposed system decomposition enables efficient deployment of deep networks for authentication on mobile phones with limited resources.

Next, we seek to improve the attribute detectors by using conditional image synthesis. We take a generative modeling approach for manipulating the semantics of a given image to provide novel examples. Previous works condition the generation process on binary attribute existence values. We take this type of approaches one step further by modeling each attribute as a distributed representation in a vector space. These representations allow us to not only toggle the presence of attributes but to transfer an attribute style from one image to the other. Furthermore, we show diverse image generation from the same set of conditions, which was not possible using existing methods with a single dimension per attribute.

We then investigate filling in the semantic gap between humans and attribute classifiers by proposing a new way to explain the pre-trained attribute detectors. We use adversarial training in conjunction with an encoder-decoder model to learn the behavior of binary attribute classifiers. We show that after our proposed model is trained, one can see which areas of the image contribute to the presence/absence of the target attribute, and also how to change image pixels in those areas so that the attribute classifier decision changes in a consistent way with human perception.

Finally, we focus on protecting the attribute models from un-interpretable behaviors provoked by adversarial perturbations. These behaviors create an inexplicable semantic gap since they are visually unnoticeable. We propose a method

based on generative adversarial networks to alleviate this issue. We learn the training data distribution that is used to train the core classifier and use it to detect and denoise test samples. We show that the method is effective for defending facial attribute detectors.

Machine Learning of Facial Attributes Using Explainable, Secure,  
and Generative Adversarial Networks

by

Pouya Samangouei

Dissertation submitted to the Faculty of the Graduate School of the  
University of Maryland, College Park in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
2018

Advisory Committee:  
Professor Rama Chellappa, Chair/Advisor  
Professor Larry Davis  
Professor Behtash Babadi  
Professor David Jacobs  
Dr. Carlos D. Castillo



© Copyright by  
Pouya Samangouei  
2018



## Acknowledgments

First and foremost I'd like to thank my advisor, Professor Rama Chellappa, for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past four years. Although he had been extremely busy, he always made himself available for help and advice. Besides this dissertation, he has greatly influenced other aspects of my life and I see him as both an academic and a life mentor. Words cannot describe how much I feel lucky and grateful to be advised by such an extraordinary individual.

I owe my sincerest thanks to my family - my mother, father, brother, and my dear aunt Minoo Mohagheghzadeh who was my first math teacher. They have always stood by and guided me throughout my career. Words cannot express the gratitude I owe them. I hope this dissertation is worth the difficulties that they had to endure because of the forced inhuman travel ban that was in place in the second half of my Ph.D. period.

My colleagues at the University of Maryland have enriched my graduate life in many ways and deserve a special mention. Vishal Patel held my hand and directed me throughout the first chapter of this dissertation that has led to many publications, a journal paper, a patent, and a book chapter. Emily Hand with whom I wrote a book chapter and collaborated on a patent and was the best office mate that anybody could want. Maya Kabkab and Mahyar Najibi were terrific coauthors of four papers, which already are attracting a lot of attention. I also want to thank Carlos Castillo, Azadeh Alavi, and Jun-Cheng Chen from whom I learned a lot and

had a significant role in making my life easier throughout my Ph.D. I want to also thank Micheal Rotkowitz, Ali Koochakzadeh, and Sina Miran for collaborating on an interesting paper that was not a part of this dissertation but had been a source of inspiration.

I want to thank my internship mentors Hamid Sheikh and Raja Bala at Samsung, Wael Abdal-Majeed at Information Sciences Institute, and Nathan Silberman at Butterfly Network who had exposed me to a diverse set of problems and great teams. I learned a lot during the periods that I was interning.

I would not have been able to enjoy my time here while working on this dissertation if it were not for my friends. I want to thank Amir Montazeri, Ali Akbari, Mohammad Ganji, Aida Mostafazadeh, Sina Zamani, Rahmtin Rotabi, Farhad Saffaraval, Alborz Alavian, Ali Moeini, Mahshid Najafi, Ladan Rabiee, Niloufar Shadab, Alireza Sheikhattar, Melika Abolhassani, Bahar Zarrin, Arian Asefzadeh, Kia Karbasi, Nima Moeini, Hasan Lotfi, Soudeh Montazeri, and many more with whom I made a lot of fantastic memories.

I would also like to acknowledge help and support from some of the University of Maryland Center for Advanced Computer Studies. They maintained the computational resources that I was using throughout my Ph.D. studies and were always there to address any issues that I had, even on weekends.

My housemates at my place of residence have been a crucial factor in my finishing smoothly. I'd like to express my gratitude to Daniel Voica, Amy Steele, Jason Fang, and Bjorn Adamatti for being wonderful housemates.

Chapter five of this research is based upon work supported by the Office of the

Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

Chapter one of this research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Defense Advanced Research Projects Agency (DARPA), via DARPA R&D Contract No. FA8750-13-2-0279. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

It is impossible to remember all, and I apologize to those I've inadvertently left out.

# Table of Contents

Acknowledgements	ii
List of Tables	vii
List of Figures	x
List of Abbreviations	xiv
1 Introduction	1
1.1 Attributes for continuous authentication . . . . .	4
1.2 Conditional image syndissertation from attribute representations . . .	6
1.3 Explaining attribute detectors . . . . .	8
1.4 Protecting against induced semantic gap . . . . .	10
1.5 Organization . . . . .	11
2 Attribute-based continuous active authentication	12
2.1 Introduction . . . . .	12
2.2 Related work . . . . .	14
2.3 Attribute-based Active Authentication on Mobile Devices . . . . .	19
2.4 Efficient Deep Features for Attribute Detection on Mobile Phones . .	41
3 Conditional Image Generation From Attribute Vectors	61
3.1 Introduction . . . . .	61
3.2 Related Work . . . . .	63
3.3 Model . . . . .	65
3.4 Training . . . . .	67
3.5 Experiments . . . . .	73
4 Model Explanation via Decision Boundary Crossing Transformations	76
4.1 Introduction . . . . .	76
4.2 Related work . . . . .	79
4.3 Model . . . . .	82
4.4 Priors for Interpretable Image Transformations . . . . .	86
4.5 Experiments . . . . .	88

5	Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models	99
5.1	Introduction	99
5.2	Related work and background information	101
5.3	Proposed Defense-GAN	108
5.4	Experiments	111
5.5	Optimality of $p_g = p_{\text{data}}$ for WGANs	120
5.6	Difficulty of GD-based white-box attacks on Defense-GAN	121
5.7	Neural network architectures	122
5.8	Qualitative examples	124
5.9	Additional results on the effect of varying the number of GD iterations $L$ and random restarts $R$	125
5.10	Additional results on white-box attacks	126
5.11	Time complexity	127
5.12	An unsuccessful attempt to attack Defense-GAN	128
6	Summary and Future Directions	132
6.1	Summaries	132
6.2	Future Directions	134
	Bibliography	138

## List of Tables

2.1	Accuracies of the 44 attribute classifiers proposed in this work on the PubFig dataset [Kumar et al., 2009]. . . . .	26
2.2	Accuracies of the attribute classifiers proposed in this work on available attributes on the FaceTracer dataset [Kumar et al., 2008]. . . . .	27
2.3	Accuracy of the attribute classifiers for CNNA [Fathy et al., 2015] dataset. . . . .	27
2.4	The EER values for different methods on the MOBIO dataset. . . . .	29
2.5	The EER values corresponding to the cross-device experiment on the MOBIO dataset. . . . .	31
2.6	The EER values of different methods for the AA01 dataset. . . . .	32
2.7	The EER values corresponding to the cross-session experiments for the AA01 dataset. 1 is the office light session, 2 is the low light session, 3 is the natural light session. . . . .	34
2.8	Average memory usage per attribute classifier for full face . . . . .	36
2.9	Comparison of EER values for LBP, attribute detectors of Section 2.3.1, linear models of Section 2.3.8. The scale L-SVM 1 is trained on images of size $128 \times 168$ and the rest are scaled by the indicated value. The best EER is gained from L-SVMs of Section 2.3.8 with scale 0.5. . . . .	39
2.10	The speed and power consumption of different realization of the classifiers learned with the simplified training framework on Google Nexus 5 device. W/O column means our algorithm extract all the attributes given aligned and cropped face. In last row we assumed that we are doing authentication with the speed 1fps. W/ column first detects the face then extracts attributes. We can authenticate 17.6 hours every second employing our classifiers with best EER of Table 2.9 on a Nexus 5. . . . .	41
2.11	The performance comparison of attribute detection methods. . . . .	42
2.12	The architectures of our networks. The number of parameters depends on the face region that they operate on and can be found in Table 2.16. . . . .	43



2.13	The face regions that are extracted by cropping around the landmark points and their corresponding number of attributes. A Multi*-CNNAA that operates on a face crop has “No. of Attributes” tasks.	44
2.14	The EER values for the different experiments on AA01 [Fathy et al., 2015] dataset. The sessions numbers are: 1. Office light 2. Low light 3. Natural light. DiscAttrs column contains the EER values using the discovered attributes.	53
2.15	The EER values corresponding to MOBIO dataset experiments.	57
2.16	Network size and prediction speed of the networks. The D-* means it has MultiDeep-CNNAA architecture and W-* means it is MultiWide-CNNAA . The Binary*-CNNAA network prediction times are just for one attribute. For all of them together it will be 40 times this value.	58
4.1	AOSPC value (higher is better, see (4.17) after 10 steps for different segmentation thresholds. Although, ExplainGAN is not directly optimized for this metric, its performance is comparable to reasonable baselines for explanation in classifiers. A larger AOSPC means that the sensitivity of the segments that are perturbed in 10 steps is higher.	94
4.2	Quantitative substitutability experiments across datasets. Class 0 and Class 1 are the classes that the given classifier is trained to identify. Transformed/Composite 0/1 column shows the accuracy of the classifiers when just transformations/compositions of the images used at training time. Ceiling represents the accuracy of the base classifier on the same test set.	96
4.3	Substitutability on Ultrasound Dataset. Transformed/Composite 0/1 shows the accuracy of a classifier on test set when the original samples are replaced with Transformed/Composite 0/1 at training phase. Both Transformed/Composite shows the accuracy of the classifier when all of the images are replaced with Transformed/Composite. Note that PixelDA is a oneway transformer.	98
5.1	Classification accuracies of different classifier and substitute model combinations using various defense strategies on the MNIST dataset, under FGSM black-box attacks with $\epsilon = 0.3$ . Defense-GAN has $L = 200$ and $R = 10$ .	115
5.2	Classification accuracies of different classifier and substitute model combinations using various defense strategies on the F-MNIST dataset, under FGSM black-box attacks with $\epsilon = 0.3$ . Defense-GAN has $L = 200$ and $R = 10$ .	115
5.3	Classification accuracy of Model F using Defense-GAN ( $L = 400$ , $R = 10$ ), under FGSM black-box attacks for various noise norms $\epsilon$ and substitute Model E.	117
5.5	Neural network architectures used for classifiers and substitute models.	123
5.8	Classification accuracy of Model F using Defense-GAN with various number of iterations $L$ ( $R = 10$ ), on the MNIST dataset, under FGSM black-box attack with $\epsilon = 0.3$ .	125

5.11	Classification accuracy of Model F using Defense-GAN with various number of random restarts $R$ ( $L = 100$ ), on the F-MNIST dataset, under FGSM black-box attack with $\epsilon = 0.3$ . . . . .	126
5.13	Average time, in seconds, to compute reconstructions of MNIST/F-MNIST images for various values of $L$ and $R$ . . . . .	127
5.4	Classification accuracies of different classifier models using various defense strategies on the MNIST (top) and F-MNIST (bottom) datasets, under FGSM, RAND+FGSM, and CW white-box attacks. Defense-GAN has $L = 200$ and $R = 10$ . . . . .	129
5.6	Neural network architectures used for GANs. . . . .	130
5.7	Neural network architecture used for the MagNet encoder. . . . .	130
5.9	Classification accuracy of Model F using Defense-GAN with various number of iterations $L$ ( $R = 10$ ), on the F-MNIST dataset, under FGSM black-box attack with $\epsilon = 0.3$ . . . . .	130
5.10	Classification accuracy of Model F using Defense-GAN with various number of random restarts $R$ ( $L = 100$ ), on the MNIST dataset, under FGSM black-box attack with $\epsilon = 0.3$ . . . . .	131
5.12	Classification accuracies of different classifier models using various defense strategies on the CelebA gender classification task, under FGSM, RAND+FGSM, and CW white-box attacks. Defense-GAN has $L = 200$ and $R = 2$ . . . . .	131

## List of Figures

2.1	Overview of our attribute-based authentication method. . . . .	19
2.2	Training phase pipeline for each attribute classifier. Landmarks are first detected on a given face. Different facial components are then extracted from these landmarks. Then for each part, features are extracted with different cell sizes and the dimensionality of features is reduced using principle component analysis. Classifiers are then learned on these low-dimensional features. Finally, top five <i>C</i> 's are selected as our attribute classifier. . . . .	20
2.3	Illustration of our attribute classifiers on sample face images from the AA01 (first two images) and the MOBIO (last image) datasets. . . .	25
2.4	Sample images from the MOBIO dataset. One can clearly see the different illumination conditions in this dataset. . . . .	28
2.5	Performance evaluation on the MOBIO dataset. . . . .	29
2.6	Cross device robustness. Laptop session videos are used for enrollment and the data from the remaining sessions are used for testing. . . . .	30
2.7	Sample images from the AA01 dataset. (a), (b) and (c) show some sample images from session 1, 2 and 3, respectively. . . . .	31
2.8	Performance evaluation for the AA01 dataset. . . . .	32
2.9	Session-specific performance evaluations for the AA01 dataset. (a) Gallery and probe data from session 1. (b) Gallery and probe data from session 2. (c) Gallery and probe data from session 3. (d) Gallery data from session 1 and probe data from sessions 2 and 3. (e) Gallery data from session 2 and probe data from sessions 1 and 3. (f) Gallery data from session 3 and probe data from sessions 2 and 1. . . . .	33
2.10	Comparison of linear SVMs with model learned in section 2.3.1 and LBP. The best result among all is achieved with linear models of scale 0.5 i.e. face crop size of $64 \times 80$ . . . . .	39
2.11	Sample images from subspace clustering of face part embedding in attribute space. . . . .	47
2.12	Sample images of the three sessions of the AA01 dataset. . . . .	52

2.13	ROC curve of different experiments on AA01 [Fathy et al., 2015] and MOBIO [McCool et al., 2012] dataset. (a) is the ROC curve of AA01 with all of the sessions together in gallery and probe. (b) is the ROC curve of MOBIO with all of the mobile sessions together with the last session videos as gallery and the rest of the session as probe. (c) is the ROC curve of the cross-device experiment. . . . .	53
2.14	Sample images of the three sessions of the MOBIO dataset. First row images are from different sites, second row is the pairs with the same identities in two different sessions. . . . .	55
3.1	Visualization of approaches to attribute manipulation as graphical models. (a) The graphical model for Fader Networks. (b) The graphical model for CRISPR. . . . .	65
3.2	Architectural overview of the CRISPR architecture, best viewed in color. The image $x$ is encoded by $E$ producing invariance vector $z$ and attribute vectors $a_0$ and $a_1$ . These are concatenated and passed to decoder $G$ which produces reconstruction $\hat{x}$ . Each continuous attribute vector is decoded using a linear classifier $H_i$ . Auxiliary components of the architecture are framed by the dashed green outline. Discriminator $D$ encourages the reconstructions to both appear natural and exhibit the expected attributes, classifier $C_{\text{inv}}$ encourages $z$ to encode only non-attribute information and classifier $C_{\text{attr}}$ encourages each attribute vector to encode only its attribute information. . . . .	66
3.3	Examples illustrating the Diverse Swap operation using the 'bangs' attribute. The left-most column demonstrates the inputs and the remaining columns illustrates the results of sampling from the space of bangs attributes. . . . .	74
3.4	Examples illustrating the Diverse Swap operation using the 'eye-glasses' attribute. The left-most column demonstrates the inputs and the remaining columns illustrates the results of sampling from the space of bangs attributes. The results demonstrate not only the diversity of the selection of eye-glasses, but also a failure case (second row, last image), the sampled attributes may not all be atomic. . . .	75
3.5	Examples illustrating the Borrow operation using the 'smile' attribute'. As these examples illustrate, CRISPR is able to effectively <i>borrow</i> the smile from the reference image and apply it to the query image. . . .	75
4.1	Model architecture of ExplainGAN. Inference (in blue frame) consists of passing an image $x$ of class $j$ into the appropriate encoder $E_j$ to produce a hidden vector $z_j$ . The hidden vector is decoded to simultaneously create its reconstruction $G_j(z_j)$ , a transformed image of the opposite class $G_{1-j}(z_j)$ and a mask showing where the changes were made $G_m(z_j)$ . Composite images $C_0$ and $C_1$ merge the reconstruction and transformation with the original image $x$ . . . . .	83

4.2	An example of Ultrasound images from our Medical Ultrasound dataset. (a) A canonical Apical 2 Chamber view. (b) A canonical Apical 4 Chamber view. (c) A difficult Apical 2 Chamber view that is easily confused for a 4 Chamber view. (d) A difficult Apical 4 Chamber view that is easily confused for a 2 Chamber view. . . . .	89
4.3	Qualitative visualization of the ExplainGAN model on two datasets: CelebA and our Medical Ultrasound dataset. The “input” column represents images $x \in S_0$ , the “transformed” column represents ExplainGAN’s transformation, $G_1(z_0)$ , to the opposite class. The “mask” column illustrates the model’s changes, $G_m(z_0)$ , and the “composite” column shows the composite images, $C_1(z_0)$ . The results indicate that in the case of object-related transformations, such as glasses or mustaches, ExplainGAN effectively performs a weakly supervised segmentation of the object. In the ultrasound case, ExplainGAN illustrates which anatomical areas the model is cuing on: the right ventricle and pericardium. . . . .	92
4.4	Comparison of different methods for explaining the model’s decision. <b>Fashion-MNIST</b> : transforming from pullover to shirt, <b>Ultrasound</b> : transforming from A2C to A4C (see 4.2 for examples of A2C and A4C views), <b>CelebA</b> : transforming from faces without eyeglasses to faces with eyeglasses, <b>MNIST</b> : transforming from 4 to 9. . . . .	95
4.5	Boundary-crossing images have varying explanatory power: images carry more explanatory power if (1) they can be used as substitutes in the original dataset without affecting the classifier and (2) they are different from a query image in small and easily localized ways. (a) displays an image classified as a jacket and not a pullover. (b) shows an image of a pullover which is substitutable and whose localized mask illustrates the models belief that removing a zipper and the jacket ribs would make the original image into a pullover. (c) shows another pullover but non-localized mask doesn’t explain why this is a pullover and not a jacket. (d) shows an adversarial image which is completely unsubstitutable and provides no localized explanation. . . . .	96
5.1	Overview of the Defense-GAN algorithm. . . . .	110
5.2	$L$ steps of Gradient Descent are used to estimate the projection of the image onto the range of the generator. . . . .	110
5.3	Classification accuracy of Model F using Defense-GAN on the MNIST dataset, under FGSM black-box attacks with $\epsilon = 0.3$ and substitute Model E. Left: various number of iterations $L$ are used ( $R = 10$ ). Right: various number of random restarts $R$ are used ( $L = 100$ ). . . .	116
5.4	ROC Curves when using Defense-GAN MSE for FGSM attack detections on the MNIST dataset (Classifier Model F, Substitute Model E). Left: Results for various number of GD iterations are shown with $R = 10$ , $\epsilon = 0.30$ . Middle: Results for various number of random restarts $R$ are shown with $L = 100$ , $\epsilon = 0.30$ . Right: Results for various $\epsilon$ are shown with $L = 400$ , $R = 10$ . . . . .	117

5.5	ROC Curves when using Defense-GAN MSE for FGSM attack detections on the F-MNIST dataset (Classifier Model F, Substitute Model E). Left: Results for various number of GD iterations are shown with $R = 10$ , $\epsilon = 0.30$ . Middle: Results for various number of random restarts $R$ are shown with $L = 100$ , $\epsilon = 0.30$ . Right: Results for various $\epsilon$ are shown with $L = 200$ , $R = 10$ . . . . .	118
5.6	Examples from MNIST and F-MNIST. Left: Original, FGSM adversarial $\epsilon = 0.3$ , and reconstruction images for $R = 1$ and various $L$ are shown. Right: Original, FGSM adversarial $\epsilon = 0.3$ , and reconstruction images for $L = 25$ and various $R$ are shown. . . . .	124
5.7	Examples from MNIST and F-MNIST: Original, FGSM adversarial and reconstruction images for $L = 50$ , $R = 15$ and various $\epsilon$ are shown.	124
5.8	Classification accuracy of different models using Defense-GAN on the MNIST dataset, under FGSM white-box attack with $\epsilon = 0.3$ , for various number of iterations $L$ and $R = 10$ . . . . .	126

## List of Abbreviations

ACAA	Attribute-based Continuous Active Authentication
CNN	Deep Convolutional Neural Networks
CPU	Central Processing Unit
CW	Carlini and Wagner
CNNAA	Convolutional Neural Networks for Active Authentication
DCGAN	Deep Convolutional GAN
DCNN	Deep Convolutional Neural Networks
DTD	Deep Taylor Decomposition
EER	Equal Error Rate
FGSM	Fast Gradient Sign Method
FAR	False Acceptance Rate
GAN	Generative Adversarial Networks
GB	Gigabyte
GPU	Graphical Processing Unit
GradCAM	Gradient-weighted Class Activation Mapping
HOG	Histogram of Oriented Gradients
KNN	K-nearest neighbors
LBP	Local Binary Patterns
LRP	Layer-wise Relevance Propagation
MB	Megabyte
ML	Machine Learning
MSE	Mean squared error
PCA	Principle Component Analysis
RAND	Random
RAM	Random Access Memory
RBF	Radial Basis Function
ROC	Receiver operating Characteristic
SSC	Sparse Subspace Clustering
SVM	Support Vector Machine
TAR	True Acceptance Rate
TV	Total Variation
WGAN	Wasserstein GAN

## Chapter 1: Introduction

The notion of the “semantic gap” between humans and machine learning (ML) algorithms/systems has become increasingly important. When the gap exists, there is a lack of interpretation for how an ML algorithm or system behaves. This disconnect can arise in different levels of the ML system, which often results in unpredictable behaviors, invalid confidence measures, and unnecessarily complicated machine learning models. Besides the semantic gaps that naturally happens, there can be induced semantic gaps which can make the ML systems insecure and show unexplainable behaviors.

In this dissertation, we focus on the concept of “attributes” for bridging the semantic gap. Given a set of entities, an attribute is a common characteristic of the items which divides the set into different non-overlapping partitions (usually two, one partition that has the attribute and the other does not). For example, if the entities are face images “gender” is an attribute that divides the set into images of “male” and “female” faces. More formally, an attribute detector  $f$  is defined as  $f : X \rightarrow [0, 1]$  where  $X \in \mathbb{R}^{W \times H \times C}$  is the input image and 1 corresponds to the presence of the attribute and 0 to the absence.

Using multiple attributes to identify entities has been a popular approach for



decomposing complicated ML tasks into simpler ones. This approach can either mimic the complex behavior or narrow down the search space significantly leading to a more robust and efficient system. For example in a face recognition system [Kumar et al., 2009] or for object recognition [Farhadi et al., 2009], a collectively large set of attributes has been shown to be enough to get an accurate identity preserving representation for faces and objects. This idea can be specially interesting for converting demanding algorithms into amenable ones for devices with limited resources. In this dissertation, we look at the problem of continuous authentication on mobile phones.

Training robust attribute detectors is therefore central to attribute-based ML systems. In this dissertation, we employ deep convolutional neural networks (CNNs) for the attribute classifiers since they have proven to work well for this task [Levi and Hassner, 2015], as well as many other computer vision tasks. However, deploying CNNs on mobile phones is challenging because of the limited resources [Sarkar et al., 2016]. We address this challenge by introducing deep CNNs for attribute-based continuous authentication which are run efficiently on mobile phones.

To be more efficient and also make use of inter-attribute dependencies, usually a single CNN is trained for predicting all of the attributes together. At training phase of such models, for each training data  $x \in \mathcal{X}_{train}$  we will have a target vector  $\mathbf{y} = [y_0 \dots y_{n-1}]$  where each dimension determines the presences of one of the  $n$  attributes. As  $n$  increases the label space expands exponentially, and therefore more training data is needed to cover the whole label space. Thus, there are usually attributes that are not present in most of the images. For example, there might not

be enough face images with the attribute "mustache" present. We take a generative modeling approach to address this issue by learning to capture the attributes in vectorized representations and transferring them to images that do not have those attributes.

After converting an ML system into an attribute-based one and training good attribute detectors, there still exists one semantic gap: what does a deep black box attribute detector do to make a prediction. Understanding the behavior of deep models can be critically important for certain applications, such as ML-based evidence in legal courts, education, or even model debugging. To understand the behavior, we should make sure to have an algorithm that is guaranteed to have interpretable outcomes. Earlier works have defined the explanations as a sensitivity map of the model output with respect to the given input image [Zhou et al., 2016, Selvaraju et al., 2016a]. However, there are no guarantees that these masks are interpretable by humans, and also it is not known how the pixel values should change in those areas so that the outcome of the binary attribute classifier is toggled. In this dissertation, we propose a new way of exploring this issue which results in interpretable masks and also provides pixel-wise values that make the classifier to cross the decision boundary.

Besides explaining the behavior of deep attribute models, it is essential to prevent the detectors from induced uninterpretable outcomes. Adversarial agents enforce these behaviors by adding small perturbations to the input pixels. These perturbations are usually unnoticeable; thus, from the user perspective, it seems that the network gives two different outputs for the same image. In the literature,

these adversarial perturbations are defended by strong prior assumptions on the type of attacks [Goodfellow et al., 2014b]. However, in a realistic scenario, we have no prior information about the type of attacks that can happen on a given ML system. We propose a method based on generative adversarial networks to solve the problem with no assumptions on the attacks.

### 1.1 Attributes for continuous authentication

Continuous authentication, is as opposed to one-time initial authentication of the users using passwords or biometrics and protects the mobile device even if the initial authentication is bypassed. This system can be designed without using any machine learning algorithms, for instance by asking the password every minute. However, it is more desirable if the continuous authentication method is non-intrusive, meaning that it does not interrupt the usage of the device. ML-based solutions have been pursued to alleviate this issue [McCool et al., 2012]. Such systems produce a similarity score between the current and enrolled user representations. These representations are inferred from sensory inputs such as camera images. The calculated representations are usually of the form of a vector that is not interpretable on their own.

We are the first to propose using attributes to for mobile continuous authentication and demonstrate its multiple benefits [Samangouei et al., 2015, Samangouei et al., 2017b, Samangouei et al., 2017a]. From the semantic gap perspective, different types of interaction are possible with our proposed system. One can enroll the users

by entering the attributes manually. This is desirable because enrolling the user using conventional methods boils down to saving an inferred representation which may not be robust enough. Another type of interaction can be an understandable explanation of why the user is locked out: “Attribute A” does not match the attribute of the enrolled user. Besides user experience advantages, this has technical benefits too: the system can first check attributes which are the easier task, and then invest resources in running a complicated model.

We train binary attribute classifiers which provide compact visual descriptions of faces. The learned classifiers are applied to the image of the current user of a mobile device to extract the attributes and then authentication is done by simply comparing the calculated attributes with the enrolled attributes of the original user. Extensive experiments on two publicly available unconstrained mobile face video datasets show that our method is able to capture meaningful attributes of faces and performs better than the previously proposed authentication method. We also provide a feasible variant of our method for efficient continuous authentication on an actual mobile device by doing extensive platform evaluations of memory usage, power consumption, and authentication speed.

### 1.1.1 Bringing CNNs to mobile phones

We also use the fact that attributes decompose complicated systems into simpler ones to bring deep convolutional neural networks (CNNs) to mobile phones [Samangouei and Chellappa, 2016]. Besides the excellent performance of CNNs in

various inference tasks, the ones that are used for face verification in the wild are too computationally expensive to be deployed on mobile phones for continuous authentication. We follow our initial approach for performing attribute-based authentication and bring deep efficient CNNs to devices. The proposed CNNs implement multi-tasking and multi-view approaches. The multi-tasking approach is needed to predict multiple attributes from the same backbone network. multi-view approach enables the processing of different components of the face, such as the eyes or mouth.

Our multi-task, part-based CNN architecture for attribute detection performs better than previously proposed methods in terms of accuracy for attribute prediction. As a byproduct of the proposed architecture, we are able to explore the embedding space of the attributes extracted from different facial parts, such as the mouth and eyes, to discover new attributes. Furthermore, through extensive experimentation, we show that the attribute features extracted by our method outperform our initial presented attribute-based method and a baseline method for the task of active authentication. Lastly, we demonstrate the effectiveness of the proposed architecture in terms of speed and power consumption by deploying it on an actual mobile device.

## 1.2 Conditional image syndissertation from attribute representations

As the number of training examples increases for training attribute detectors, there is a higher chance of missing or having a low number of examples for a set of attribute combinations. To address this issue, we try to disentangle the attribute

information from invariant information using generative models. As a result, we can transfer the attribute from one image to the other or change the existence of multiple attributes at the same time. Our method falls under the umbrella of conditional image syndissertation. Aside from this application, such methods can be used for semantic image manipulation control the existence of specific attributes in the image.

Early works on conditional image generation have represented an attribute with a single dimension [Lample et al., 2017, Choi et al., 2017] and allotted a representation for non-attribute characteristics of the image. These methods consists an encoder  $f(X) \rightarrow z$  for mapping the images into latent codes  $z$  and a generator  $g(z, y) \rightarrow \hat{X}$  that gets  $z$  and attribute labels  $y$  and generates the image  $\hat{X}$ . Transferring attributes of image  $X_1$  to  $X_2$  is therefore trivial and happens by generating the image  $g(z_2, y_1)$ . Although these methods can toggle the presence of attributes, they have no sense of the style of attributes. Style of an attribute can be thought of as the attribute of the attribute. For example in the case of face images, the attribute “smiling” happens either with either of these styles: mouth open or closed. This happens because the encoder is encouraged to output a representation  $z$  that has no information about the attributes. This results in a generator that couples the “style” of the attributes to the  $z$  vector. Thus, when toggling the “smiling” dimension from “off” to “on” for some input image, the generator has no way of knowing which type of smile to put on the face.

We introduce a new method for conditional image generation based on attribute manipulation. As mentioned before, recently introduced algorithms explic-

itly encode images as a combination of an embedding vector and a series of discrete binary attributes which encode whether a visual attribute is present or not in a given image. By encoding a query image and flipping these binary attributes, visual elements can be toggled, e.g., turned on or off. Unfortunately, such an approach is limited to representing the presence, or lack thereof, of a particular, categorical attribute and cannot be used to sample from a variety of attribute realizations. For example, this approach might be used to specify whether or not a person is wearing a hat, but cannot be used to sample from a variety of hats. To address this limitation, we introduce a model for attribute-based image manipulation that represents visual attributes in a continuous embedding space. This approach allows for two new types of attribute-based manipulations: Diverse Swaps and Borrows. Diverse Swaps allows us to turn on and off visual attributes by sampling and producing diverse results. A Borrow allows us to encode a particular realization of an attribute from a reference image and inject it into a query image. We demonstrate the efficacy of our method on a challenging dataset.

### 1.3 Explaining attribute detectors

Attribute detectors can be studied themselves in terms of the semantic gap. The semantic gap can be investigated from multiple views. In this dissertation we are interested in looking at this problem from another viewpoint: given a black box function  $f_\theta$  (an attribute detector for example), “what” does it do with respect to a given input? If we can understand what a classifier is doing, we know how we

should change the input image to change the decision of the classifier. Therefore, answering the question boils to understanding what pixel values we should change, and how we should change those values to change the output of  $f_\theta$ . Therefore one expects an “explanation” algorithm to produce a saliency map and also a pixel value map that together crosses the decision boundary of the black box classifier  $f_\theta$ . One other component of “explanation” is the context that we want to explain  $f_\theta$  in. In this dissertation, we focus on explanations that are meant for humans.

Previous methods have focused on the gradient of the classification loss function  $f_\theta$  with respect to the input [Simonyan and Zisserman, 2014]. Such solutions have some limitations. First of all, the gradient values depend on the loss function which is not a part of  $f_\theta$  and may not necessarily be the same loss function that  $f_\theta$  is trained with. The second fundamental problem comes from the fact that gradients are first-order approximation of highly nonlinear functions. Therefore, as we get further away from the input image  $X$  the gradient directions become less accurate in changing the loss value. Besides conceptual problems, there are also technical issues such as the gradient vanishing phenomena that occur when  $f_\theta$  is a deep architecture. Because of these issues, the saliency maps that come out of these methods are not visually comprehensible by humans. Although there have been attempts to make the produced saliency map more visually plausible [Fong and Vedaldi, 2017], existing works provide no means of changing the pixel values so that the decision of  $f_\theta$  is changed.

We introduce a new method for interpreting computer vision models: visually perceptible, decision-boundary crossing transformations [Samangouei et al., 2018b].



Our goal is to answer a simple question: why did a model classify an image as being of class A instead of class B? Existing approaches to model interpretation, including saliency and explanation-by-nearest neighbor, fail to explicitly illustrate examples of transformations required for a specific input to alter a model’s prediction. On the other hand, algorithms for creating decision-boundary crossing transformations (e.g., adversarial examples) produce differences that are visually imperceptible and do not enable insightful explanation. To address this we introduce ExplainGAN, a generative model that produces visually perceptible decision-boundary crossing transformations. These transformations provide high-level conceptual insights which illustrate how a model makes decisions. We validate our model using both traditional quantitative interpretation metrics and introduce a new validation scheme for such an approach.

## 1.4 Protecting against induced semantic gap

The goal of adversarial attacks is to change the input image in a way that decision of the target classifier changes. These types of attacks are carried on usually by solving an optimization problem with respect to the input image and an adversarial loss function. The existing defense methods [Goodfellow et al., 2014b, Meng and Chen, 2017] assume prior information about the type of attack that is going to happen and thus is limited to the type of attack they are exploring.

We propose Defense-GAN [Samangouei et al., 2018a], a new framework leveraging the expressive capability of generative models to protect deep neural networks

against such attacks. Defense-GAN is trained to model the distribution of unperturbed images. At inference time, it finds a close output to a given image which does not contain the adversarial changes. This output is then fed to the classifier. Our proposed method can be used with any classification model and does not modify the classifier structure or training procedure. It can also be used as a defense against any attack as it does not assume knowledge of the process for generating the adversarial examples. We empirically show that Defense-GAN is consistently effective against different attack methods and improves on existing defense strategies.

## 1.5 Organization

The organization of this dissertation is as follows. Chapter 2 talks discusses attribute-based continuous authentication and a deep model for authenticating on mobile phones. In Chapter 3 we propose the conditional model for manipulating semantic attributes. Chapter 4 introduces our method for explaining black box attribute detectors. Chapter 5 presents our method for protecting the attribute classifiers against adversarial attacks. Finally, Chapter 6 concludes the dissertation and discuss future research directions.

## Chapter 2: Attribute-based continuous active authentication

### 2.1 Introduction

Attributes are semantic entities that are easier to learn than individual classes, however, a collection of them intuitively and in practice [Kumar et al., 2009], is beneficial for discriminating the object classes. We exploit this fact to design efficient methods for attribute detection. To test the efficiency, we focus on continuous authentication on mobile devices which have constraints on computation and power resources. Mobile devices, such as cell phones, tablets, and smart watches have become integral components of people’s lives. The users often store valuable information such as bank account details or credentials to access their sensitive accounts on their mobile phones. Typical devices integrate no automatic mechanism to authenticate the users. According to the survey in [Inc, 2013], nearly half of the users do not have any form of authentication for their phones. Besides this, if the initial password-based authentication is compromised, the personal information of the users is exposed since there are no active authentication methods incorporated in the mobile phone.

In the first part of this chapter, we focus on demonstrating that attribute features are practical for active authentication on mobile devices as a part of DARPA’s

active authentication program. We are the first ones to propose using attributes as an intuitive, effective, and efficient way for authentication users on smartphones. We train support vector machines (SVMs) classifiers on conventional features such as Histogram of Oriented Gradients (HOG) [Dalal and Triggs, 2005] and Local Binary Pattern (LBP) [Ahonen et al., 2006] features on different face parts and fuse their score for final attribute response. We show the feasibility of this approach by implementing it on a smartphone and testing the speed and battery consumption of different parts.

In the second part, with the emergence of a large-scale face attribute dataset CelebA [Liu et al., 2015], we train convolutional neural networks for attribute detection. An exclusive CNN per attribute is impractical to use on mobile devices since they consume a lot of resources. In the second part of this section, we bring CNN’s to cell phones by exploiting the fact that each attribute can correspond to specific regions of the face. We use lightweight deep networks for each predefined area of the face that are obtained by cropping around facial landmarks. Instead of learning a single model per attribute, we use multi-task training to share the base networks for each face region and their assigned attributes. Since an attribute can be assigned to multiple regions of the face, we get the final prediction by fusing the features from different parts. We also deploy these networks on a cell phone and show their response time, and power consumption is reasonable when they are using the phone’s CPU to compute the attribute features.

The rest of this chapter is organized as follows. In section 2.2 we go over some relevant works to this chapter. We talk about attribute-based authentication

in Section 2.3. In Section 2.4 we describe how to train efficient deep architectures for attribute detection.

## 2.2 Related work

In computer vision, almost in all problems, the very first step is to extract features from a given visual signal. The first use of attributes as higher order features was introduced in Content Based Image Retrieval where they are presented as a solution to decrease the semantic gap [Liu et al., 2007, Datta et al., 2005, Obeid et al., 2001]. Attributes were also referred to as a kind of “intermediate features”. This term initially appeared in [Obeid et al., 2001] referred to the features that are “low-level” semantic features but “high level” image features.

Later applications of attributes were in object recognition domain and human identification. Ferrari et al. [Ferrari and Zisserman, 2007] learned visual attributes for objects such as “dotted” or “striped”. In [Farhadi et al., 2009] Farhadi et al. use L1-regularized logistic regression to learn object attributes such as “has wheels” or “metallic” from images of PASCAL VOC 2008 [Everingham et al., 2008] and then use them to describe objects in the image. In [Lampert et al., 2014] Lampert et al. learn object attributes via kernel Support Vector Machines (SVMs) [Cortes and Vapnik, 1995] in two learning paradigms, Direct and Indirect Attribute Prediction and then use those to perform object recognition. They demonstrate good results on their Animal with Attributes dataset. There are several other areas where attribute features have been shown to be useful: zero-shot learning [Liu et al., 2014], scene

classification [Patterson and Hays, 2012], and action recognition [Liu et al., 2011].

Human attributes or “soft biometrics” such as age and gender suggested in [Jain et al., 2004], have been successfully used for identity recognition/verification in many applications. In [Jain et al., 2004], Jain et al. combine height, race and gender information with fingerprint to improve the recognition accuracy on an in-house dataset. Face image retrieval solely based on attributes was investigated in [Kumar et al., 2008] by Kumar et al. For face verification, Kumar et al. in [Kumar et al., 2009] extracted attribute feature vectors. Zhang et al. [Zhang et al., 2014a] used attributes to improve face clustering and overcome variations of faces like pose and illumination. Klare et al. [Klare et al., 2014] defined 46 facial attributes to perform suspect identification task. In [Layne et al., 2012] Layn et al. showed that attributes such as “jeans”, “headphones”, “sunglasses” etc. can help re-identifying people seen on different cameras of a distributed camera network. Vaquero et al. [Vaquero et al., 2009] developed a method for searching with attributes in surveillance environments using Viola-Jones attribute detectors.

Detecting the presence of each attribute has been focus of many researchers. These algorithms can be roughly divided into two groups, those which learn a specific model per attribute and those which present a general framework to learn all the target attributes together at once. Our focus in this chapter is on the second group of attributes. Bourdev et al. [Bourdev et al., 2011] define poselets-based on Histogram of Oriented Gradients (HOGs) [Dalal and Triggs, 2005] features and train SVMs on them. Zhang et al. [Zhang et al., 2014b] train a Convolutional Neural Network (CNN) on parts extracted from full body person images to detect attributes, they

achieved good results on Berkley Attributes of People dataset and Attributes25k [Bourdev et al., 2011] dataset. Berg et al. [Berg and Belhumeur, 2013] learned one SVM per class pairs and part pairs to take into account the class relationship and part relationship and then create a feature vector out of all the SVMs. Then these features were used to learn classifiers for each attribute. Kumar et al. [Kumar et al., 2008] trained their local SVMs and let Adaboost to optimize for best ones for ten attributes and show the performance on FaceTracer [Kumar et al., 2008] dataset. In [Kumar et al., 2009] Kumar et al. concatenated different low-level features extracted from face components and incrementally learn SVMs for each attribute and test them on PubFig [Kumar et al., 2009]. We present two approaches, one consists of model selection between different SVMs, and another simpler one which gives efficient linear SVMs for platform implementation.

The early research to find alternatives for password-based authentication were focused on extracting unique characteristics from users' keystrokes. In [Spillane, 1975], Spillane et al. suggested to use timing between key presses and the pressure patterns of keystrokes to identify users. Then in [Monrose et al., 2002] Monrose et al. created a method using pseudorandom polynomials to generate a secure sequence based on keystroke time interval of users to increase password security. In [Klosterman and Ganger, 2000], Klosterman et al. introduce the first continuous face verification system implemented in Linux. They also present a comprehensive set of differences between biometric and password-based authentication systems. The next biometric based continuous authentication system design was introduced by Carrillo [Carrillo, 2003] to secure aircraft cockpit against unverified access. Then

followed many studies on continuous authentication mostly for desktop computers like [Altinok and Turk, 2003, Sim et al., 2007, Niinuma and Jain, 2010, Niinuma et al., 2010, Janakiraman et al., 2005].

With exponential growth in the use of mobile devices, active authentication on them has become the focus of many researchers. Various biometrics have been proposed to continuously authenticate the users. In [Frank et al., 2013] Frank et al. proposed a set of 30 behavioral touch features and then use a k-nearest neighbor classifier and Gaussian kernel SVM for horizontal and vertical strokes of the user to perform authentication. [Feng et al., 2012], [Zhang et al., 2015b] also use touch-screen gestures for this purpose. Gait as well as device movement patterns measured by the smartphone accelerometer were used in [Derawi et al., 2010], [Primo et al., 2014] for continuous authentication. Stylometry, GPS location, web browsing behavior, and application usage patterns were used in [Fridman et al., 2015] for active authentication.

Face-based continuous user authentication has also been under study by researchers. In [Hadid et al., 2007] Hadid et al. use Haar-like features and Adaboost of [Viola et al., 2005] by Viola et al. is employed for part detection and, Local Binary Pattern (LBP) [Ahonen et al., 2006] followed by nearest neighbor thresholding for identification. In [Fathy et al., 2015], Fathy et al. extracted two intensity features for images, one from the whole face and one from face components. Then they compare four still image algorithms and five convex hull image set comparison methods for the AA01 dataset and compared the recognition rates of the algorithms. Lastly, [Gunther et al., 2013] Gunther et al. provides an overview of methods that



depend on low-level features for this task such as [Zhao et al., 1998], [Cox and Pinto, 2011], [Zhang et al., 2005], [Wiskott et al., 1997] and their results on the MOBIO [McCool et al., 2012] dataset.

Multi-modal methods have always been of interest when it comes to biometrics. Fusion of speech and face was proposed in [McCool et al., 2012] by McCool et al, they extract LBP features and use nearest neighbor thresholding for faces. [Crouse et al., 2015] proposed to fuse face images with the inertial measurement unit data to continuously authenticate the users. A low-rank representation-based method was proposed in [Zhang et al., 2015a] for fusing touch gestures with faces for continuous authentication. Finally, a domain adaptation method was proposed in [Zhang et al., 2015c] for dealing with data mismatch problem in continuous authentication.

Face-based continuous user authentication has also been proposed in [Hadid et al., 2007], [Fathy et al., 2015], [McCool et al., 2012], [Samangouei et al., 2015]. Fusion of speech and face was proposed in [McCool et al., 2012] while [Crouse et al., 2015] proposed to fuse face images with the inertial measurement unit data to continuously authenticate the users. Finally, a domain adaptation method was proposed in [Zhang et al., 2015c] for dealing with data mismatch problem in continuous authentication. Fusing touch gestures with faces for continuous authentication using a low-rank representation-based method was proposed in [Zhang et al., 2015a].

State of the art methods for face recognition employ Deep Convolutional Neural Networks (DCNN) [Taigman et al., 2014], [Parkhi et al., 2015], [Schroff et al., 2015], [Sun et al., 2014a]. Since deep networks are very scalable, they achieve good

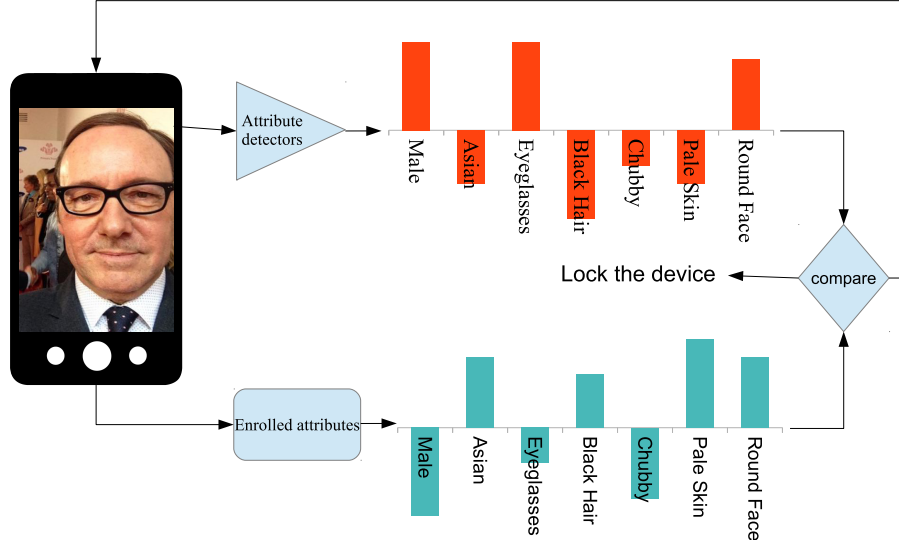


Figure 2.1: Overview of our attribute-based authentication method.

results by having large number of parameters learned using large datasets. The harder the problem, the more number of parameters and data are required. As a result of their size, their architectures are not efficient for to be deployed on a mobile phone. It has been shown in [Sarkar et al., 2016] that DCNN with an architecture similar to AlexNet [Krizhevsky et al., 2012] can drain the battery very fast.

### 2.3 Attribute-based Active Authentication on Mobile Devices

In this section, we present the details of the proposed attribute-based authentication system. In particular, we describe the training data used to learn the attribute classifiers, how different classifiers are trained for each attribute and how verification is performed using the attributes.

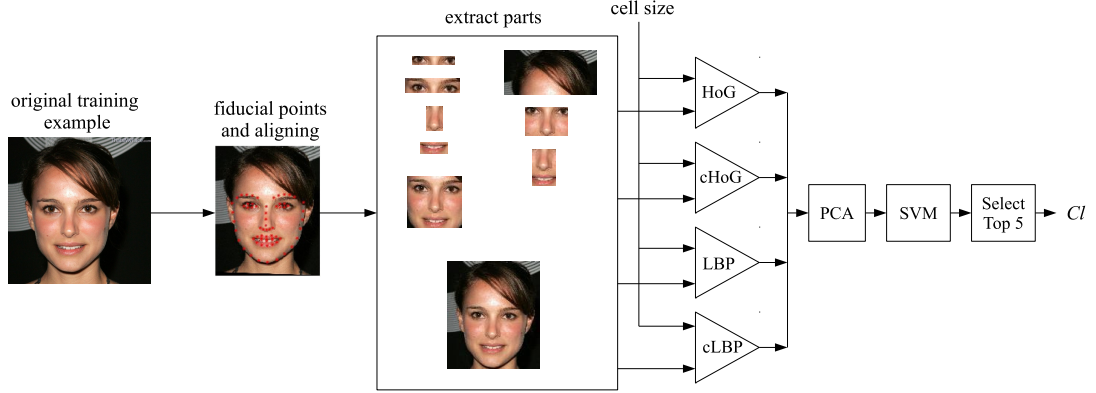


Figure 2.2: Training phase pipeline for each attribute classifier. Landmarks are first detected on a given face. Different facial components are then extracted from these landmarks. Then for each part, features are extracted with different cell sizes and the dimensionality of features is reduced using principle component analysis. Classifiers are then learned on these low-dimensional features. Finally, top five  $Cl$ s are selected as our attribute classifier.

### 2.3.1 Methodology

**Training Data** PubFig dataset [Kumar et al., 2009] is one of the few publicly available datasets that provides facial attributes along with face images. We use this dataset to train our attribute classifiers. PubFig dataset consists of unconstrained faces collected from the Internet by using a person’s name as the search query on a variety of image search engines, such as Google Images and flickr. However, there are several challenges have to be overcome before this dataset can be effectively utilized for our application. Since the release of this dataset in 2009, many links to the images in this dataset are broken. Hence, not all the images listed in this dataset are available for downloading. As a result, we use a subset of this dataset where we could establish proper links to the images. Furthermore, the true attribute labels of the images are not provided, instead the output of their attribute classifiers are

provided. As a result, we used a proper threshold to get the labels for each attribute of the available images to ensure that the classifier is certain enough about the label given to the image. Finally, rather than using all the 73 binary attributes in the PubFig dataset, we selected a more meaningful subset of 44 attributes in our implementation.

FaceTracer [Kumar et al., 2008] is another publicly available dataset that has face images with 18 attributes. This dataset is smaller than the PubFig dataset and again a several hyperlinks to the images in this dataset are broken. Also, only a subset of attribute labels has been provided.

### 2.3.2 Attributes Classifiers

Each attribute classifier  $Cl_i \in \{Cl_1, \dots, Cl_N\}$  is trained by an automatic procedure of model selection for each attribute  $A_i \in \{A_1, \dots, A_N\}$ , where  $N$  is the total number of attributes. Automatic selection is necessary since each attribute needs a different model. Our models are indexed as follows:

- 1 **Facial parts:** For each attribute, a set of different facial components can be more discriminative. The face components considered for training are: *eyes*, *nose*, *mouth*, *hair*, *eyes&nose*, *mouth&nose*, *eyes&nose&mouth*, *eyes&eyebrows*, and the *full* face. In total, nine different face components are considered.

- 2 **Features:** For different attributes, different types of features may be needed. For example, for the attribute “blond hair”, features related to color can be more discriminative than features related to texture. The following features

are considered in this work: *LBP* [Ahonen et al., 2006], *ColorLBP*, *HoG* [Dalal and Triggs, 2005], and *ColorHoG*. ColorLBP and ColorHOG are obtained by concatenating the HoG/LBP feature of each RGB channel. In total, four types of features are extracted using the VLFeat toolbox [Vedaldi and Fulkerson, 2008].

**3 Locality of features:** In order to capture the local information, we consider different cell sizes for the HOG and LBP features. In total, six different cell sizes, 6, 8, 12, 16, 24, 32, are used.

The implementation of the algorithm for this section is done in Matlab [MATLAB, 2014]. We use a state-of-the-art publicly available fiducial point detection method [Asthana et al., 2013] to extract the different facial components. Furthermore, the detected landmarks are also used to align the faces to a canonical coordinate system. After extracting each set of features, the Principal component analysis (PCA) is used with 99% of the energy to project each feature onto a low-dimensional subspace. An SVM with the RBF kernel is then learned on these features. This process is run exhaustively to train all possible models. For each attribute classifier, 80% of the available data is used for training the SVMs and 20% of the data is used for model selection. The face images in the test set do not overlap with those in the training set. The total number of negative and positive classes are the same for both training and testing. Finally, among all 216 SVMs, five with the best accuracies are selected.

For a given test face image  $F$ , a feature vector  $[f_{a_1} \dots f_{a_N}]$  is calculated by

$$f_{a_k} = \frac{\sum_{i=1}^5 w_k^i Cl_k^i(F)}{\sum_{i=1}^5 w_k^i}, \quad (2.1)$$

where  $Cl_k^i(F) \rightarrow \{0, 1\}$  is the output of the  $i$ th accurate classifier for the  $k$ th attribute  $A_k$  on face image  $F$ , and  $w_i$  is the accuracy of  $Cl_k^i$ . The entire training pipeline of our method is shown in Figure 2.2.

### 2.3.3 Verification

We consider the continuous authentication problem as a verification problem in which given two pairs of videos or images, we determine whether they correspond to the same person or not. The well-known receiver operating characteristic (ROC) curve, which describes the relations between false acceptance rates (FARs) and true acceptance rates (TARs), is used to evaluate the performance of verification algorithms. As the TAR increases, so does the FAR. Therefore, one would expect an ideal verification framework to have TARs all equal to 1 for any FARs. The ROC curves can be computed given a similarity matrix.

We use the proposed framework to extract the attribute vector from each image in a given video. We then simply average them to obtain a single attribute vector that represents the entire video. Then, the  $(i, j)$  entry of the similarity matrix  $S_{attrs}$  is calculated as

$$s_{i,j} = \frac{1}{\|\mathbf{e}_i - \mathbf{t}_j\|_2}, \quad (2.2)$$

where  $\mathbf{e}_i$  is the  $i$ th attribute vector representing the gallery (or enrollment) video, and  $\mathbf{t}_j$  is the  $j$ th attribute vector representing the probe video. We evaluate the performance of the proposed attribute-based authentication method on two publicly available mobile video datasets - MOBIO [McCool et al., 2012] and AA01 [Fathy et al., 2015]. In addition to the ROC curves, the Equal Error Rate (EER) is used to measure the performance of different methods. The EER is the error rate at which the probability of false acceptance rate is equal to the probability of false rejection rate. The lower the EER value, the higher the accuracy of the authentication system.

We use an LBP-based method as a baseline for comparison. In this method, each detected face is represented by the histogram of LBP features. The same aligned faces that are used for attribute feature extraction are also used to extract the LBP features. Similar to the attribute features, the LBP features from each image in a video are extracted and averaged to represent a single video. The LBP features are extracted using the VLfeat toolbox. The similarity matrix,  $S_{LBP}$ , is then built by comparing two feature vectors. This LBP-based method has been used for mobile face authentication in [McCool et al., 2012] and [Hadid et al., 2007]. A third fusion score matrix,  $S_{fusion} = \tilde{S}_{LBP} + \tilde{S}_{attrs}$ , is calculated by  $z$ -score normalization

$$\tilde{s}_{i,j} = \frac{s_{i,j} - \bar{S}}{\sigma(S)}, \quad (2.3)$$

where  $\bar{S}$  and  $\sigma(S)$  are the mean and the standard deviation of the entries in similarity matrix  $S$ , respectively.





Attribute	Accuracy	Attribute	Accuracy
Blond Hair	0.9089	Child	0.9538
Partially Visible Forehead	0.8645	Narrow Eyes	0.7777
Round Face	0.9156	Big Nose	0.8039
Indian	0.9714	Male	0.9451
Gray Hair	0.9091	Pointy Nose	0.816
Bags Under Eyes	0.8986	Asian	0.9225
Obstructed Forehead	0.8913	White	0.6992
Shiny Skin	0.9532	Youth	0.7299
No Eyewear	0.8875	Brown Hair	0.6725
Middle Aged	0.929	Bald	0.7909
Senior	0.8867	Wavy Hair	0.9357
Eyeglasses	0.9397	Straight Hair	0.7408
Sunglasses	0.9701	Bangs	0.9397
Mustache	0.8606	Arched Eyebrows	0.6462
Chubby	0.8815	Strong Lines	0.9308
Receding Hairline	0.8164	Pale Skin	0.793
Round Jaw	0.9357	Flushed Face	0.7819
Big Lips	0.7578	Double Chin	0.9727
No Beard	0.7766	Black Hair	0.8029
Goatee	0.9775	Curly Hair	0.8746
Black	0.7818	Bushy Eyebrows	0.836
Sideburns	0.8756	Oval Face	0.82

Table 2.1: Accuracies of the 44 attribute classifiers proposed in this work on the PubFig dataset [Kumar et al., 2009].

classifiers give high scores. This clearly matches with the image shown on the left. For the second face, it is interesting to see that the Male classifier produces a negative score since the image corresponds to a female subject. Finally, for the last face, “mustache”, “goatee”, “chubby” and “bags under eyes” produce high positive scores which clearly match with the image shown on the left.

Attribute	Accuracy	Attribute	Accuracy
Asian	0.8786	middle aged	0.7321
eyeglasses	0.7214	black	0.808
sunglasses	0.89	female	0.88
smiling false	0.8	senior	0.7933
no eyewear	0.7481	hair color blond	0.7875
child	0.8276	white	0.763
mustache	0.815	youth	0.692

Table 2.2: Accuracies of the attribute classifiers proposed in this work on available attributes on the FaceTracer dataset [Kumar et al., 2008].

Attribute	Indoor	Lights off	Outdoor	Attribute	Indoor	Lights off	Outdoor
Asian	0.64	0.62	0.54	Bags Under Eyes	0.96	0.96	0.96
Bald	0.98	0.98	0.98	Bangs	0.88	0.88	0.88
Big Lips	0.80	0.80	0.80	Big Nose	0.90	0.90	0.92
Black	0.98	0.98	0.98	Black Hair	0.62	0.62	0.72
Blond Hair	0.96	0.96	0.96	Brown Hair	0.96	0.96	0.96
Bushy Eyebrows	0.94	0.94	0.94	Child	0.74	0.76	0.78
Chubby	0.74	0.74	0.76	Curly Hair	0.96	0.96	0.96
Double Chin	0.92	0.94	0.94	Eyeglasses	0.60	0.58	0.58
Flushed Face	0.98	0.98	0.98	Goatee	0.96	0.96	0.96
Gray Hair	0.96	0.96	0.96	Indian	0.86	0.86	0.86
Male	0.82	0.82	0.84	Middle Aged	0.96	0.96	0.96
Mustache	0.86	0.86	0.86	Narrow Eyes	0.64	0.68	0.62
No Beard	0.58	0.56	0.58	No Eyewear	0.74	0.74	0.74
Obstructed Forehead	0.84	0.84	0.88	Oval Face	0.78	0.78	0.78
Pale Skin	0.98	0.98	0.98	Partially Visible Forehead	0.70	0.70	0.70
Pointy Nose	0.98	0.98	0.98	Receding Hairline	0.86	0.86	0.90
Round Face	0.96	0.96	0.96	Round Jaw	0.76	0.74	0.76
Senior	0.98	0.98	0.98	Shiny Skin	0.98	0.98	0.98
Sideburns	0.98	0.98	0.98	Straight Hair	0.72	0.72	0.74
Strong Nose-Mouth Lines	0.82	0.82	0.82	Sunglasses	0.98	0.98	0.98
Wavy Hair	0.96	0.96	0.96	White	0.90	0.90	0.90

Table 2.3: Accuracy of the attribute classifiers for CNNA [Fathy et al., 2015] dataset.

### 2.3.5 MOBIO Dataset

The MOBIO dataset [McCool et al., 2012] consists of video data taken from 152 subjects. The dataset was collected in six different sites from five different countries. In total twelve sessions were captured for each subject - six sessions for phase 1 and six sessions for phase 2. The database was recorded using two mobile devices: a NOKIA N93i mobile phone and a standard 2008 MacBook laptop computer. The laptop was only used to capture videos of part of the first session. So the first session consists of data captured with both the laptop and the mobile phone. Figure 2.4 shows some frames from the MOBIO dataset.



Figure 2.4: Sample images from the MOBIO dataset. One can clearly see the different illumination conditions in this dataset.

In the MOBIO protocol, for each person, the data from one session is used for enrollment and the data from the remaining sessions are used for testing. In the first set of experiments with the MOBIO dataset, we do not consider the data from the laptop session. The first mobile session is considered as the enrollment session and the data from the next 11 sessions are considered for testing. The ROC curves

Site	LBP	Attributes	Fusion
but	0.29	0.28	<b>0.25</b>
idiap	0.18	0.20	<b>0.14</b>
lia	0.31	<b>0.24</b>	0.25
uman	0.20	0.25	<b>0.18</b>
unis	0.24	0.28	<b>0.24</b>
uoulu	0.27	0.24	<b>0.23</b>
All together	0.22	0.23	<b>0.19</b>

Table 2.4: The EER values for different methods on the MOBIO dataset.

corresponding to this experiment are shown in Figure 2.5 for the entire dataset. As can be seen from this figure, our attribute-based method performs comparably to the LBP-based methods. However, the best performance is achieved when the similarity matrices corresponding to the LBP and attribute features are fused. The EER values corresponding to this experiment are compared in Table 2.4.

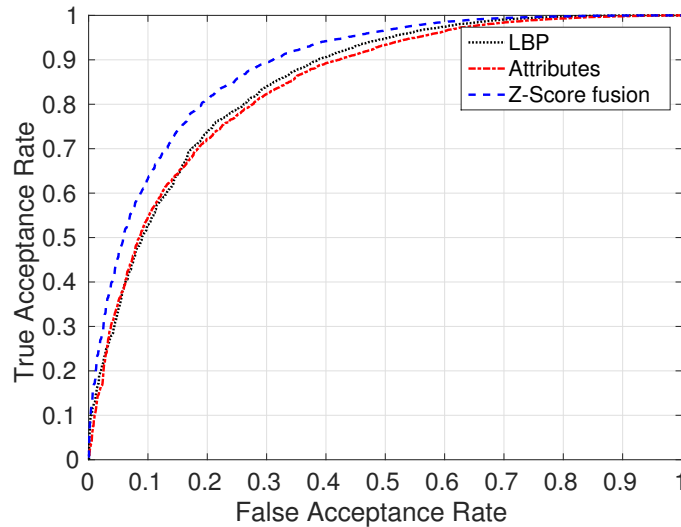


Figure 2.5: Performance evaluation on the MOBIO dataset.

### 2.3.6 Cross-device Experiments

Images captured by different cameras have different characteristics. Since the MOBIO dataset has videos that were captured using different sensors, we conduct cross-session experiments in which the data from the laptop session are considered as the enrollment data and the data from the cell phone are used as the test videos. This experiment essentially allows us to study the robustness of different algorithms with respect to different image quality. Figure 2.6 and Table 2.5 show the ROC curves and the EER values corresponding to this experiment. As can be seen from this results, attributes are more robust to camera sensor change than LBP features. In this experiment, fusion does not necessarily improve the performance over the attributes since LBP features perform poorly.

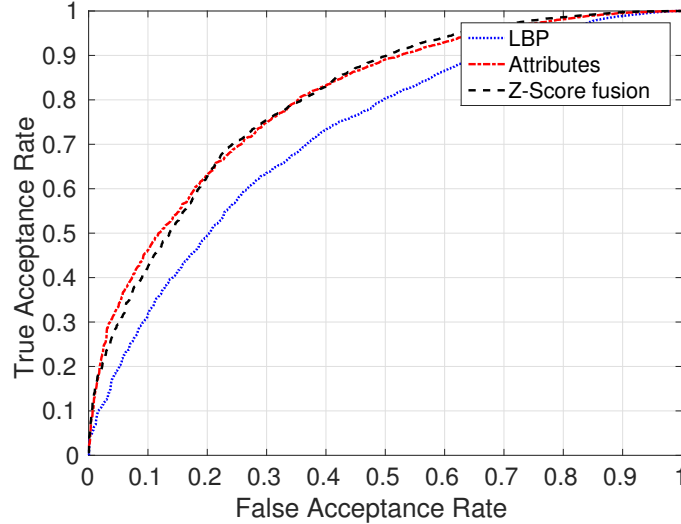


Figure 2.6: Cross device robustness. Laptop session videos are used for enrollment and the data from the remaining sessions are used for testing.

Enrollment	LBP	Attributes	Fusion
Laptop	0.33	<b>0.27</b>	0.27

Table 2.5: The EER values corresponding to the cross-device experiment on the MOBIO dataset.

### 2.3.7 AA01 Dataset

The AA01 dataset consists of 750 videos from 50 different individuals collected in three different sessions corresponding to three different illumination conditions. The UMDAA-01 dataset was collected using an app on an iPhone 5s. Each user performed five tasks in three sessions. The different tasks were enrollment task, document task, picture task, popup task and scrolling task. Figure 2.7 shows some sample images from the UMDAA-01 dataset where one can clearly see the different illumination conditions present in this dataset.

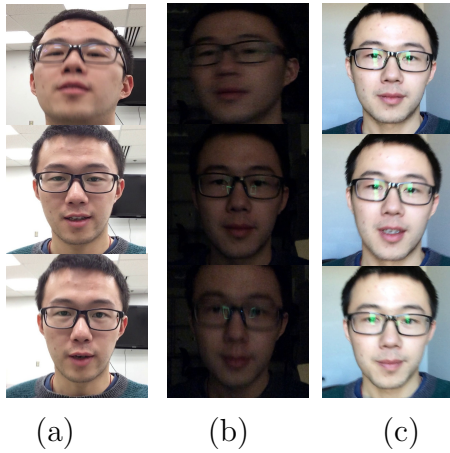


Figure 2.7: Sample images from the AA01 dataset. (a), (b) and (c) show some sample images from session 1, 2 and 3, respectively.

In the first set of experiments using this dataset, we use the data corresponding to the enrollment task as gallery and the data from the remaining tasks for testing. Figure 2.8 and Table 2.6 show the ROC curves and the EER values, respectively

Enrollment	LBP	Attributes	Fusion
Indoor light	0.13	0.14	<b>0.10</b>
Low light	0.31	<b>0.18</b>	0.20
Natural light	0.19	0.16	<b>0.14</b>
CNNAA_all	0.34	<b>0.30</b>	0.30

Table 2.6: The EER values of different methods for the AA01 dataset.

corresponding to this experiment. As can be seen from these results, our attribute-based method performs much better than the LBP-based authentication system. Fusion of the LBP and the attribute similarity matrices results in performance comparable to our method as the LBP features do not perform well on this dataset.

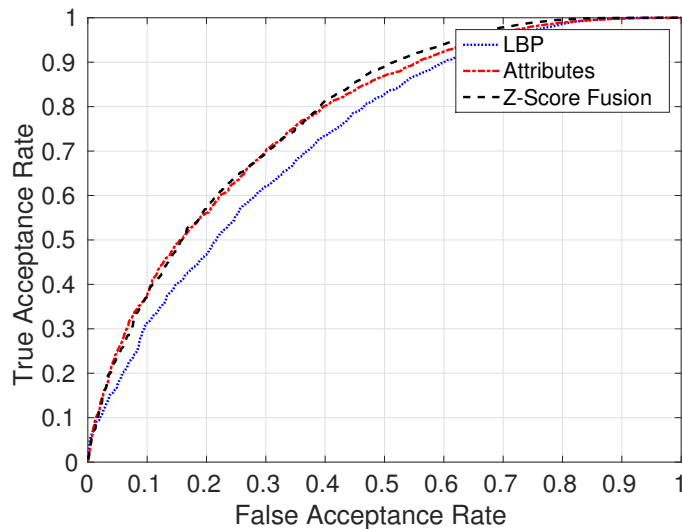


Figure 2.8: Performance evaluation for the AA01 dataset.

Furthermore, we conducted several session-specific experiments on this dataset. We used the enrollment data as gallery and the data from other tasks from the same session as probe. The ROC curves corresponding to these experiments are shown in Figures 2.9(a)-(c). It can be seen from these figures that our attribute-based method works better than the LBP-based method, and fusion improves the result

as expected. The reason that attributes work better here is that the sessions are all taken in the same day so the change in attributes are less severe than in the MOBIO dataset.

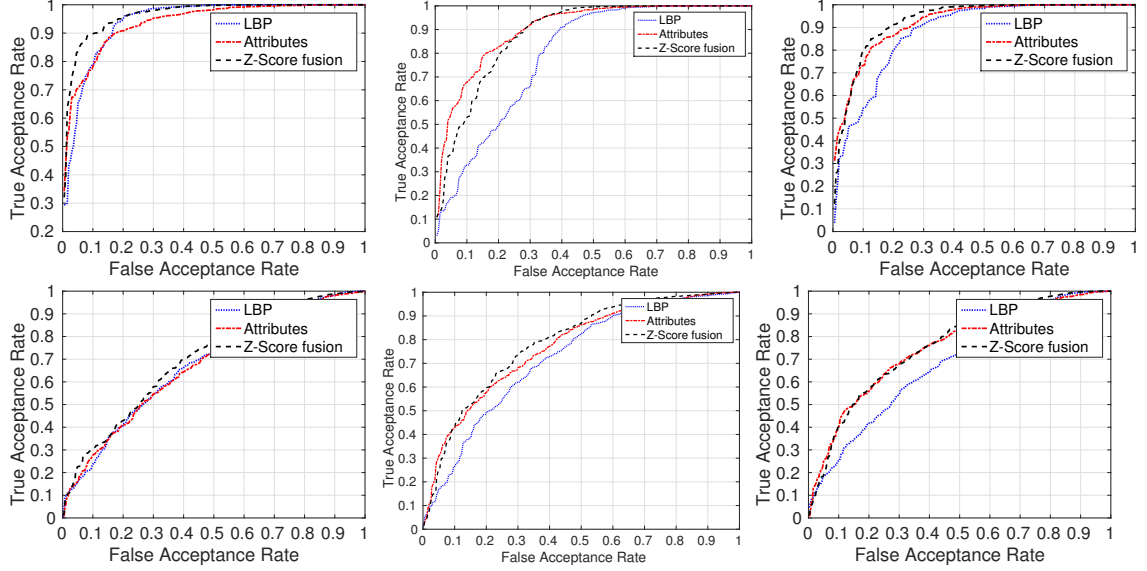


Figure 2.9: Session-specific performance evaluations for the AA01 dataset. (a) Gallery and probe data from session 1. (b) Gallery and probe data from session 2. (c) Gallery and probe data from session 3. (d) Gallery data from session 1 and probe data from sessions 2 and 3. (e) Gallery data from session 2 and probe data from sessions 1 and 3. (f) Gallery data from session 3 and probe data from sessions 2 and 1.

Finally, similar to the cross-device experiments on the MOBIO dataset, we conducted cross-session experiments on the AA01 dataset. We used the data from the enrollment task from one session as gallery and the data from the other sessions as probe. This experiment shows the robustness of our attribute-based method to different illumination conditions. From Figures 2.9(d)-(f), we see that even when the illumination conditions are different, our attribute-based method is more robust than the LBP feature-based method. From Figures 2.9(d)-(f) and Table 2.7 we see that in all cases, attributes performed better than LBP and the fusion of both gives



Gallery→Probe	LBP	Attributes	Fusion
1 → 2, 3	0.36	0.33	<b>0.32</b>
2 → 1, 3	0.35	0.31	<b>0.30</b>
3 → 1, 2	0.38	0.33	<b>0.31</b>

Table 2.7: The EER values corresponding to the cross-session experiments for the AA01 dataset. 1 is the office light session, 2 is the low light session, 3 is the natural light session.

the best results.

### 2.3.8 Platform implementation and evaluations

One set of the challenges of continuous mobile authentication is the computational complexity and memory usage of the algorithm. The limited computation capacity of a mobile phone is shared among many processes. So if the algorithm takes most of the CPU time, other processes will slow down. Also, computations consume energy. The more complex they are, the sooner the battery of the phone needs to be recharged. In addition, the memory capacity of the phones are limited. Algorithms with high memory usage, will force other running processes to go in the swap memory of the phone. This costly I/O operations results in both slow down and high power consumption.

As a consequence, algorithms with high complexity are run on a server and the mobile device is used just as a client that takes pictures, sends them to the server and waits for the response as suggested in [Takacs et al., 2008]. This solution has two drawbacks for continuous authentication. The phone will get locked if the

mobile device gets disconnected from the server. Furthermore, the system will be less secure since the communication between mobile and server can be interfered. This can result in either locking the device of the victim, or even worse unlocking it by creating a fake server which responds in a way that keeps the phone unlocked. Also, depending on the enrollment policy, we may need to re-enroll the user multiple times to account for changes in appearance or environment after the first enrollment to create a better template. It will also take time to re-enroll the user on the server again. This will be an unproductive experience for the user. In this section, we show that our approach allows enrollment and authentication of the user on the device.

Our implementation is tested on a Google Nexus 5 with 2GB of RAM and a quad core 2.2GHz CPU. The implementation is done on the Android operating system using the well-known OpenCV [[Bradski, 2000](#)] library. Since the authentication should be done continuously, efficiency-accuracy trade-off will become very important. To explore this trade-off, we looked at three measures: memory, running time, and power consumption. Fully changing all the parameters and performing evaluations is out of the scope of the research, but we will present one pathway to platform implementation which highlights the decisions that impact the efficiency-accuracy trade-off.

Learning method	PCA+RBFSVM	RBFSVM	Linear SVM
Average memory usage	80MB	54MB	1MB

Table 2.8: Average memory usage per attribute classifier for full face

### 2.3.9 Memory

Memory usage or spacial complexity has always been a challenge while implementing computer vision algorithms. We changed the last two steps of our learning method to evaluate the memory usage of different models. The average test time memory requirement of each learning approach for attribute classifiers can be found in Table 2.8. The memory usage is calculated by first loading all the attribute classifiers and looking at the increase in memory usage and dividing that change by the number of classifiers. We use LBP features of intensity image and RGB channels in this experiment on  $128 \times 168$  face images, with dimensionality of 76800 per face crop. In PCA, we keep 90% of energy. As can be seen from Table 2.8, since we have 44 classifiers at least, using PCA for dimensionality reduction or RBF kernel will need more than 2GB of memory in total. So we focus on linear SVMs for attributes.

### 2.3.10 Final attribute classifiers for platform

By looking at the memory usage per classifier given in Table 2.8, we have no choice but to simplify our classifier learning framework. For training the classifiers, we use the LFW [Huang et al., 2007] dataset. It has more subjects, hence containing more variations for each attribute. Also, the output of classifiers from [Kumar et al.,

[2009] is available for the LFW dataset, so to train our classifiers, the same framework as in Section 2.3.1 is followed with some changes. Since we can not afford 5 PCA-RBFSVM per attribute, our goal is to train the least number of classifiers possible which gives us the desired accuracy.

We simplify our training procedure to learn one single linear SVM for each attribute while trying to consider challenges of learning attribute classifiers addressed in Section 2.3.1. We reran the experiments for the AA01 dataset from Section 2.3.7 with the linear classifiers learned with our simplified learning procedure. The resulting ROC curves can be seen in Figure 2.10 and the corresponding EER values in 2.9. The differences with classifiers of Section 2.3.1 and Figure 2.2 are:

- **Feature extraction:** In the approach discussed in Section 2.3.1, we extracted different types of features to capture the dependence of each attribute on color and scale. In the simplified model, we just extract LBP feature on gray scale image and the three channels and concatenate them together. We don't change the cell size of LBP to capture dependence on locality. Instead, we perform evaluation with different image sizes and choose the one that works best for all attributes together.
- **No PCA** No dimensionality reduction step is employed after feature extraction, because loading the PCA basis sets on a phone needs significant memory space.
- **No kernel** A linear classifier is learned, because from memory usage in Table 2.8 it is impractical to load the kernelized classifiers into memory.

- **No part-based attribute classifier** We just train classifiers on the full face image in the simplified model. Extracting the face parts from face image is not a trivial task and adds to the complexity of the model. Also linear classifiers optimize the weights that are directly related to pixel values, so choosing face parts is taken care of by SVM optimization objective to some extent.
- **Attribute dimension value** In Section 2.3.1, we took the weighted average of binary decision values of the top five attribute classifiers as the dimension value. In the simplified learning approach, we just use the distance from margin of each attribute classifier. This is valid since we trained the attribute classifiers with the same image size.

The interesting result is that the classifier with scale 0.5 performs better than the ones from Section 2.3.1. The most important one is probably the last step of the approach presented in Section 2.3.1. For the last step, we fuse the output of top the top five SVMs to get a score by taking the weighted average of binary decision values of each attribute classifier. This results in a discrete and finite range of scores. However, the scores of the simplified model are the distances from the margins of the linear classifiers which gives a continuous value and hence more discriminative range of value for different faces.

### 2.3.11 Frames per second and power consumption

Since linear attribute classifiers on the full face turned out to be the winner of accuracy-efficiency tradeoff, we test their speed and power consumption. For

Method	LBP	AttrrsSection3	L-SVM 1	L-SVM 0.7	L-SVM 0.5	L-SVM 0.3
Feature dim	19200	variable	76800	33280	16128	3840
Indoor lighting	0.13	0.14	0.16	0.19	<b>0.11</b>	0.22
Low lighting	0.31	0.18	0.20	0.20	<b>0.15</b>	0.24
Natural light	0.19	0.16	0.18	0.18	<b>0.11</b>	0.21
Altogether	0.33	0.30	0.29	0.29	<b>0.25</b>	0.37

Table 2.9: Comparison of EER values for LBP, attribute detectors of Section 2.3.1, linear models of Section 2.3.8. The scale L-SVM 1 is trained on images of size  $128 \times 168$  and the rest are scaled by the indicated value. The best EER is gained from L-SVMs of Section 2.3.8 with scale 0.5.

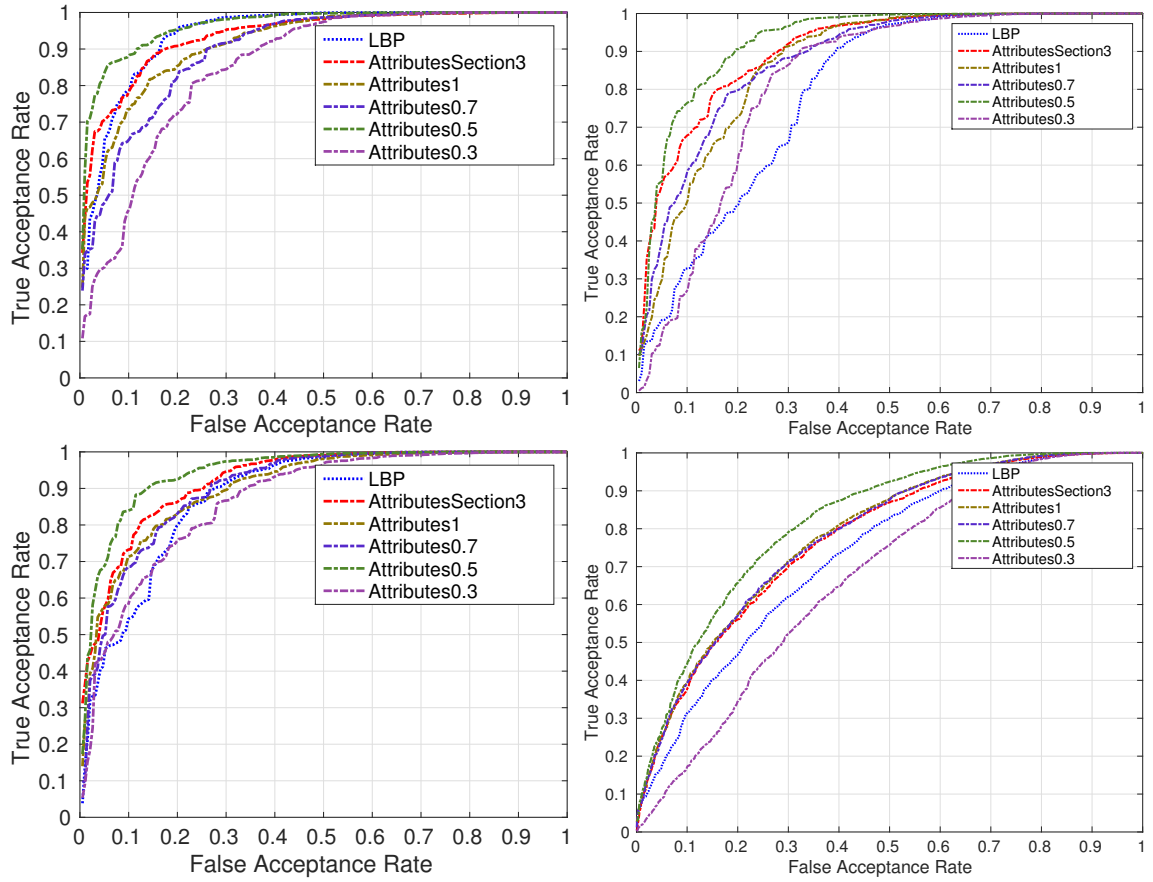


Figure 2.10: Comparison of linear SVMs with model learned in section 2.3.1 and LBP. The best result among all is achieved with linear models of scale 0.5 i.e. face crop size of  $64 \times 80$ .

speed, we look at the number of frames that we can authenticate per second and for power consumption we use the power consumption profiler presented by Zhang

et al. [Zhang et al., 2010]. We extract the 44 dimensional feature vector for 5000 frames with each set of attribute classifiers indexed by scale. Android provides a mechanism to get the time in milliseconds, so we can measure the exact time up to milliseconds that it takes for each set of classifiers to process the 5000 frames. Also the power profiler provides the energy consumption in Joules up to  $0.1J$  for each running application. The numbers for different setups of our algorithm are provided in 2.10. The landmark detection which was done with Asthana et al. [Asthana et al., 2013] in Section 2.3.1 is replaced by the algorithm of Kazemi et al. [Kazemi and Sullivan, 2014] which is implemented using DLib [King, 2009]. This algorithm adds a 90MB to memory consumption but it is very fast. The evaluation for each scale is done in two settings, one with Haar face detection and DLib alignment and one without this step. From the table we see that face detection and alignment step add around 20mJ energy consumption per frame and reduces the FPS significantly. In the worst case with fastest available face and landmark detection methods, our algorithm can authenticate users at the speed of 4 frames per second. This is more than enough for authentication task which probably requires authenticating every couple of seconds.

Google Nexus 5 battery capacity is  $2300mAh$  and the average working voltage is  $3.8V$  which can be verified by running the power profiler of [Zhang et al., 2010]. This means that it has in total  $8740mWh$ . If we run the profiler without our authentication program on the phone for more than 5 minutes, it shows the average power usage as  $520mW$  which means the phone will last for 16.8 hours. The last row of Table 2.10 shows how many hours our algorithm can run in background if

Scale	0.3		0.5		0.7		1	
Size/dim	$32 \times 48/3840$		$64 \times 80/16128$		$88 \times 112/33280$		$128 \times 168/76800$	
Detection/Alignment	W/O	W/	W/O	W/	W/O	W/	W/O	W/
FPS	114	29	31	16	13	8	5	4
Energy	26.8J	128.9J	93.5J	201.2J	207J	369.1J	524.9J	603J
Energy per frame	5.4mJ	25.8mJ	18.7mJ	40.2mJ	41.4mJ	73.8mJ	105mJ	120.6mJ
Endurance (hours)	16.6	16	16.2	15.6	15.6	14.7	14	13.6

Table 2.10: The speed and power consumption of different realization of the classifiers learned with the simplified training framework on Google Nexus 5 device. W/O column means our algorithm extract all the attributes given aligned and cropped face. In last row we assumed that we are doing authentication with the speed 1fps. W/ column first detects the face then extracts attributes. We can authenticate 17.6 hours every second employing our classifiers with best EER of Table 2.9 on a Nexus 5.

we do authentication once every second considering these numbers.

## 2.4 Efficient Deep Features for Attribute Detection on Mobile Phones

### 2.4.1 Methodology

In the mobile setting, there is a trade-off between hardware limitations such as battery life and accuracy of the models. We design our models with the goal of balancing this trade-off. Namely, we move from a computationally expensive but specialized model to a computationally cheaper but accurate model.

We train and test four different sets of DCNNs, in total 100 of them, for the task of attribute classification on a set of face regions. We crop the functional face regions using landmarks detected by [Asthana et al., 2013]. These face regions can be seen in Table 2.13. Then for each part, we find the maximum size of the window for that part in the dataset, then we crop the regions by putting the center of the face part at the center of the crop window to avoid any scaling.



Attribute	DeepMulti-CNNAA	WideMulti-CNNAA	DeepBinary-CNNAA	WideBinary-CNNAA	FaceTracer	PANDA	LNet+ANet
5 o Clock Shadow	<b>93</b>	93	91	89	85	88	91
Arched Eyebrows	81	82	82	<b>83</b>	76	78	79
Attractive	81	81	81	<b>82</b>	78	81	81
Bags Under Eyes	83	<b>84</b>	83	82	76	79	79
Bald	<b>99</b>	99	96	98	89	96	98
Bangs	<b>95</b>	95	94	94	88	92	95
Big Lips	67	<b>70</b>	69	67	64	67	68
Big Nose	82	<b>83</b>	78	78	74	75	78
Black Hair	86	86	<b>88</b>	87	70	85	88
Blond Hair	<b>95</b>	95	94	94	80	93	95
Blurry	<b>95</b>	95	92	80	81	86	84
Brown Hair	<b>86</b>	86	86	84	60	77	80
Bushy Eyebrows	<b>92</b>	92	89	89	80	86	90
Chubby	<b>95</b>	95	87	91	86	86	91
Double Chin	<b>96</b>	96	89	93	88	88	92
Eyeglasses	<b>99</b>	99	99	99	98	98	99
Goatee	<b>97</b>	97	93	96	93	93	95
Gray Hair	<b>98</b>	98	92	97	90	94	97
Heavy Makeup	90	90	90	<b>91</b>	85	90	90
High Cheekbones	86	85	<b>87</b>	87	84	86	87
Male	<b>98</b>	97	97	98	91	97	98
Mouth Slightly Open	93	93	<b>94</b>	94	87	78	92
Mustache	<b>97</b>	96	88	95	91	87	95
Narrow Eyes	<b>87</b>	87	83	81	82	73	81
No Beard	95	95	95	<b>96</b>	90	75	95
Oval Face	72	<b>73</b>	73	70	64	72	66
Pale Skin	<b>97</b>	97	93	94	83	84	91
Pointy Nose	75	73	75	74	68	<b>76</b>	72
Receding Hairline	<b>92</b>	92	88	90	76	84	89
Rosy Cheeks	<b>94</b>	94	87	91	84	73	90
Sideburns	95	95	95	<b>96</b>	94	76	96
Smiling	<b>92</b>	92	92	92	89	89	92
Straight Hair	<b>79</b>	79	78	79	63	73	73
Wavy Hair	71	73	<b>82</b>	81	73	75	80
Wearing Earrings	83	84	86	79	73	<b>92</b>	82
Wearing Hat	98	98	98	98	89	82	<b>99</b>
Wearing Lipstick	92	92	<b>93</b>	93	89	93	93
Wearing Necklace	<b>86</b>	86	71	71	68	86	71
Wearing Necktie	95	<b>96</b>	93	95	86	79	93
Young	87	87	87	<b>88</b>	80	82	87
Average	<u>89.4</u>	<b>89.5</b>	87.7	87.9	81.1	83.6	87.3

Table 2.11: The performance comparison of attribute detection methods.

## 2.4.2 Network architecture

The DCNNs have two different architectures. The architectures of the Deep Convolutional Neural Network for Active Authentication (Deep-CNNAA) and Wide-CNNAA can be found in Table 2.12. The four sets of models compared are: BinaryDeep-CNNAA, BinaryWide-CNNAA, MultiDeep-CNNAA, and MultiWide-CNNAA. First we describe the shared configuration that is used to train these networks and then the ones that are specific to each class of the networks.

*Wide-CNNAA		*Deep-CNNAA	
input	$w \times h \times 128$	input	$w \times h \times 3$
type	patch size	type	patch size
conv_relu	$7 \times 7 \times 128$	conv_relu	$7 \times 7 \times 32$
maxpool $3 \times 3/2$ (stride)			
conv_relu	$5 \times 5 \times 128$	conv_relu	$5 \times 5 \times 32$
		conv_relu	$5 \times 5 \times 32$
		conv_relu	$5 \times 5 \times 32$
maxpool $3 \times 3/2$			
conv_relu	$3 \times 3 \times 128$	conv_relu	$3 \times 3 \times 32$
		conv_relu	$3 \times 3 \times 32$
		conv_relu	$3 \times 3 \times 32$
		conv_relu	$3 \times 3 \times 32$
maxpool $3 \times 3/2$			
FC_relu	$dim \times 128$	FC_relu	$dim \times 64$
FC_relu	$128 \times 128$	FC_relu	$64 \times 32$
logits $Num\_Attr \times 2$			
Softmax loss			

Table 2.12: The architectures of our networks. The number of parameters depends on the face region that they operate on and can be found in Table 2.16.

**Shared configuration** All of these 100 networks, 20 Multi\*-CNNAA and 80











face part					
No. of Attributes	10	10	10	7	16
Size	$53 \times 39$	$115 \times 41$	$65 \times 38$	$40 \times 56$	$90 \times 62$
face part					
No. of Attributes	15	21	15	15	14
Size	$55 \times 82$	$115 \times 107$	$128 \times 52$	$128 \times 45$	$62 \times 100$

Table 2.13: The face regions that are extracted by cropping around the landmark points and their corresponding number of attributes. A Multi\*-CNNAA that operates on a face crop has “No. of Attributes” tasks.

Binary\*-CNNAA , are trained on the publicly available CelebA [Liu et al., 2015] dataset. It has 200 thousands images of 10 thousands identities, each with 40 attribute labels. It is divided into 160k training, 20k development, and 20k test images. The DCNNs are trained using the recently released Tensorflow [Abadi et al., 2015a] which also has a mobile implementation. All of the networks are initialized with random weights and are trained with the same policy. The Adam optimizer is used to train all of these networks since it incorporates the adaptive learning rate update step, and performs well without careful fine tuning of the learning parameters [Kingma and Ba, 2014]. Subsequent fine tuning can give better results. Early stopping [Prechelt, 1998] using the accuracy on the development set is used to select the final model for each network. The inputs are colored face part images that are randomly flipped and also their contrast and gamma are randomly changed to augment the data that we have to prevent over-fitting.

Due to the nature of the attributes, most of them have an unequal number of positive and negative labels. Extra care has been taken to make sure the networks

are not biased toward one class with the help of data augmentation and stochastic optimization.

**Binary\*-CNNAA** The binary networks are for a single task and are trained by the labels of one single attribute. The input face images are aligned to a canonical coordinate using the landmarks given by [Asthana et al., 2013]. To balance the training data, the class with the lower number of training data is distorted and added to the input queue so that the number of images for each class is equal. Then the data is shuffled and fed in batches to the training algorithm. The softmax cross entropy loss  $l_B$  is used to train these binary networks

$$l_B(w) = \frac{1}{N} \sum_{i=1}^N (1 - y_i) \log p(y_i = 0|w) + y_i \log p(y_i = 1|w) \quad (2.4)$$

where  $y_j \in \{0, 1\}$  is the attribute presence label,  $p(y = j|w) = \frac{\exp(f_j^w(x))}{\sum_{i=0}^1 \exp(f_i^w(x))}$  where  $f_i^w(x)$  is the logits of the  $i$ th output neuron of the network with weights  $w$ .

**Multi\*-CNNAA** The Multi\* networks are the proposed models that are as complex as the binary models but predict multiple attributes at once. The face parts and the number of attributes that are assigned to them can be found in Table 2.13. For each part, the corresponding network has an output layer that contains neurons for each attribute that is assigned to that face part. We use the softmax

cross entropy loss for part  $q$  as specified below:

$$\begin{aligned}
l^q(w) = & \frac{1}{N} \sum_{a=1}^{N^q} \sum_{i=1}^{n_a^q} (1 - y_i^a) \log p(y_i = 0|w) \\
& + y_i^a \log p(y_i = 1|w)
\end{aligned} \tag{2.5}$$

where  $N^q$  is the number of attributes assigned to part  $q$ .  $n_a^q$  is the number of images with the  $a$ th attribute of part  $q$  in the current batch.  $y_i^a \in \{0, 1\}$  is 1 if the  $i$ th image has the  $a$ th attribute and  $N$  is the batch size.  $p(y_i^a = 1|w)$  is the same softmax as Eq 2.4.

To deal with the class ratio imbalance of the attributes, we shuffle the training data in a way that the network sees the rare class for each attribute frequently. For example, for the attribute “Mustache”, the positive class is the rare one since most of the 202k images do not have this attribute. To handle this imbalance, a queue is created for each attribute and images that have the rare class are added to that queue. A queue is also created for images with all the attributes belonging to the major class. Then all of the queues are shuffled. We treat each queue as a circular buffer so that the training batches are created by sampling with replacement from one of these queues at random. Also, each time the images are distorted differently.

After training all the networks, most of the attributes are present in multiple networks. For each attribute, we only take the embeddings of relevant parts. For instance, for the attribute ”Mustache” in the MultiDeep-CNNAA , the 32 dimensional embedding of the parts, mouth, mouth and nose, and mouth and chin are taken and concatenated together. At first, 10000 examples, sampled from the train-

ing portion of CelebA, are selected for training and the development set of CelebA is used for fine tuning the linear SVMs hyperparameters. Then, following the protocol of [Liu et al., 2015], linear SVMs are trained with the selected parameters on the development set as their training set and tested on the test set.

### 2.4.3 Comparison of attribute detection methods



Figure 2.11: Sample images from subspace clustering of face part embedding in attribute space.

We compare our proposed networks with FaceTracer [Kumar et al., 2008], PANDA [Zhang et al., 2014b], and CelebA [Liu et al., 2015] attribute networks. These models capture a broad spectrum of possible automatic attribute detection models.

FaceTracer [Kumar et al., 2008] attribute classifiers are trained by extracting traditional low-level features like HOG and color histogram from aligned face parts by incrementally finding the best set of features and training the Support Vector

Machines (SVM’s) on the selected features and parts for attribute detection. The face crops are extracted from the ground truth landmarks.

PANDA ensembles multiple CNNs for the face parts and concatenates the outputs of the last layer and train SVMs for each attribute. There are two differences between our network architecture and PANDA networks. First, in PANDA, all of the attributes are associated with all of the parts. Second, in our Multi\*-CNNAA networks, the last layer is shared between all of the attributes softmax losses, but in PANDA there are two fully connected layers after the shared fully connected layer for each one. As a result, in our network, the different attributes that are associated with one network lie in the same Euclidean space of the last fully connected layer of the network.

CelebA takes a different approach by pre-training their network with face identities of CelebFaces [Sun et al., 2014b] for both face verification and identification. Then they extract features from multiple overlapping crops of the face and train SVMs for each crop for each attribute. To predict the attribute they average over the scores of SVMs. They use a localization network to detect the face region and pass them onto the classifier networks.

We also follow [Levi and Hassner, 2015] and train a single task network for each attribute in Binary\*-CNNAA on the full face. Table 2.11 shows the accuracy of each of these methods.

As it can be seen, our Multi\*-CNNAA networks give equal or better results than the rest. The MultiWide-CNNAA performs slightly better than the MultiDeep-CNNAA in attribute prediction. This may be due to the large number of parameters

that they have as shown in Table 2.16. However they are slower and consume more energy.

#### 2.4.4 Attribute discovery

As mentioned in the previous section, our Multi\*-CNNAA networks transform the input face regions to a shared Euclidean space for the attributes associated with that part. To further explore this Euclidean space, we perform Sparse Subspace Clustering (SSC) [Elhamifar and Vidal, 2009] on 10000 points that are selected from the training portion of CelebA dataset. The intuition behind this clustering is that the face parts that have the same attribute lie in the same subspace. SSC uses the fact that each data point can be represented by a sparse linear combination of the other points in the same subspace. Therefore it formulates the clustering problem as

$$\underset{C \in \mathbb{R}^{n \times n}}{\text{minimize}} \quad |C|_1 + \|D - DC\|_F^2 \quad (2.6)$$

$$\text{subject to} \quad \text{diag}(C) = 0 \quad (2.7)$$

where  $D \in \mathbb{R}^{d \times n}$  is the data matrix containing  $n$  points of dimension  $d$  and  $C \in \mathbb{R}^{n \times n}$  is the affinity matrix. To enforce the constraint, for each datapoint they take it out of  $D$  and then perform sparse coding. To get the clusters they perform spectral clustering on  $C$ . We find 10 clusters per face regions. The clusters corresponding to the “Hair-Forehead” region of the face and the “eyes” region can be seen in Figure 2.11. As illustrated, the “discovered” attributes overlap with the labels that we had



in the training time mostly, but also some attributes are divided into finer categories. For example, in the mouth and chin category (b) contains images of people with chin shape similar to African-Americans which was not present in the labels. In the “Hair-Forehead” region cluster (c) contains male images with short hair which again was not seen in the labels. Also, the gender of the people in the same cluster is the same for these two parts. As shown in the next section, these attributes give good result for authentication.

#### 2.4.5 Experiments

We evaluate the performance of CNNAA for the task of active authentication using two publicly available datasets MOBIO [McCool et al., 2012] and AA [Fathy et al., 2015]. These datasets contain videos of the users interacting with cell phones. We compare the authentication performance of our DCNN attribute detectors and discovered attributes with baseline Local Binary Patterns [Ahonen et al., 2006] and ACAA [Samangouei et al., 2015] which is the only attribute-based approach for this task. The extracted attribute features of ACAA [Samangouei et al., 2015] from the videos of these two datasets are provided by the authors. We follow the same protocol as ACAA to extract facial parts and video features. So, we average over the extracted attribute outputs for the video frames to get the video descriptors.

We cast the problem of continuous authentication as a face verification problem in which a pair of videos is given and we determine whether they contain the same identity or not. To compare the performance of the algorithms, the receiver

operating characteristic (ROC) curve is used. Many other measures of performance can be readily extracted from the ROC curve. ROC curve plots the relationship between false acceptance rates (FARs) and true acceptance rates (TARs). The ROC curve can be computed from a similarity matrix  $S$  between gallery and probe videos. We also report the EER value where TAR and FAR are equal. EER value gives a good idea of the ROC curve shape since it can be extracted by plotting the diagonal line on the curve and see how soon it hits it. Thus, the better the algorithm, the lower is its EER value.

We give each video frame to the CNNAA networks and predict the attributes with linear SVMs. For the learned attributes, we put the probabilistic output of the SVMs which are trained by LIBSVM [Chang and Lin, 2011] as our final attribute feature. Since the attribute outputs of our models are probability values we get the similarity value  $s_{i,j} = \langle e_i, t_j \rangle$ , where  $e_i$  is the feature vector for the enrollment video and  $t_i$  is the test video features.

To use the discovered attributes (DiscAttrs) for authentication, we extract the attribute features by a similar approach to Sparse Representation Classification [Wright et al., 2009]. Each face crop from the video frame is embedded to the attribute space of MultiDeep-CNNAA . It is represented by the dictionary which we used in Section 2.4.4, so that we know the cluster assignment of its atoms. We normalize all of the dictionary atoms and the embedding and then get each feature value by a softmax over the representation contribution of each cluster in



Figure 2.12: Sample images of the three sessions of the AA01 dataset.

the attribute space. To do so, we first solve

$$\underset{f \in \mathbb{R}^n}{\text{minimize}} \quad \|f\|_1 + \|f - Df\|_F^2 \quad (2.8)$$

to get the sparse representation  $f$  of the face crop of that video frame. Then we set the  $i$ th feature for that face crop to  $p(c = i|D)$  which is calculated by

$$p(c = i|D) = \frac{\exp(\|D_{:,i}f_i\|)}{\sum_{k=1}^{10} \exp(\|D_{:,k}f_k\|)} \quad (2.9)$$

where  $D_{:,i}$  is the dictionary atoms of cluster  $i$  and  $f_i$  are the coefficients corresponding to those atoms. Thus, if  $f$  is in the subspace spanned by the points in  $D$  that are in

cluster  $i$ , it will have more energy in non-zero values for those atoms. To solve (2.8) we use the Orthogonal Matching Pursuit [Tropp and Gilbert, 2007] algorithm with sparsity 20. We choose 20 because the subspaces of DeepMulti-CNNAA embedding must have dimension less than the embedding space dimension which is 32 for Deep\*-CNNAA. Then we concatenate

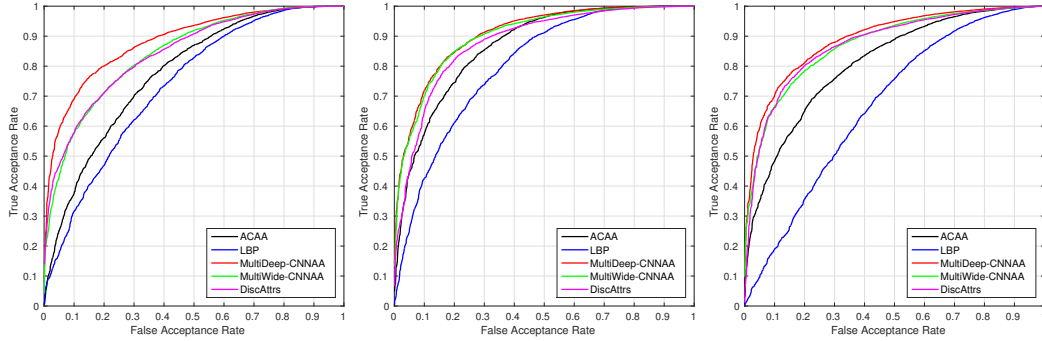


Figure 2.13: ROC curve of different experiments on AA01 [Fathy et al., 2015] and MOBIO [McCool et al., 2012] dataset. (a) is the ROC curve of AA01 with all of the sessions together in gallery and probe. (b) is the ROC curve of MOBIO with all of the mobile sessions together with the last session videos as gallery and the rest of the session as probe. (c) is the ROC curve of the cross-device experiment.

	ACAA	LBP	MD-CNNAA	MW-CNNAA	DiscAttrs
$e \rightarrow t$					
$1 \rightarrow 1$	0.14	<u>0.13</u>	<b>0.11</b>	0.14	0.16
$2 \rightarrow 2$	0.19	0.31	<b>0.18</b>	0.22	<u>0.17</u>
$3 \rightarrow 3$	0.16	0.20	<b>0.10</b>	<b>0.10</b>	0.13
$1 \rightarrow 2, 3$	0.38	0.38	<b>0.18</b>	0.25	<u>0.23</u>
$2 \rightarrow 1, 3$	0.31	0.33	<b>0.26</b>	0.30	<u>0.31</u>
$3 \rightarrow 1, 2$	0.31	0.38	<b>0.19</b>	<u>0.24</u>	0.25
Altogether	0.30	0.34	<b>0.20</b>	<u>0.25</u>	<u>0.25</u>

Table 2.14: The EER values for the different experiments on AA01 [Fathy et al., 2015] dataset. The sessions numbers are: 1. Office light 2. Low light 3. Natural light. DiscAttrs column contains the EER values using the discovered attributes.

## Results

To plot the ROC curves and evaluate our method, in each dataset, for each person one session's videos are considered as the enrollment videos and the other videos as test videos. The similarity matrix is then generated by pairwise distance between the enrollment and the test videos. The corresponding ROC curve is plotted for each experiment.

**AA01** AA01 is a mobile dataset with 750 videos of 50 subjects. Each subject has three sets of videos with three different lighting conditions. Each user is asked to perform a set of actions on the phone while the front camera is recording the video. The videos are captured by an iPhone 4 camera. The three lighting conditions are: office light, low light, and natural light. The sample images of this dataset in Figure 2.12 show the three different illuminations in each session. Figure 2.12 also presents some partial faces in the dataset. Each person has five videos of performing five different tasks on the phone. There is a designated enrollment video for each person. Three different experiments have been conducted on this dataset.

First, the enrollment and the test videos for all of the 50 subjects are taken from the session with the same lighting condition. The EER values of this experiment can be found in the first three rows of Table 2.14. It can be seen that our MultiDeep-CNNAA has the lowest EER in all cases. This experiment reveals the discriminative power of the features when the surrounding environment is the same. It can be seen that in this case, high dimensional LBP features even beats ACAA in the office light session.

In the second one, the enrollment video is taken from one illumination session and the test videos from another. The EER values corresponding to this experiment are depicted in the next three rows of Table 2.14. The performance drop in our method is 0.08 on average while ACAA suffers 0.17 and LBP 0.15. The reason is that ACAA attribute classifiers use low level features that are sensitive to illumination changes, but CNNAA is trained on a large-scale unconstrained dataset containing a lot of variations and thus gives more robust features.

In the last experiment, all enrollment videos of the three sessions are put in the gallery and all the test videos in the probe of to get the similarity matrix. The ROC curve corresponding to the third general experiment is plotted in Figure 2.4.5. It can be seen that MultiDeep-CNNAA performs the best and MultiWide-CNNAA and the discovered attributes are tied as second best.

One explanation for lower performance of MultiWide-CNNAA compared to MultiDeep-CNNAA is that it has many more parameters than MultiDeep-CNNAA according to Table 2.16 and has overfitted to the celebrity faces distribution.

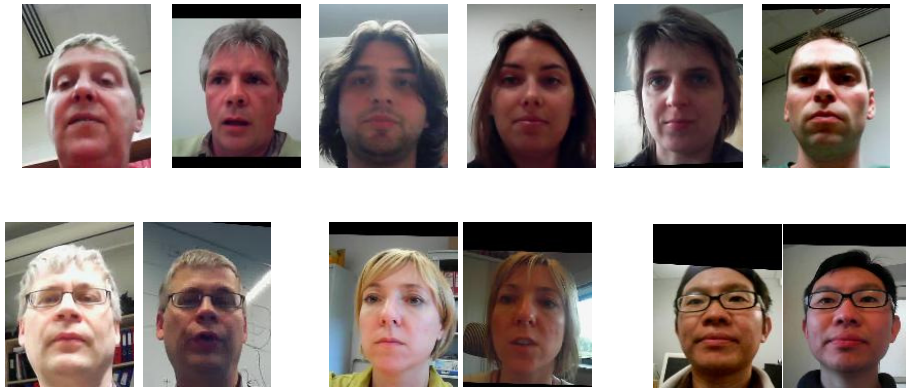


Figure 2.14: Sample images of the three sessions of the MOBIO dataset. First row images are from different sites, second row is the pairs with the same identities in two different sessions.

**MOBIO** MOBIO [McCool et al., 2012] is a more challenging dataset of 152 subjects. The videos are taken in six different universities across Europe. For most of the subjects 12 sessions of video are captured. All of the mobile videos are captured with a Nokia N93i. The first session’s videos are also recorded with a 2008 MacBook laptop. We perform two experiments on this dataset. We take the 12th session videos as our training videos since they are the mostly available videos accross the dataset. A few subjects have less than 12 session videos.

In the first experiment, we just consider videos that are taken by the mobile device. We show the EER values for the mobile videos of the subjects within each site as well as all of the videos together in Table 2.15. This experiment is similar to the third experiment of AA01 dataset since the environment conditions for enrollment videos and test videos can be the same or different. However, there are three times more subjects in MOBIO and more variations in illumination condition of the videos. The ROC curve for this experiment is plotted in Figure 2.4.5.

The second experiment is about the cross sensor authentication, in which you enroll yourself on one device and test on another device. To see how important sensor change can be for low level features, one can look at the performance drop of LBP feature in this experiment and the previous one in Table 2.15. The decrease is 0.10 for the LBP feature and then 0.05 for ACAA which depends on low level features, while CNNAA methods just have a decrease of 0.01 in EER value. The ROC curve for this experiment is presented in Figure 2.4.5. Again, this is due to the fact that CNNAA has seen more variations in the large training set. Our method can also handle partial face verification if a partial face detector like [Mahbub et al.,

	ACAA	LBP	MD-CNNAA	MW-CNNAA	DiscAttrs
but	0.26	0.36	<b>0.19</b>	<u>0.20</u>	0.23
idiap	0.25	0.35	0.27	0.25	<b>0.24</b>
lia	0.24	0.34	0.17	<b>0.15</b>	<u>0.16</u>
uman	0.27	0.33	<b>0.18</b>	<u>0.20</u>	0.21
unis	0.2	0.27	<b>0.07</b>	<u>0.1</u>	<u>0.1</u>
uoulu	<u>0.18</u>	0.23	<b>0.14</b>	<b>0.14</b>	0.19
Altogether	0.22	0.28	<b>0.17</b>	<u>0.18</u>	0.19
Mobile-PC	0.27	0.38	<b>0.19</b>	0.21	<u>0.2</u>

Table 2.15: The EER values corresponding to MOBIO dataset experiments.

2016] is available.

## Mobile Efficiency

There is a trade-off among power consumption, authentication speed, and accuracy of the model for the task of active authentication on mobile devices. The response time is important since we do not want to freeze other running processes and create an unpleasant user experience while authenticating. Power consumption is also important because as frequent demands for charging the battery can be annoying.

To show the effectiveness of our approach, we measure the attribute prediction speed of our networks and the battery consumption on an LG Nexus 5 device. The results are shown in Table 2.16. This mobile device has a quad-core QUALCOMM



Input size	Network	Parameters	Prediction time	Network	Parameters	Prediction time
$128 \times 52$	D-UpperHead	275,360	0.15s	W-UpperHead	1,825,664	0.26s
$115 \times 41$	D-BothEyes	227,936	0.11s	W-BothEyes	1,447,552	0.19s
$90 \times 62$	D-EyesNose	244,704	0.13s	W-EyesNose	1,580,160	0.22s
$40 \times 56$	D-Nose	170,400	0.06s	W-Nose	988,032	0.1s
$55 \times 82$	D-NoseMouth	232,352	0.10s	W-NoseMouth	1,481,600	0.18s
$65 \times 38$	D-Mouth	164,448	0.06s	W-Mouth	939,648	0.11s
$115 \times 107$	D-EyesNoseMouth	441,632	0.28s	W-EyesNoseMouth	3,154,304	0.48s
$128 \times 45$	D-MouthChin	244,640	0.13s	W-MouthChin	1,579,904	0.23s
$62 \times 100$	D-Ear	256,864	0.14s	W-Ear	1,677,952	0.25s
$53 \times 39$	D-Eye	162,400	0.06s	W-Eye	923,264	0.08s
Overall	MultiDeep-CNNAA	2.4M	1.22s	MultiWide-CNNAA	15.6M	2.10s
$128 \times 128$	BinaryDeep-Full	584,160	0.36s	BinaryWide-Full	4,289,664	0.637s

Table 2.16: Network size and prediction speed of the networks. The D-\* means it has MultiDeep-CNNAA architecture and W-\* means it is MultiWide-CNNAA . The Binary\*-CNNAA network prediction times are just for one attribute. For all of them together it will be 40 times this value.

Snapdragon 800 clocked at 2.26 GHz and 2 GB of RAM. This specification is considered average compared to the current smartphones. We use the Tensorflow [Abadi et al., 2015a] implementation of CNNs on Android devices.

We take one shot with the smartphone camera and feed it to the network for 200 times and measure the prediction speed by looking at the average duration per frame. To measure the power usage we use PowerTutor [Zhang et al., 2010] which registers the energy usage per running application and also in total. We do not use the camera continuously because it will bias the response time and power usage of the network. We take the image and the application works in background. The default Android processes are the only other processes that are running besides the application that runs the networks and PowerTutor application.

According to Tabel 2.16 all of the attributes are detected in 1.22s with MultiDeep-CNNAA running on CPU in the background without blocking other applications.

MultiWide-CNNAA takes 2.10s. The BinaryDeep-CNNAA takes 14.4s and BinaryWide-CNNAA 25.5s.

The MultiDeep-CNNAA architecture consumes 780mW power on average and MultiWide-CNNAA drains 1100mW of the battery power. The average battery usage of Android when it is not running the CNNAA networks is 600mW according to PowerTutor. To see how this affects the battery life, suppose the battery capacity is  $C$  Watt-hours (Wh). Then

$$d = \frac{C}{P_n + \beta\alpha P_d} \quad (2.10)$$

where  $d$  is the mobile device's battery life,  $P_n$  is the power consumption in normal use,  $P_d$  is the power usage of the attribute detection algorithm,  $\beta$  is the fraction of time that the mobile device is being used,  $\alpha$  is the authentication ratio constant.  $\alpha$  shows how often we want to authenticate the user considering the prediction time of the algorithm, i.e., we authenticate every  $\frac{T_a}{\alpha}$  where  $T_a$  is the prediction speed of the model. For instance, if  $\alpha = 0.5$  we authenticate every 2.44s using MultiDeep-CNNAA and every 4.2s using MultiWide-CNNAA .

To make the feasibility of CNNAA clearer, suppose we authenticate the user using the MultiDeep-CNNAA architecture on the Nexus 5 device. We choose the MultiDeep-CNNAA since it performs well in the authentication task and also it has a better runtime and power usage. The Nexus 5 has a 2300mAh battery with 3.8V voltage, so  $C = 8.74Wh$ .  $P_n = 0.6W$  for the "normal usage" state which is when just Android 5 and the default applications are running. This gives 14.5

hours battery life. Now if  $\alpha = 1$  which means we want to authenticate with the highest speed possible and if we are using the phone all the time with  $\beta = 1$  then the battery life will be reduced to 6.3 hours in the worst case. In a realistic setting with  $\beta = 0.2$  and  $\alpha = 0.5$  it becomes 12.85 hours which is reasonable. Also, if a GPU implementation of CNNs on Android [[Sarkar et al., 2016](#)] is used, attribute prediction can happen much faster with less energy consumption.

## Chapter 3: Conditional Image Generation From Attribute Vectors

### 3.1 Introduction

High fidelity conditional image generation remains an elusive but highly sought after goal of computer vision. Any such well-trained algorithm would allow users to create an unlimited quantity of digital content for use in consumer photography, digital photo editing, and dataset generation. While this ultimate goal remains out of reach of state-of-the-art algorithms, an approximation of this task has been phrased as attribute-level image manipulation. The ability to manipulate and edit images based on pre-defined attributes has various real-world applications. Consumer photo editing software would ideally allow users to edit their images in specific ways, such as re-touching hair-color or removing a hat from someone’s head. Additionally, the ability to adequately model visual attributes may improve supervised learning generalization via data augmentation.

Two recent works [Lample et al., 2017, Perarnau et al., 2016] have focused on the problem of attribute-based image manipulation. In each, they train a conditional generator on discretely represented binary attributes and a nuisance vector which together completely encode an input image. This mechanism allows users to manipulate, or *Swap*, a binary attribute of the input image and obtain a re-rendered

output image. For example, if one of the attributes is “has hat”, they can optionally add a hat to a person’s head or conversely, remove one.

These efforts are indeed impressive. However, attribute Swaps performed on discrete binary representations are limited. They do not allow users to sample from the space of possible realizations of a binary attribute. For example, if I want to activate the “has hat” attribute, I might want to sample from various hats. Furthermore, a user might observe a realization of a particular attribute in a reference image and apply it to a query image. For example, one might want to activate the “has hat” attribute with a particular Fedora from a reference image.

To address these shortcomings, we introduce *CRISPR* an algorithm for image attribute manipulation that supports two new types of attribute manipulation operations: Diverse Swaps and Borrows. A Swap is a change to a visual attribute such that the input and output to our model have different visual attribute values, such as hat versus no hat. A Diverse Swap is a change to a visual attribute that is random: each time the operation is performed represents a sampling from the visual space that both ensures that the input and output images have different visual attributes but also that subsequent samples have variety. For example, if the input attribute is ‘no hat’, then all output images should be wearing hats. However, each sample from our model should produce different hats. Alternatively, a Borrow is an operation where we can change a visual attribute in such a way that it ‘borrows’ the visual component from a reference image and substitutes it into a query image. In particular, say the input has no hat and we wish to flip the ‘hat’ attribute to ‘has hat’ in such a way that the particular hat should match a top hat, then we should

be able to 'borrow' the 'top hat' representation from an example image and apply it to the query image.

To summarize, our method exhibits the following unique characteristics:

- Models visual attributes more flexibly through a continuous, rather than discrete, representation.
- Allows a user to alter an image by sampling from a learned distribution of image attributes.
- Allows a user to alter an image by 'borrowing' an attribute exemplar from another image.

## 3.2 Related Work

Our work is related to research in two problems in generative models of images; attribute-based image manipulation and learning disentangled representation.

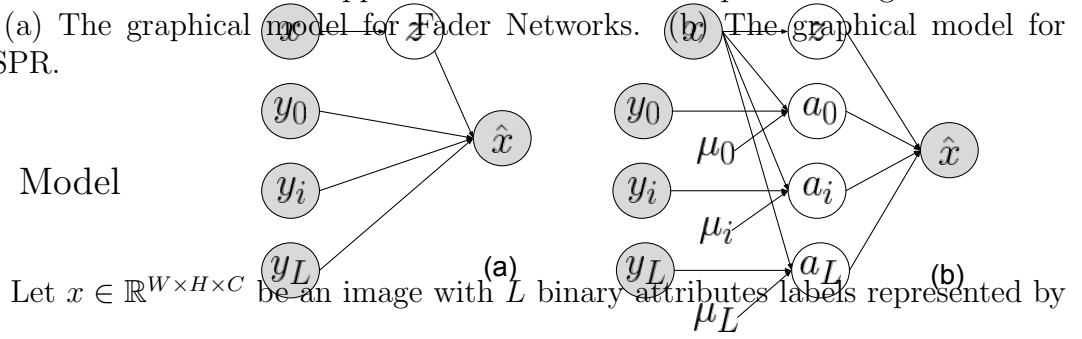
[[Yan et al., 2016](#)] was the first to use a variational auto-encoder (VAE) to build a conditional generative model where the image manipulation is performed by inferring the latent state given the correct attributes and then changing the attributes. [[Perarnau et al., 2016](#)] employed a similar encoder-decoder architecture, but devised a adversarial training regime [[Goodfellow et al., 2014a](#)] in order to improve the realism of manipulated images. [[Antipov et al., 2017](#)] utilized the same idea to alter the facial appearance as a function of age. However, in both cases, the training schemes lack a mechanism to ensure the decoder to utilize the annotator labels information, which limits the quality of image manipulation. Fader Networks

[Lample et al., 2017] aimed to alleviate this problem by imposing an additional constraint on the latent code to be invariant under any attribute specific information of input images, and showed a great improvement. However, all these approaches feed the attribute labels as input to the decoder, which incurs two limitations; (1) detailed information about the manifestation of each attribute label is lost e.g. we may know if the person wears glasses, but not the kind of the glasses; (2) there is no natural means to generate diverse examples of altered attributes. Our method in contrast learns multi-dimensional, continuous representation of attribute-specific information with a mechanism to sample from these spaces.

Another strand of related research is that of learning disentangled representation. The closest to our work is Inverse Graphics Network proposed in [Kulkarni et al., 2015] in which attribute-specific latent codes are learned within an auto-encoder network that can be used to alter image appearance. However, the model does not learn the attribute-invariant latent code, and thus limits the range of viable image manipulation operations e.g. attribute transfer from one image to another is not possible with this approach. By contrast, CRISPR models both attribute-invariant and attribute-specific representation. Many other work in this space such as predictability minimization framework [Schmidhuber, 1992], InfoGAN [Chen et al., 2016], conditional GAN based approach in [Mathieu et al., 2016] and neural photo editor [Brock et al., 2016], employed fully unsupervised methods, without specification of attribute types. These methods aim to automatically extract factors of variations, which may not be necessarily aligned with the specific demands of users in image editing applications.

Figure 3.1: Visualization of approaches to attribute manipulation as graphical models. (a) The graphical model for Fader Networks. (b) The graphical model for CRISPR.

### 3.3 Model



Let  $x \in \mathbb{R}^{W \times H \times C}$  be an image with  $L$  binary attributes labels represented by  $\mathbf{y} = [y_1, \dots, y_L] \in \{0, 1\}^L$ . For example, visual attribute labels assigned to images of faces can include *man/woman*, *glasses/no glasses*, *beard/no beard*. We define our model as an encoder-generator architecture, endowed with disentangled latent structures between attribute invariant and attribute specific components. More precisely, the encoder network  $E$  is a convolutional network that maps a given image  $x$  to a set of multi-dimensional, continuous vectors  $\{z, a_1, \dots, a_L\}$ , where  $z \in \mathbb{R}^p$  is a representation of  $x$ , invariant under any of the visual attributes  $[y_1, \dots, y_L]$  while each  $a_i \in \mathbb{R}^q$  encodes information in  $x$  specific to the  $i^{\text{th}}$  attribute. On the other hand, the generator  $G$  is a deconvolutional network that maps latent codes to an image  $\hat{x}$ . We refer to  $E_{inv}(x) := z$  and  $E_{attr}(x) := [a_1, \dots, a_L]$  as *attribute invariant* and *attribute specific* representations, respectively. We discuss our proposed training scheme to impose such structures on the latent space in Sec.3.4.

While we represent each attribute in a high-dimensional continuous space, we need to make sure that each attribute vector can be mapped to a discrete binary label  $y_i$ . Let  $H_i(a_i) \rightarrow y_i$  represent an attribute-specific classifier that maps each continuous  $a_i$  to a binary variable. Let  $\mu_i^0$  and  $\mu_i^1$  represent cluster centroids of attribute  $i$  for values 0 and 1, respectively. The graphical model of CRISPR can be



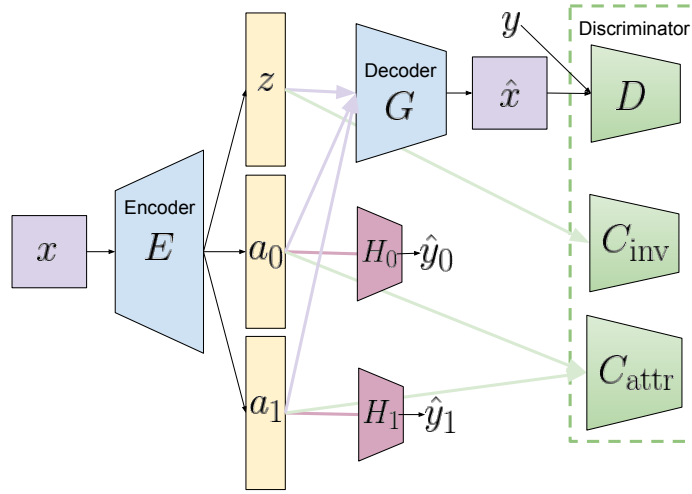


Figure 3.2: Architectural overview of the CRISPR architecture, best viewed in color. The image  $x$  is encoded by  $E$  producing invariance vector  $z$  and attribute vectors  $a_0$  and  $a_1$ . These are concatenated and passed to decoder  $G$  which produces reconstruction  $\hat{x}$ . Each continuous attribute vector is decoded using a linear classifier  $H_i$ . Auxiliary components of the architecture are framed by the dashed green outline. Discriminator  $D$  encourages the reconstructions to both appear natural and exhibit the expected attributes, classifier  $C_{\text{inv}}$  encourages  $z$  to encode only non-attribute information and classifier  $C_{\text{attr}}$  encourages each attribute vector to encode only its attribute information.

seen in Figure 3.1.

### 3.3.1 Inference

Reconstruction with our model is straightforward:  $\hat{x} = G(E(x))$ . However, it is our continuous representation of attributes that permits two unique modes of inference, Diverse Swaps and Borrows.

By using a discrete representation of attributes, regular attribute swaps are easy in that one need only artificially alter a discrete binary embedding. However, one cannot sample from the attribute space. Our model allows for **Diverse Swaps**: stochastic invocations of the swap operation that produce different visual manifestations. To formalize this operation, consider the oracle classifier  $C_{\text{oracle}}(x, i) \rightarrow y_i$  which is able to classify the attributes of an image  $x$ . Then a swap on attribute  $i$  is an operation  $S(x, i) \rightarrow \hat{x}$  such that  $C_{\text{oracle}}(x, i) \neq C_{\text{oracle}}(\hat{x}, i)$ . A Diverse Swap adheres

to the same constraint but because it is a stochastic transformation, subsequent invocations of  $S$  produce  $\hat{x}$  and  $\hat{x}'$  such that  $\hat{x} \neq \hat{x}'$ .

In order to perform a Diverse Swap, one first embeds image  $x$  as  $\{a_0, \dots, a_i \dots z\}$ . Next, one needs only sampling from a Gaussian distribution with mean  $\mu_i^0$  or  $\mu_i^1$  and scaled standard deviation to produce a new sampled attribute vector  $a'_i$ . Finally, the resulting image is decoded via  $\hat{x} = G(a_0, \dots, a'_i, \dots, z)$ . This partitioned representation not only allows us to swap visual attributes, but also lets us produce a diverse set of possible swaps.

A **Borrow** represents a transformation on attribute  $i$  such that attribute  $i$  is *borrowed* from exemplar image  $x_{\text{exemplar}}$  to image  $x$ . To perform a borrow,  $x$  is encoded as  $\{a_0, \dots, a_i \dots z\}$ ,  $x_{\text{exemplar}}$  is encoded as  $\{a_0^{\text{ex}}, \dots, a_i^{\text{ex}} \dots z^{\text{ex}}\}$ , and an attribute from  $x_{\text{exemplar}}$  is borrowed to produce encoding  $\{a_0, \dots, a_i^{\text{ex}} \dots z\}$ . Finally, the resulting image is decoded via  $\hat{x} = D(a_0, \dots, a_i^{\text{ex}}, \dots, z)$ . The overview of our method is shown in Figure 3.2.

### 3.4 Training

An ideally trained CRISPR model should exhibit several characteristics. First, the image reconstructions should be high fidelity. Secondly, the encoder should produce a partitioned representation with attribute-invariant and attribute-specific components. Third, the latent attribute space must be amenable to sampling in order to produce diverse and plausible samples. To this end, we introduce a set of auxiliary components and losses to the model.

### 3.4.1 Reconstruction

To ensure that our reconstructions are high fidelity, we use the thresholded mean squared error  $\mathcal{L}_{\text{recon}}$ :

$$\min_{G,E} \mathbb{E}_x \left[ \max(N^{-1} \|G(E(x))_i - \hat{x}_i\|^2, \kappa) \right] \quad (3.1)$$

where  $N = H \times W \times C$  is the number of pixels and  $\kappa$  is a threshold used to avoid the reconstruction loss dominating optimization. Additionally, we use an adversarial loss to capture fine textures [Isola et al., 2017]. Formally, let discriminator  $D : \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$  be a binary classifier trained to discriminate if a given image-label pair  $(x, y)$  is real or fake. The following loss function is additionally optimized during training  $\mathcal{L}_{\text{cGAN}}$ :

$$\min_{G,E} \max_D \mathbb{E}_{x,y} \left[ \log(D(x, y)) + \log(1 - D(G(E(x)), y)) \right] \quad (3.2)$$

### 3.4.2 Learning attribute-invariant representation, $z$

Encoder  $E$  should learn to produce latent codes  $z$  that are independent attributes  $[y_0, \dots, y_L]$ . In particular, following [Lample et al., 2017, Mathieu et al., 2016] we employ adversarial training on  $z$  where additional classifier  $C_{\text{inv}}$  is trained to identify the true attributes  $y$  of an input image  $x$  from its latent code  $z = E(x)$ . This scheme aims to obtain invariance in  $z$  by training encoder  $E$  such that  $C_{\text{inv}}$  is unable to identify the attributes from any input images.

The classifier  $C_{\text{inv}}$  takes  $z = E_{\text{inv}}(x)$  and generates a vector of probabilities

$[C_{\text{inv}}^{(1)}(z), \dots, C_{\text{inv}}^{(L)}(z)] \in [0, 1]^L$  of respective attributes being present in the image i.e. we estimate  $P(y|z) \approx \prod_{i=1}^L C_{\text{inv}}^{(i)}(z)^{y_i} \cdot (1 - C_{\text{inv}}^{(i)}(z))^{1-y_i}$ . The weights of  $C_{\text{inv}}$  are trained in an adversarial fashion: In one stage, we train the encoder to produce  $z$  vectors that maximize the probability of misclassification by  $C_{\text{inv}}$ . In another step we train the classifier  $C_{\text{inv}}$  to predict all of the attributes correctly from  $z$ . More formally we optimize the invariance loss  $\mathcal{L}_{\text{inv}}$ :

$$\min_{C_{\text{inv}}} \max_{E_{\text{inv}}} \mathbb{E}_{x,y} \left[ - \sum_{i=1}^L y_i \cdot \log C_{\text{inv}}^{(i)}(E_{\text{inv}}(x)) + (1 - y_i) \cdot \log(1 - C_{\text{inv}}^{(i)}(E_{\text{inv}}(x))) \right] \quad (3.3)$$

### 3.4.3 Decoupling attribute specific representations $\{a_i\}$

In addition to latent code  $z$  not representing any attribute specific information, we also want each attribute code  $a_i$  to encode only information about that particular attribute. To this end, we use an additional classifier  $C_{\text{attr}}$  to identify the true attributes  $y$  based on one of latent codes  $\{a_1, \dots, a_L\}$ . On the other hand, for each attribute  $i$ , the encoder  $E$  is trained to generate a code  $a_i = E_{\text{attr}}^{(i)}(x)$  such that the classifier  $C_{\text{attr}}$  can correctly infer about the attribute  $y_i$  while being maximally confused about the other attribute identity i.e.  $C_{\text{attr}}^{(i)}(a_i) = y_i$  and  $C_{\text{attr}}^{(j)}(a_i) = 0.5$  for  $j \neq i$  for any input images. More concretely, we propose the following encoder loss  $\mathcal{L}_{\text{attr}}$  to be maximized:

$$\min_{C_{\text{attr}}} \max_{E_{\text{attr}}} \mathbb{E}_{x,y} \left[ \sum_{i,j=1}^L (-1)^{\mathbb{1}_{i=j}} \cdot [y_j \cdot \log C_{\text{attr}}^{(j)}(E_{\text{attr}}^{(i)}(x)) + (1 - y_j) \cdot \log(1 - C_{\text{attr}}^{(j)}(E_{\text{attr}}^{(i)}(x)))] \right] \quad (3.4)$$

In the maximization step, the encoder tries to fool the classifier with  $i$ th attribute representation on all other labels but  $y_i$ . In the discriminator step, the classifier tries adjust its weights to learn to predict all the attributes from every single  $a_i$ .

### 3.4.4 Sampling Attributes

Up to now, by just training with the aforementioned losses, we can already map images to attribute specific representations  $\{a_i\}_{i=1}^L$ , which allows us to perform the “Borrow” operation mentioned in section 3.3 by transplanting these codes from one image to another. However, there is no mechanism to map each of these attribute specific codes to a discrete binary label  $y_i$ , which prevents us from performing attribute swap let alone generating multiple plausible samples of it (“DiverseSwap”). To address this limitation, we further regularize the latent space with an addition of center loss [Wen et al., 2016], which not only improves the quality of embedding by encouraging images with similar attributes to cluster together, but also provide “switches” in the latent space for manipulating attributes.

We now formalize our proposition. Suppose we want to alter the first attribute of image  $x$  from  $y_1 = 1$  to  $y_1 = 0$ . Given the latent representation  $E(x) = [z, a_1, \dots, a_L]$ , we can define this operation of attribute swap as finding the alternative attribute code such that the probability of the reconstructed image does not have the first attribute is maximized i.e.  $a_1^{new} = \operatorname{argmax}_{a_1} P(y_1 = 0 | z, a_1, \dots, a_L)$ , and subsequently reconstruct on the altered latent codes  $x_{\text{new}} = G(z, a_1^{new}, \dots, a_L)$ . If we assume that given the code  $a_1$ , the attribute label  $y_1$  is conditionally inde-

pendent of the invariant code  $z$  and other attribute specific codes  $a_2, \dots, a_L$ , and the marginal distributions  $P(y_1)$  and  $p(a_1)$  are constant, we see that the attribute swap reduces to finding  $a_1^{new} = \operatorname{argmax}_{a_1} P(a_1|y_1 = 0)$ . Our center loss based scheme estimates  $P(a_1|y_1 = 0)$  with a Gaussian distribution  $\mathcal{N}(a_1; \mu_1^0, \sigma_1^0)$ , and thus alters the attribute by setting  $a_1^{new} = \mu_1^0$ . To perform ‘‘DiverseSwap’’ in this example, we generate a set of plausible attribute codes instead by sampling from this distribution  $\{a_1^{new,1}, a_1^{new,2}, \dots\} \sim \mathcal{N}(a_1; \mu_1^0, \sigma_1^0)$ . More generally, this scheme models the latent of every attribute state as a Gaussian distribution i.e.  $P(a_i|y_i) \approx \mathcal{N}(a_i; \mu_i^{y_i}, \sigma_i^{y_i})$  for  $i = 1, \dots, L, y_i \in \{0, 1\}$ .

We now describe how to learn means (centroids) and standard deviations  $\{\mu_i^0, \mu_i^1, \sigma_i^0, \sigma_i^1\}_{i=1}^L$  of the attribute-specific latent codes. We learn the two in an alternating fashion. First, given a set of  $\{\mu_i^0, \mu_i^1\}$  initialized to zero vectors, we estimate the standard deviations as follows. We train attribute specific linear classifiers  $Cl_{spec}(a_i) = \mathbf{w}^T a_i + b_i$  in the space of attribute representations by minimizing the standard cross-entropy loss  $\mathcal{L}_{spec}^i$ , and estimate  $\sigma_i^0, \sigma_i^1$  by the distance of the centroid from the decision boundary of  $Cl_{spec}$  where  $c \in \{0, 1\}$ :

$$\sigma_i^c = \frac{|\mathbf{w}_i^T \mu_i^c + b_i|}{\|\mathbf{w}_i\|} \quad (3.5)$$

We then estimate the centroids  $\{\mu_i^0, \mu_i^1\}$  by minimizing  $\mathcal{L}_{center}$ :

$$\min_{E, \mu_i^0, \mu_i^1} \mathbb{E}_{x,y} \left[ \sum_{i=1}^L \frac{1}{\sigma_i^{y_i}} \|E_{attr}^{(i)}(x) - \mu_i^{y_i}\|_2^2 \right] \quad (3.6)$$

where  $E_{attr}^{(i)}(x)$  is the attribute code  $a_i$  for image  $x$ . The centers are updated in every step of stochastic gradient descent with a batch-wise maximum likelihood update step with momentum as in [Wen et al., 2016]:

$$\nabla \mu_i^c \{t\} = \frac{1}{|Ind(y_i = c)|} \sum_{k \in Ind(y_i = c)} \mu_i^c - E(x_k) \quad (3.7)$$

$$\mu_i^c \{t+1\} = \mu_i^c \{t\} + \alpha \nabla \mu_i^c \{t\} \quad (3.8)$$

where  $\mu_c^i$  is the centroid of  $i$ th attribute for binary class  $c$  and  $Ind(y_i = c)$  denotes the set of indices in the batch in which attribute  $i$  has  $y_i = c$ . On the other hand, minimizing this loss with respect to the parameters of the encoder  $E$  encourages the points to be closer to centroids that are near the hyperplane.

### 3.4.5 Prior on latent codes

We regularize the outputs of the encoder to be in the interval  $[-\beta, \beta]$  by  $L_{prior}$ :

$$\min_E \mathbb{E}_x [ReLU(|E(\mathbf{x})| - \beta)] \quad (3.9)$$

### 3.4.6 Optimization

Our final loss is defined as:

$$\mathcal{L}_{CRISPR} = \mathcal{L}_{GAN} + \mathcal{L}_{recon} + \mathcal{L}_{inv} + \mathcal{L}_{attr} + \mathcal{L}_{center} + \sum_i \mathcal{L}_{spec}^i \quad (3.10)$$

We iteratively optimize the model via three repetitive steps. First, we optimize  $\mathcal{L}_{\text{GAN}}, \mathcal{L}_{\text{inv}}, \mathcal{L}_{\text{attr}}$  for two steps with respect to  $D, C_{\text{inv}}, C_{\text{attr}}$  and  $H$ . Next, we optimize  $\mathcal{L}_{\text{prior}}, \mathcal{L}_{\text{GAN}}, \mathcal{L}_{\text{inv}}, \mathcal{L}_{\text{attr}}, \mathcal{L}_{\text{center}}$  for a single step with respect to  $E$ . Finally, we optimize  $\mathcal{L}_{\text{prior}}, \mathcal{L}_{\text{GAN}}, \mathcal{L}_{\text{recon}}, \mathcal{L}_{\text{attr}}, \mathcal{L}_{\text{center}}$  for a single step with respect to  $G$ . All three are optimized using an Adam optimizer (beta=0.5).

### 3.5 Experiments

We evaluate our model on the CelebA dataset. Our goal is to verify that the model produces plausible images that we can sample from the space of possible attribute realizations and observe good diversity (Diverse Swaps) and we can encode attributes from a reference image and borrow those attributes to apply to a query image.

We use the standard train/validation/test splits in the following manner: 2k images were used from the original validation set as the classifier-training set, all 160k images were used to train *CRISPR*, the remaining 14k validation images were used for validation. We used the standard test set. We used binary class pairs (glasses, no glasses) and (mustache, no mustache).

We illustrate examples of **Diverse Swaps** in Figure 3.3 and Figure 3.4. As these images illustrate, our model is able to produce examples via sampling that are not only visually pleasing but also diverse. For example, each row and column in Figure 3.3 illustrates a sample from CRISPR with a different 'bangs' attribute vector. As the images illustrate, some of the samples differ subtly while others



differ significantly. Figure 3.4 shows different samples of the 'eyeglasses' attribute vector. As the results indicate, different vectors result in very different styles of attributes. For example, the final row shows several different glasses of varying shape and shading. The second row also illustrates an interesting failure mode: one side of the eyeglasses is light while the other is dark. This suggests that while our sampling strategy largely produces coherent attributes, we may be sampling from a part of the space that is not atomic: in other words, certain aspects of attributes are being incorrectly broken up in the representation.

Examples of the **Borrows** are shown in Figure 3.5. The first column shows the reference image from which we want to borrow a smile. The second column shows the query image, whose smile we want to alter. The final column shows the output of CRISPR when borrowing the smile attribute vector from the reference image.

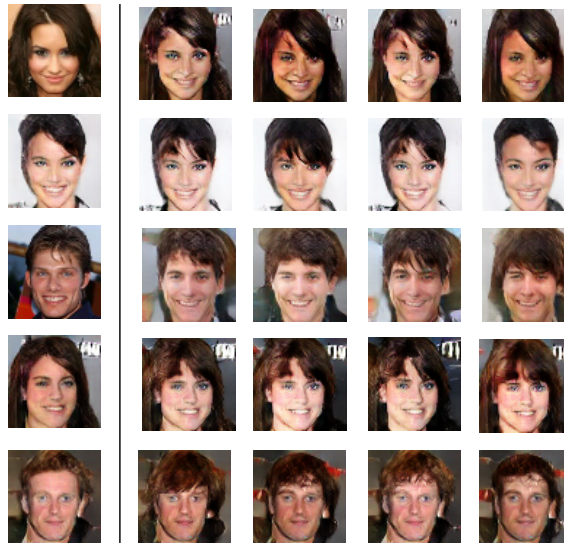


Figure 3.3: Examples illustrating the Diverse Swap operation using the 'bangs' attribute. The left-most column demonstrates the inputs and the remaining columns illustrates the results of sampling from the space of bangs attributes.

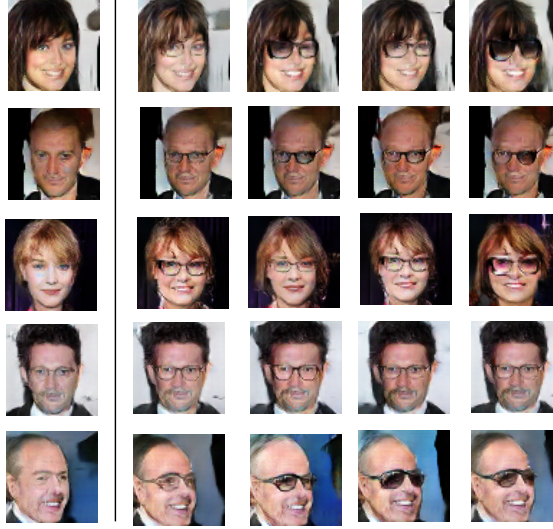


Figure 3.4: Examples illustrating the Diverse Swap operation using the 'eye-glasses' attribute. The left-most column demonstrates the inputs and the remaining columns illustrates the results of sampling from the space of bangs attributes. The results demonstrate not only the diversity of the selection of eye-glasses, but also a failure case (second row) be atomic.

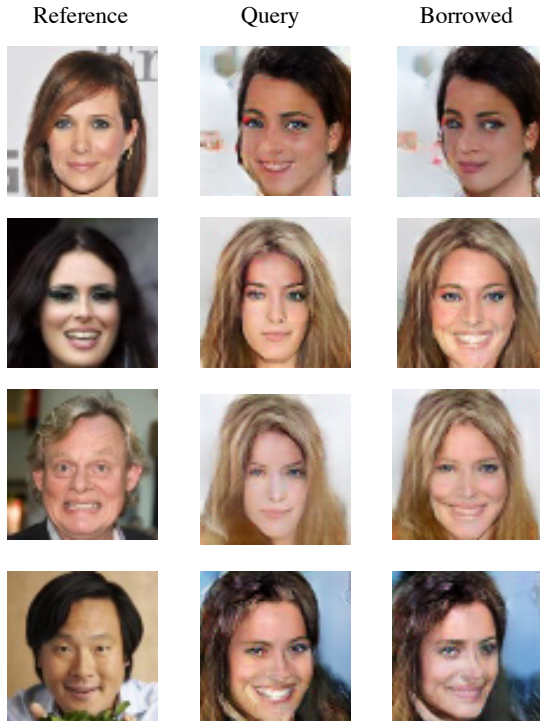


Figure 3.5: Examples illustrating the Borrow operation using the 'smile' attribute'. As these examples illustrate, CRISPR is able to effectively *borrow* the smile from the reference image and apply it to the query image.

## Chapter 4: Model Explanation via Decision Boundary Crossing Transformations

### 4.1 Introduction

Given a classifier, one may ask: What high-level, semantic features of an input is the model using to discriminate between specific classes? Being able to reliably answer this question amounts to an understanding of the classifier’s decision boundary at the level of concepts or attributes, rather than pixel-level statistics.

The ability to produce a conceptual understanding of a model’s decision boundary would be extremely powerful. It would enable researchers to ensure that a model is extracting relevant, high-level concepts, rather than picking up on spurious features of a dataset. For example, criminal justice systems could determine whether their ethical standards were consistent with that of a model [Goodman and Flaxman, 2016]. Additionally, it would provide some measure of validation to consumers (e.g., medical applications, self-driving cars) that a model is making decisions that are difficult to formalize and automatically verify.

Unfortunately, directly visualizing or interpreting decision boundaries in high dimensions is effectively impossible and existing post-hoc interpretation methods

fall short of adequately solving this problem. Dimensionality reduction approaches, such as T-SNE [Maaten and Hinton, 2008], are often highly sensitive to their hyperparameters whose values may drastically alter the visualization [Wattenberg et al., 2016]. Saliency maps are typically designed to highlight the set of pixels that contributed highly to a particular classification. While they can be useful for explaining factors that are present; they cannot adequately describe predictions made due to objects that are missing from the input. Explanation-by-Nearest-Neighbor-Example can indeed demonstrate similar images to a particular query, but there is no guarantee that similar enough images exist to be useful and similarity itself is often ill-defined.

To overcome these limitations, we introduce a novel technique for post-hoc model explanation. Our approach visually explains a model’s decisions by producing images on either side of its decision boundary whose differences are perceptually clear. Such an approach makes it possible for a practitioner to conceptualize how a model is making its decisions at the level of semantics or concepts, rather than vectors or pixels.

Our algorithm is motivated by recent successes in both pixel-wise domain adaptation [Bousmalis et al., 2017, Liu et al., 2017, Zhu et al., 2017] and style transfer [Johnson et al., 2016] in which generative models are used to transform images from one domain to another. Given a pre-trained classifier, we introduce a second, post-hoc explaining network called ExplainGAN, that takes a query image that falls on one side of the decision boundary and produces a transformed version of this image that falls on the other. ExplainGAN exhibits three important properties that make

it ideal for post-hoc model interpretation:

**Easily Visualizable Differences:** Adversarial example [Szegedy et al., 2013] algorithms produce decision boundary crossing images whose differences from the originals are not perceptible, by design. In contrast, our model transforms the input image in a manner that is clearly detectable by the human eye.

**Localized Differences:** Style transfer [Gatys et al., 2015] and domain adaptation approaches typically produce low-level, global changes. If every pixel in the image changes, even slightly, it is not clear which of those changes actually influenced the classifier to produce a different prediction. In contrast, our model yields changes that are spatially localized. Such sparse changes are more easily interpretable by a viewer as fewer elements change.

**Semantically Consistent:** Our model must be consistent with the behavior of the pre-trained classifier to be useful: the class predicted for a transformed image must not match with the predicted class of the original image.

We evaluate our model using standard approaches as well as a new metric for evaluating this novel style of model interpretation by visualizing boundary-crossing transformations. We also utilize a new medical images dataset where the concept of objectness is not well defined, making it less amenable to domain adaptation approaches that hinge on identifying an object and altering / removing it. Furthermore, this dataset represents a clear and practical use-case for model explanation. To summarize, our work makes several contributions:

1. A new approach to model interpretation: visualizing human-interpretable,

decision-boundary crossing images.

2. A new model, ExplainGAN, that produces post-hoc model-explanations via such decision-boundary crossing images.
3. A new metric for evaluating the amount of information retained in decision-boundary crossing transformations.
4. A new and challenging medical image dataset.

## 4.2 Related work

**Post-Hoc Model Interpretation** methods typically seek to provide some kind of visualization of why a model has made a particular decision in terms of the saliency of local regions of an input image. These approaches broadly fall into two main categories: perturbation-based methods and gradient-based methods.

Perturbation-based methods [Zeiler and Fergus, 2014, Fong and Vedaldi, 2017], perturb the input image and evaluate the consequent change in the output of the classifier. Such perturbations remove information from specific regions of the input by applying blur or noise, among other pixel manipulations. Perturbation-based methods require multiple iterations and are computationally more costly than activation-based methods.

The perturbation of finer regions also makes these methods vulnerable to the artifacts of the classifier, potentially resulting in the assignment of high saliency to arbitrary, uninterpretable image regions. In order to combat these artifacts, current

methods such as [Fong and Vedaldi, 2017] are forced to perturb larger, less precise regions of the input.

Gradient-based methods such as [Simonyan et al., 2013, Sundararajan et al., 2017, Shrikumar et al., 2017, Shrikumar et al., 2016, Springenberg et al., 2014] back-propagate the gradient for a given class label to the input image and estimate how moving along the gradient affects the output. Although these methods are computationally more efficient compared to perturbation-based methods, they rely on heuristics for backpropagation and may not support different network architectures.

A subset of gradient-based methods, which we call activation-based methods, also incorporate neuron activations into their explanations. Methods such as Gradient-weighted Class Activation Mapping Grad-CAM [Selvaraju et al., 2016b], layer-wise Relevance Propagation (LRP) [Bach et al., 2015] and Deep Taylor Decomposition (DTD) [Montavon et al., 2017] can be considered as activation-based methods. Grad-CAM visualizes the linear combination of (typically) the last convolution layer and class specific gradients. LRP and DTD decompose the activations of each neuron in terms of contributions (i.e. relevances) from its input.

All these explanation methods are based on identifying pixels which contribute the most to the model output. In other words, these methods explain a model’s decision by illustrating which pixels most affect a classifier’s prediction. This takes the form of an attribution map, a heat map of the same size as the input image, in which each element of the attribution map indicates the degree to which its associated pixel contributed to the model output. In contrast, our model takes a different approach by generating a similar image on the other side of the model’s

decision boundary.

**Adversarial Examples** [Szegedy et al., 2013, Goodfellow et al., 2014b] are created by performing minute perturbations to image pixels to produce decision-boundary crossing transformations which are visually imperceptible to human observers. Such approaches are extremely useful for exploring ways in which a classifier might be attacked. They do not, however, provide any high-level intuition for why a model is making a particular decision.

**Image-to-Image Transformation** approaches, such as those used in domain adaptation [Bousmalis et al., 2017, Liu and Tuzel, 2016, Ganin et al., 2016] have shown increased success in transforming an image in one domain to appear as if drawn from another domain, such as synthetic-to-real or winter-to-summer. These approaches are clearly the most similar to our own in that we seek to transform images predicted as one class to appear to a pre-trained classifier as those from another. These approaches do not, however, constrain the types of transformations allowed and we demonstrate Section 4.5.3 that significant constraints must be applied Section 4.4 to ensure that the transformations produced are easily interpretable. Other image-to-image techniques such as Style Transfer [Zhu et al., 2017, Gatys et al., 2015, Gatys et al., 2016] typically produce very low-level and comprehensive transformations to every pixel. In contrast, our own approach seeks highly localized and high-level, semantic changes.



### 4.3 Model

The goal of our model is to take a pre-trained binary classifier and a query image and generate both a new, transformed image and a binary mask. The transformed image should be similar to the query image, excepting a visually perceptible difference, such that the pre-trained classifier assigns different labels to the query and transformed image. The binary mask indicates which pixels from the query image were changed in order to produce the transformed image. In this way, our model is able to produce a decision-boundary crossing transformation of the query image and illustrate both *where*, via the binary mask, and *how*, via the transformed image, the transformation occurs.

More formally, given a binary classifier  $C(x) \in \{0, 1\}$  operating on an image  $x$ , we seek to learn a function which predicts a transformed image  $t$  and a mask  $m$  such that:

$$C(x) \neq C(t) \tag{4.1}$$

$$x \odot m \neq t \odot m \tag{4.2}$$

$$x \odot \neg m = t \odot \neg m \tag{4.3}$$

where (4.1) indicates that the model believes  $x$  and  $t$  to be of different classes, (4.2) indicates that the query and transformed image differ in pixels whose mask values are 1 and (4.3) indicates that the query and transformed image match in

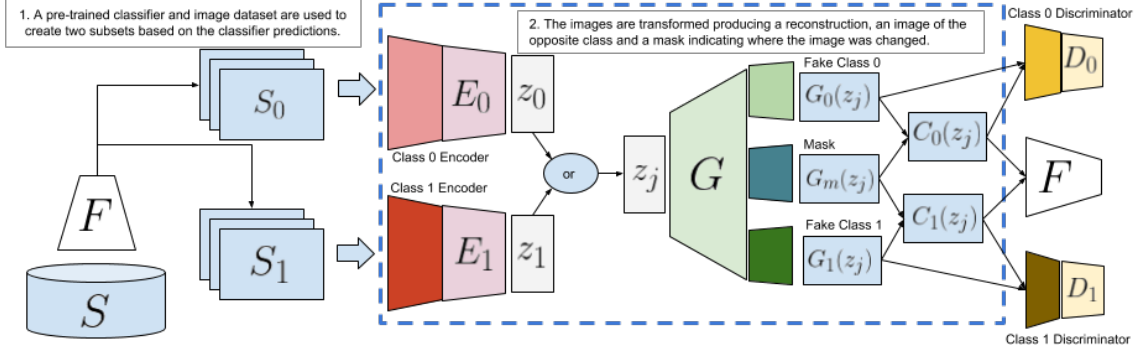


Figure 4.1: Model architecture of ExplainGAN. Inference (in blue frame) consists of passing an image  $x$  of class  $j$  into the appropriate encoder  $E_j$  to produce a hidden vector  $z_j$ . The hidden vector is decoded to simultaneously create its reconstruction  $G_j(z_j)$ , a transformed image of the opposite class  $G_{1-j}(z_j)$  and a mask showing where the changes were made  $G_m(z_j)$ . Composite images  $C_0$  and  $C_1$  merge the reconstruction and transformation with the original image  $x$ .

pixels where mask values are 0.

### 4.3.1 Prerequisites

Given a dataset of images  $S = \{x_i | i \in 1 \dots N\}$ , our pre-trained classifier produces a set of predictions  $\{\bar{y}_i | i \in 1 \dots N\}$ . Given these predictions, we now can split the dataset into two groups  $S_0 = \{x_i | \bar{y}_i = 0\}$  and  $S_1 = \{x_i | \bar{y}_i = 1\}$ .

### 4.3.2 Inference

Given a query image and a predicted label for that image, our model maps to a reconstructed version of that image, an image of the opposite class and a mask that indicates which pixels it changed. Formally, our model is composed of several components. First, our model uses two class-specific encoders to produce hidden codes:

$$z_j = E_j(x) \quad j \in \{0, 1\}, \quad x \in S_j \quad (4.4)$$

Next, a decoder  $G$  maps the hidden representation  $z_j$  to a reconstructed image  $G_j(z_j)$ , a transformed image of the opposite class  $G_{1-j}(z_j)$  and a mask indicating which pixels changed  $G_m(z_j)$ . In this manner, images of either class can be transformed into similar looking images of the opposite class with a visually interpretable change.

We also define the concept of a composite image  $C_j(x)$  of class  $j$ :

$$C_j(x_{1-j}) = x_{1-j} \odot (1 - G_m(z_{1-j})) + G_j(z_{1-j}) \odot G_m(z_{1-j}) \quad (4.5)$$

where  $z_{1-j}$  is the code produced by encoding  $x_{1-j}$ . The composite image uses the mask to blend the original image  $x$  with either the reconstruction or the transformed image.

### 4.3.3 Training

To train the model, several auxiliary components of the network are required. First, two discriminators  $D_j(x) \rightarrow \{\text{real}, \text{fake}\}, j \in \{0, 1\}$  are trained to evaluate between real and fake images of class  $j$ .

To train the model we optimize the following objective:

$$\min_{G, E_0, E_1} \max_{D_0, D_1} \mathcal{L}_{\text{GAN}} + \mathcal{L}_{\text{classifier}} + \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{prior}} \quad (4.6)$$

where  $\mathcal{L}_{\text{GAN}}$  is a typical GAN loss,  $\mathcal{L}_{\text{classifier}}$  is a loss that encourages the generated and composite images to be likely according to the classifier,  $\mathcal{L}_{\text{recon}}$  ensures that the reconstructions are accurate, and  $\mathcal{L}_{\text{prior}}$  encodes our prior for the types of transformations we want to encourage.  $\mathcal{L}_{\text{GAN}}$  is a combination of the GAN losses for each class:

$$\mathcal{L}_{\text{GAN}} = \mathcal{L}_{\text{GAN:0}} + \mathcal{L}_{\text{GAN:1}} \quad (4.7)$$

$\mathcal{L}_{\text{GAN:}j}$  for class  $j$  discriminates between images  $x$  originally classified as class  $j$  and reconstructions of  $x$ , transformations from  $x$  and composites from  $x$ . It is defined as:

$$\begin{aligned} \mathcal{L}_{\text{GAN:}j} = & \mathbb{E}_{\mathbf{x} \sim S_j} \log(D_j(x)) \\ & + \mathbb{E}_{x \sim S_j} [\log(1 - D_j(G_j(E_j(x))))] \\ & + \mathbb{E}_{x \sim S_{1-j}} [\log(1 - D_j(G_j(E_{1-j}(x))))] \\ & + \mathbb{E}_{x \sim S_{1-j}} [\log(1 - D_j(C_j(E_{1-j}(x))))] \end{aligned} \quad (4.8)$$

Note that this formulation, in which the reconstructions of  $x$  are also penalized are part of ensuring that the auto-encoded images are accurate [Larsen et al., 2015] and are included here, rather than as part of  $\mathcal{L}_{\text{recon}}$  out of convenience.

Next, we encourage the composite images to produce images that the classifier correctly predicts:

$$\mathcal{L}_{\text{classifier}} = \mathbb{E}_{x \in S_0} -\log(\text{C}(\text{C}_1(x))) \quad (4.9)$$

$$+ \mathbb{E}_{x \in S_1} -\log(1 - \text{C}(\text{C}_0(x))) \quad (4.10)$$

Finally, we have an auto-encoding loss for the reconstruction:

$$\mathcal{L}_{\text{recon}} = \sum_{j \in \{0,1\}} \mathbb{E}_{x \in S_j} \|G_j(E_j(x)) - x\|^2 \quad (4.11)$$

The mask priors are discussed in the following section.

## 4.4 Priors for Interpretable Image Transformations

There are many image transformations that will transform an image of one class to appear like an image from another class. Not all of these transformations, however, are equally useful for interpreting a model’s behavior at a conceptual level. Adversarial example transformations will change the label but are not perceptible. Style transfer transformations make low-level but not semantic changes. Domain

Adaptation approaches may change every pixel in the image which makes it difficult to determine which of these changes actually influenced the classifier. We want to craft set of priors that encourage transformations that are local to a particular part of the image and visually perceptible. To this end, we define our prior loss term as:

$$\mathcal{L}_{\text{prior}} = \mathcal{L}_{\text{const}} + \mathcal{L}_{\text{count}} + \mathcal{L}_{\text{smoothness}} + \mathcal{L}_{\text{entropy}} \quad (4.12)$$

The **consistency loss**  $\mathcal{L}_{\text{const}}$  ensures that if a pixel is not masked, then the transformed image hasn't altered it.

$$\mathcal{L}_{\text{const}} = \sum_{j \in \{0,1\}} \mathbb{E}_{x \in S_j} [\|(\mathbf{1} - G_m(z_j)) \odot x_j - (\mathbf{1} - G_m(z_j)) \odot G(1-j)(z_j)\|^2] \quad (4.13)$$

where  $z_j = E_j(x)$ . The **count loss**  $\mathcal{L}_{\text{count}}$  allows us to encode prior information regarding a coarse estimate of the number of pixels we anticipate changing. We approximate the  $l_0$  norm via an  $l_1$  norm:

$$\mathcal{L}_{\text{count}} = \sum_{j \in \{0,1\}} \mathbb{E}_{x \in S_j} [\max(\frac{1}{n} |G_m(z_j)|, \kappa)] \quad (4.14)$$

where  $\kappa$  is a constant that corresponds to the ratio of number of changed pixels to the total number of the pixels. The **smoothness loss** encourages masks that are localized by penalizing transitions via a total variation [Rudin et al., 1992] penalty:

$$\mathcal{L}_{\text{smoothness}} = \sum_{j \in \{0,1\}} \mathbb{E}_{x \in S_j} |\nabla G_m(z_j)| \quad (4.15)$$

Finally, we want to encourage the mask to be as binary as possible:

$$\mathcal{L}_{\text{entropy}} = \sum_{j \in \{0,1\}} \mathbb{E}_{x \in S_j} [\min(G_m(z_j), 1 - G_m(z_j))] \quad (4.16)$$

## 4.5 Experiments

Our goal is to provide model explainability via visualization of samples on either side of a model’s decision boundary. This is an entirely new way of performing model explanation and requires a unique approach to evaluation.

To this end, we first demonstrate qualitative results of our approach and compare to related approaches Section 4.5.3. Next, we evaluate our model using traditional criteria by demonstrating that our model’s inferred masks are highly competitive as saliency maps when compared to state-of-the-art attribution approaches Section 4.5.4. Next, we introduce two new metrics for evaluating the explainability of decision-boundary crossing examples Section 4.5.5 and evaluate how our model performs using these quantitative methods.

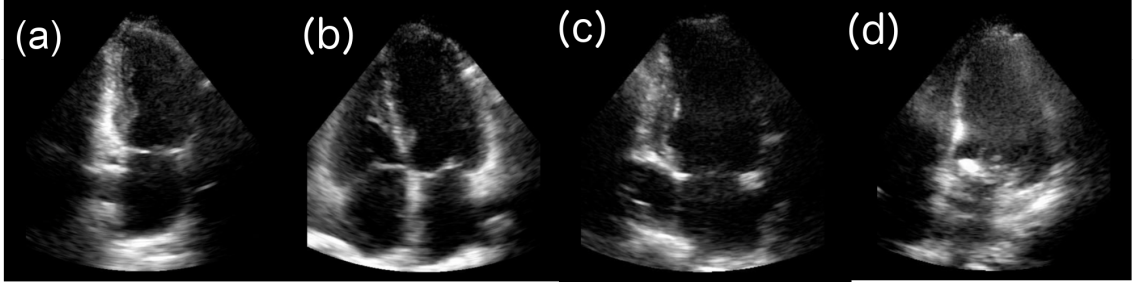


Figure 4.2: An example of Ultrasound images from our Medical Ultrasound dataset. (a) A canonical Apical 2 Chamber view. (b) A canonical Apical 4 Chamber view. (c) A difficult Apical 2 Chamber view that is easily confused for a 4 Chamber view. (d) A difficult Apical 4 Chamber view that is easily confused for a 2 Chamber view.

#### 4.5.1 Datasets

We used four datasets as part of our evaluation: MNIST [LeCun et al., 1998], Fashion-MNIST [Xiao et al., 2017a], CelebA [Liu et al., 2015] and a new Medical Ultrasound dataset that will be released with the publication of this work. For each dataset, four splits were used: A classifier-training set used to train the black-box classifier, a training set used to train ExplainGAN, a validation set used to tune hyperparameters and a test set.

**MNIST, Fashion-MNIST:** We use the standard train/test splits in the following manner: The 60k training set is first split into 3 components: a 2k classifier-training set, a 50k training set and an 8k validation set. We used the standard test set. For MNIST, we used binary class pairs (3, 8), (4, 9) and (5, 6). For Fashion-MNIST, we used binary class pairs (coat, shirt), (pullover, shirt) and (coat, pullover).

**CelebA:** We use the standard train/validation/test splits in the following



manner: 2k images were used from the original validation set as the classifier-training set, all 160k images were used to train ExplainGAN, the remaining 14k validation images were used for validation. We used the standard test set. We used binary class pairs (glasses, no glasses) and (mustache, no mustache).

**Medical Ultrasound:** Our new medical ultrasound dataset is a collection of 72k cardiac images taken from 5 different views of the heart. Each image was labeled by several cardiac sonographers to determine the correct labels. An example of images from the dataset can be found in 4.2. As the Figure illustrates, the dataset is very challenging and is not as amenable to certain senses of 'objectness' found in most standard vision datasets. Of the 72k images, 2k were used as the classifier-training set, 60k were used for training ExplainGAN, 4k were used for validation and 6k were used for testing. We used the binary class pair (Apical 2-Chamber, Apical 4-Chamber).

## 4.5.2 Implementation

The model architecture implementation for  $E$ ,  $G$  and  $D$  is quite similar to the DCGAN architecture [Radford et al., 2015]. We share the last few layers of  $E_0$  and  $E_1$  and the last few layers of  $D_0$  and  $D_1$ . Each loss term in our objective is scaled by a coefficient whose values were obtained via cross-validation. In practice, the coefficients were quite stable across datasets (we use the same set), other than the  $\kappa$  hyperparameter which controls the effect of the count loss and the scaling coefficient for  $\mathcal{L}_{\text{smoothness}}$ , the smoothness loss.

### 4.5.3 Explanation by Qualitative Evaluation

We evaluated our model qualitatively on a number of datasets. We show results on both the Medical Ultrasound dataset and CelebA dataset in 4.3. The use of CelebA and a medical image dataset provides a useful contrast between images whose relationships should be quite familiar to the average reader (glasses vs no-glasses) and relationships that are likely to be foreign to the average reader (apical 2 chamber views versus apical 4 chamber views).

In each block, the “input” column represents images  $x \in S_0$ , the “transformed” column represents ExplainGAN’s transformation,  $G_1(z_0)$ , to the opposite class. The “mask” column illustrates the model’s changes,  $G_m(z_0)$ , and the “composite” column shows the composite images,  $C_1(z_0)$ .

The CelebA (top) results in 4.3 illustrates that the model’s transformations for both “glasses vs no-glasses” and “mustache vs no-mustache” perform highly localized changes and the corresponding mask effectively produces a segmentation of the only visual feature being altered. Furthermore, the model is able to make quite minimal but perceptible changes. For example, in the first row of the “glasses vs no-glasses” task, the mask has preserved the hair over the eyeglasses.

The Ultrasound (bottom) results in 4.3 illustrates that the model has both learned to model the anatomy of the heart and is able to transform from one view of the heart to the other with minimal changes. The transformations and masks clearly illustrate that the model is cuing predominantly on the presence of the right ventricle, but interestingly not the right atrium, and the shape of the pericardium.



Figure 4.3: Qualitative visualization of the ExplainGAN model on two datasets: CelebA and our Medical Ultrasound dataset. The “input” column represents images  $x \in S_0$ , the “transformed” column represents ExplainGAN’s transformation,  $G_1(z_0)$ , to the opposite class. The “mask” column illustrates the model’s changes,  $G_m(z_0)$ , and the “composite” column shows the composite images,  $C_1(z_0)$ . The results indicate that in the case of object-related transformations, such as glasses or mustaches, ExplainGAN effectively performs a weakly supervised segmentation of the object. In the ultrasound case, ExplainGAN illustrates which anatomical areas the model is cuing on: the right ventricle and pericardium.

#### 4.5.4 Explanation via Pixel-Wise Attribution

Many post-hoc explanation methods that use attribution or saliency rely on visual, qualitative comparisons of attribution maps. Recently, [Samek et al., 2017]

introduced a quantitative approach for comparing attribution maps in which pixels are progressively perturbed in the order of predicted saliency. Performance is judged by evaluating which methods require fewer perturbations to affect the classifier’s prediction.

Our model is not designed for attribution / saliency as it produces a binary, rather than continuous mask, which is also paired to a particular transformation image. However, it is possible to loosely interpret our masks as an attribution map in which pixel priority for all pixels in the mask is not known.

While the work of [Samek et al., 2017] perturbed individual pixels, we wanted to avoid a comparison in which individual pixel changes, which are neither themselves interpretable, nor plausible as images, might alter the classification results. Consequently, we adapt the approach of [Samek et al., 2017] by perturbing the image by segments, rather than pixels. To choose the order of perturbation, we normalize the maps to the range  $[0, 1]$ , threshold them with  $t \in [0.5, 0.7, 0.9]$  and segment the resulting binary maps. We then rank the segments based on the average map value within each segment<sup>1</sup>. For perturbation, we replace each pixel in each segment with uniform random noise in the range of the pixel values.

More concretely, we denote the image with  $k$  segments perturbed by  $x_{\text{SP}}^{(k)}$ . We compute the area over the segment perturbation curve (AOSPC) as follows:

$$\text{AOSPC} = \frac{1}{K+1} \left\langle \sum_{k=0}^K f(x_{\text{SP}}^{(0)}) - f(x_{\text{SP}}^{(k)}) \right\rangle_{p_x}, \quad (4.17)$$

---

<sup>1</sup>For ExplainGAN we take the average of the sigmoid outputs over all pixels in a segment.

where  $K$  is the number of steps,  $\langle \cdot \rangle_{p_x}$  denotes the average over all the images, and  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  is the classification function.

We report AOSPC after 10 steps for the explanation methods of Section 5.2 in Section 4.1. We choose the methods to cover the 3 main groups of methods (i.e. perturbation-based, gradient-based and activation-based). A larger AOSPC means that the sensitivity of the segments that are perturbed in 10 steps is higher. To avoid cases where the segmentation assigns all or more than half of the pixels to one segment we choose our threshold from  $\geq 0.5$  values. Our results demonstrate that, despite not being explicitly optimized for finding the most informative pixels, ExplainGAN performs on par with other explanation methods for classifiers. For qualitative comparison of these methods see 4.4.

Table 4.1: AOSPC value (higher is better, see (4.17) after 10 steps for different segmentation thresholds. Although, ExplainGAN is not directly optimized for this metric, its performance is comparable to reasonable baselines for explanation in classifiers. A larger AOSPC means that the sensitivity of the segments that are perturbed in 10 steps is higher.

Dataset	MNIST			Ultrasound		
Threshold	0.5	0.7	0.9	0.5	0.7	0.9
Grad [Shrikumar et al., 2016]	1474	1563	240	712	291	81
Grad-CAM [Selvaraju et al., 2016b]	17.2	8	—	—	70	<b>432</b>
Saliency [Simonyan et al., 2013]	817	718	126	30	63	298
Occlusion [Zeiler and Fergus, 2014]	2099	1946	<b>1486</b>	<b>1215</b>	539	142
LRP [Bach et al., 2015]	1736	1478	244	700	511	71
ExplainGAN	<b>2622</b>	<b>2083</b>	1474	1167	<b>542</b>	374

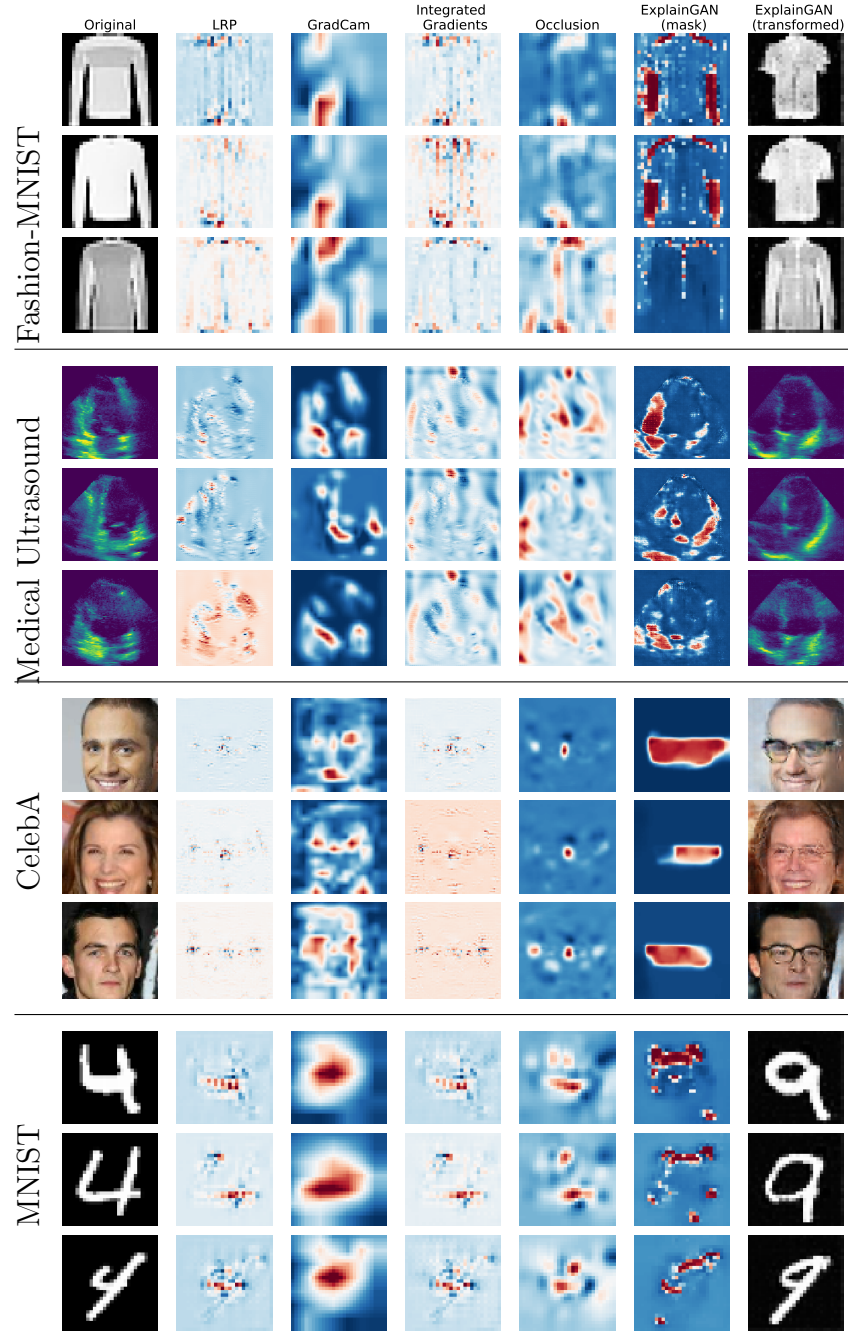


Figure 4.4: Comparison of different methods for explaining the model’s decision. **Fashion-MNIST**: transforming from pullover to shirt, **Ultrasound**: transforming from A2C to A4C (see 4.2 for examples of A2C and A4C views), **CelebA**: transforming from faces without eyeglasses to faces with eyeglasses, **MNIST**: transforming from 4 to 9.



Figure 4.5: Boundary-crossing images have varying explanatory power: images carry more explanatory power if (1) they can be used as substitutes in the original dataset without affecting the classifier and (2) they are different from a query image in small and easily localized ways. (a) displays an image classified as a jacket and not a pullover. (b) shows an image of a pullover which is substitutable and whose localized mask illustrates the models belief that removing a zipper and the jacket ribs would make the original image into a pullover. (c) shows another pullover but non-localized mask doesn’t explain why this is a pullover and not a jacket. (d) shows an adversarial image which is completely unsubstitutable and provides no localized explanation.

Table 4.2: Quantitative substitutability experiments across datasets. Class 0 and Class 1 are the classes that the given classifier is trained to identify. Transformed/Composite 0/1 column shows the accuracy of the classifiers when just transformations/compositions of the images used at training time. Ceiling represents the accuracy of the base classifier on the same test set.

Dataset	Class 0	Class 1	Transformed 0	Transformed 1	Composite 0	Composite 1	Ceiling
Ultrasound	A2C	A4C	95.5	94.2	91.4	95.6	99.6
CelebA	W/O Eyeglasses	W/ Eyeglasses	93.6	96.2	96.05	96.2	96.5
CelebA	W/O Mustache	W/ Mustache	76.65	75.2	74.05	71.4	83.9
CelebA	W/O Black hair	W/ Blackhair	75.65	74.8	79.05	77.4	84.3
FMNIST	Coat	Pullover	75.8	73.7	84.8	69.1	94.1
FMNIST	Coat	Shirt	79.7	78.5	71.8	77.2	91.7
MNIST	Three	Eight	99.6	99.1	99.3	98.9	99.9
MNIST	Four	Nine	98.6	99.0	98.6	98.5	99.0
MNIST	Three	Five	98.5	99.3	98.2	98.2	99.2

#### 4.5.5 Quantitative Assessment of Explainability

Given two similar images on either side of a model’s decision boundary, how can we determine quantitatively whether they provide a conceptual explanation of why a model discriminates between them? There are several high-level criteria that must be met in order for people to find such explanatory images useful.

**Localized but not minimal:** In order for the boundary-crossing image to clear demonstrate what pixels caused a label-changing event, it must deviate from

the original image in a way that is localized to a clear sub-component of the image, as opposed to every pixel changing or only one or two pixels changing.

**Substitutable:** If we are explaining a model by comparing an original image from class A, and a boundary-crossing image is produced to appear like it came from class B, then we define *substitutability* to be the property that we can substitute our boundary-crossing image for one of the original images labeled as class B without affecting our classifier’s performance.

To this end, we propose two metrics aimed at quantifying such an explanations utility. First, the degree to which changes to a query image are localized can be represented by the number of non-zero elements of the mask. Note that while other measures of locality can be used (cohesiveness, connected components), we make no such assumption as we found empirically that often such specific measures do not correlate well with conveying the set of items changing.

Second, we define the substitutability metric as follows: Let an original training set  $\mathcal{D}_{\text{train}} = \{(x_i, y_i) | i = 1..N\}$ , a test set  $\mathcal{D}_{\text{test}}$ , and a classifier  $\mathcal{F}(x) \rightarrow y$  whose empirical performance on the test set is some score  $S$ . Given a new set of model-generated boundary-crossing images  $\mathcal{D}_{\text{trans}} = \{(x'_i, y'_i) | i = 1..N\}$  we say that this set is  $R\%$ -substitutable if our classifier can be retrained using  $\mathcal{D}_{\text{trans}}$  to achieve performance that is  $R\%$  of  $S$ . For example, if our original dataset and classifier yield 90% performance, and we substitute a generated dataset for our original dataset and a re-trained classifier yields 45%, we would say the new dataset is 50% substitutable.

Table 4.2 illustrates the substitutability performance of our model on various datasets. These results illustrate that our model produces images that are nearly



Table 4.3: Substitutability on Ultrasound Dataset. Transformed/Composite 0/1 shows the accuracy of a classifier on test set when the original samples are replaced with Transformed/Composite 0/1 at training phase. Both Transformed/Composite shows the accuracy of the classifier when all of the images are replaced with Transformed/Composite. Note that PixelDA is a oneway transformer.

	Transformed 0	Transformed 1	Both Transformed	Composite 0	Composite 1	Both composite
PixelDA	87.6	N/A	N/A	N/A	N/A	N/A
CycleGAN	94	64	84.1	N/A	N/A	N/A
ExplainGAN-norec	94.5	83.9	96.1	N/A	N/A	N/A
ExplainGAN-nomask	93.9	97.3	95.1	N/A	N/A	N/A
ExplainGAN-full	95.5	94.2	97.3	91.4	95.6	91.4
Ceiling	99.7	99.7	99.7	99.7	99.7	99.7

perfectly substitutable on MNIST, the Ultrasound dataset, and CelebA for the Eyeglasses attribute. That being said, despite compelling qualitative results (Figure 4.4), there is still much room for improvement in terms of substitutability for the other CelebA attributes.

## Chapter 5: Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models

### 5.1 Introduction

Despite their outstanding performance on several machine learning tasks, deep neural networks have been shown to be susceptible to *adversarial attacks* [Szegedy et al., 2013, Goodfellow et al., 2014b]. These attacks come in the form of *adversarial examples*: carefully crafted perturbations added to a legitimate input sample. In the context of classification, these perturbations cause the legitimate sample to be misclassified at inference time [Szegedy et al., 2013, Goodfellow et al., 2014b, Papernot et al., 2016b, Liu et al., 2016]. Such perturbations are often small in magnitude and do not affect human recognition but can drastically change the output of the classifier.

Recent literature has considered two types of threat models: *black-box* and *white-box* attacks. Under the black-box attack model, the attacker does not have access to the classification model parameters; whereas in the white-box attack model, the attacker has complete access to the model architecture and parameters, including potential defense mechanisms [Papernot et al., 2017, Tramèr et al., 2017, Carlini and

Wagner, 2017].

Various defenses have been proposed to mitigate the effect of adversarial attacks. These defenses can be grouped under three different approaches: (1) modifying the training data to make the classifier more robust against attacks, e.g., *adversarial training* which augments the training data of the classifier with adversarial examples [Szegedy et al., 2013, Goodfellow et al., 2014b], (2) modifying the training procedure of the classifier to reduce the magnitude of gradients, e.g., defensive distillation [Papernot et al., 2016], and (3) attempting to remove the adversarial noise from the input samples [Hendrycks and Gimpel, 2017, Meng and Chen, 2017]. All of these approaches have limitations in the sense that they are effective against either white-box attacks or black-box attacks, but not both [Tramèr et al., 2017, Meng and Chen, 2017]. Furthermore, some of these defenses are devised with specific attack models in mind and are not effective against new attacks.

In this chapter, we propose a novel defense mechanism which is effective against both white-box and black-box attacks. We propose to leverage the representative power of GANs [Goodfellow et al., 2014a] to diminish the effect of the adversarial perturbation, by “projecting” input images onto the range of the GAN’s generator prior to feeding them to the classifier. In the GAN framework, two models are trained simultaneously in an adversarial setting: a generative model that emulates the data distribution, and a discriminative model that predicts whether a certain input came from real data or was artificially created. The generative model learns a mapping  $G$  from a low-dimensional vector  $\mathbf{z} \in \mathbb{R}^k$  to the high-dimensional input sample space  $\mathbb{R}^n$ . During training of the GAN,  $G$  is encouraged to generate samples

which resemble the training data. It is, therefore, expected that legitimate samples will be close to some point in the range of  $G$ , whereas adversarial samples will be further away from the range of  $G$ . Furthermore, “projecting” the adversarial examples onto the range of the generator  $G$  can have the desirable effect of reducing the adversarial perturbation. The projected output, computed using Gradient Descent (GD), is fed into the classifier instead of the original (potentially adversarially modified) image. We empirically demonstrate that this is an effective defense against both black-box and white-box attacks on two benchmark image datasets.

The rest of the chapter is organized as follows. We introduce the necessary background regarding known attack models, defense mechanisms, and GANs in Section 5.2. Our defense mechanism, which we call *Defense-GAN*, is formally motivated and introduced in Section 5.3. Finally, experimental results, under different threat models, as well as comparisons to other defenses are presented in Section 5.4.

## 5.2 Related work and background information

In this work, we use GANs for the purpose of defending against adversarial attacks in classification problems. Before detailing our approach in the next section, we explain related work in three parts. First, we discuss different attack models employed in the literature. We, then, go over related defense mechanisms against these attacks and discuss their strengths and shortcomings. Lastly, we explain necessary background information regarding GANs.

### 5.2.1 Attack models and algorithms

Various attack models and algorithms have been used to target classifiers. All attack models we consider aim to find a perturbation  $\delta$  to be added to a (legitimate) input  $\mathbf{x} \in \mathbb{R}^n$ , resulting in the adversarial example  $\tilde{\mathbf{x}} = \mathbf{x} + \delta$ . The  $\ell_\infty$ -norm of the perturbation is denoted by  $\epsilon$  [Goodfellow et al., 2014b] and is chosen to be small enough so as to remain undetectable. We consider two threat levels: black- and white-box attacks.

#### White-box attack models

White-box models assume that the attacker has complete knowledge of all the classifier parameters, i.e., network architecture and weights, as well as the details of any defense mechanism. Given an input image  $\mathbf{x}$  and its associated ground-truth label  $y$ , the attacker thus has access to the loss function  $J(\mathbf{x}, y)$  used to train the network, and uses it to compute the adversarial perturbation  $\delta$ . Attacks can be *targeted*, in that they attempt to cause the perturbed image to be misclassified to a specific target class, or *untargeted* when no target class is specified.

In this work, we focus on untargeted white-box attacks computed using the Fast Gradient Sign Method (FGSM) [Goodfellow et al., 2014b], the Randomized Fast Gradient Sign Method (RAND+FGSM) [Tramèr et al., 2017], and the Carlini-Wagner (CW) attack [Carlini and Wagner, 2017]. Although other attack models exist, such as the Iterative FGSM [Kurakin et al., 2016], the Jacobian-based Saliency Map Attack (JSMA) [Papernot et al., 2016b], and Deepfool [Moosavi-Dezfooli et al.,

2016], we focus on these three models as they cover a good breadth of attack algorithms. FGSM is a very simple and fast attack algorithm which makes it extremely amenable to real-time attack deployment. On the other hand, RAND+FGSM, an equally simple attack, increases the power of FGSM for white-box attacks [Tramèr et al., 2017], and finally, the CW attack is one of the most powerful white-box attacks to-date [Carlini and Wagner, 2017].

Fast Gradient Sign Method (FGSM): Given an image  $\mathbf{x}$  and its corresponding true label  $y$ , the FGSM attack sets the perturbation  $\boldsymbol{\delta}$  to:

$$\boldsymbol{\delta} = \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} J(\mathbf{x}, y)). \quad (5.1)$$

FGSM [Goodfellow et al., 2014b] was designed to be extremely fast rather than optimal. It simply uses the sign of the gradient at every pixel to determine the direction with which to change the corresponding pixel value.

Randomized Fast Gradient Sign Method (RAND+FGSM): The RAND+FGSM [Tramèr et al., 2017] attack is a simple yet effective method to increase the power of FGSM against models which were adversarially trained. The idea is to first apply a small random perturbation before using FGSM. More explicitly, for  $\alpha < \epsilon$ , random noise is first added to the legitimate image  $\mathbf{x}$ :

$$\mathbf{x}' = \mathbf{x} + \alpha \cdot \text{sign}(\mathcal{N}(\mathbf{0}^n, \mathbf{I}^n)). \quad (5.2)$$

Then, the FGSM attack is computed on  $\mathbf{x}'$ , resulting in

$$\tilde{\mathbf{x}} = \mathbf{x}' + (\epsilon - \alpha) \cdot \text{sign}(\nabla_{\mathbf{x}'} J(\mathbf{x}', y)). \quad (5.3)$$

The Carlini-Wagner (CW) attack: The CW attack is an effective optimization-based attack model. In many cases, it can reduce the classifier accuracy to almost 0% [Carlini and Wagner, 2017]. The perturbation  $\boldsymbol{\delta}$  is found by solving an optimization problem of the form:

$$\begin{aligned} \min_{\boldsymbol{\delta} \in \mathbb{R}^n} \quad & \|\boldsymbol{\delta}\|_p + c \cdot f(\mathbf{x} + \boldsymbol{\delta}) \\ \text{subject to} \quad & \mathbf{x} + \boldsymbol{\delta} \in [0, 1]^n, \end{aligned} \quad (5.4)$$

where  $f$  is an objective function that drives the example  $\mathbf{x}$  to be misclassified, and  $c > 0$  is a suitably chosen constant. The  $\ell_2$ ,  $\ell_0$ , and  $\ell_\infty$  norms are considered. We refer the reader to [Carlini and Wagner, 2017] for details regarding the approach to solving (5.4) and setting the constant  $c$ .

## Black-box attack models

For black-box attacks we consider untargeted FGSM attacks computed on a substitute model [Papernot et al., 2017]. As previously mentioned, black-box adversaries have no access to the classifier or defense parameters. It is further assumed that they do not have access to a large training dataset but can query the targeted DNN as a black-box, i.e., access labels produced by the classifier for specific query

images. The adversary trains a model, called *substitute*, which has a (potentially) different architecture than the targeted classifier, using a very small dataset augmented by synthetic images labeled by querying the classifier. Adversarial examples are then found by applying any attack method on the substitute network. It was found that such examples designed to fool the substitute often end up being misclassified by the targeted classifier [Szegedy et al., 2013, Papernot et al., 2017]. In other words, black-box attacks are easily transferrable from one model to the other.

### 5.2.2 Defense mechanisms

Various defense mechanisms have been employed to combat the threat from adversarial attacks. In what follows, we describe one representative defense strategy from each of the three general groups of defenses.

#### Adversarial training

A popular approach to defend against adversarial noise is to augment the training dataset with adversarial examples [Szegedy et al., 2013, Goodfellow et al., 2014b, Moosavi-Dezfooli et al., 2016]. Adversarial examples are generated using one or more chosen attack models and added to the training set. This often results in increased robustness when the attack model used to generate the augmented training set is the same as that used by the attacker. However, adversarial training does not perform as well when a different attack strategy is used by the attacker. Additionally, it tends to make the model more robust to white-box attacks than to



black-box attacks due to *gradient masking* [Papernot et al., 2016c, Papernot et al., 2017, Tramèr et al., 2017].

## Defensive distillation

Defensive distillation [Papernot et al., 2016e] trains the classifier in two rounds using a variant of the distillation [Hinton et al., 2015] method. This has the desirable effect of learning a smoother network and reducing the amplitude of gradients around input points, making it difficult for attackers to generate adversarial examples [Papernot et al., 2016e]. It was, however, shown that, while defensive distillation is effective against white-box attacks, it fails to adequately protect against black-box attacks transferred from other networks [Carlini and Wagner, 2017].

## MagNet

Recently, [Meng and Chen, 2017] introduced MagNet as an effective defense strategy. It trains a *reformer* network (which is an auto-encoder or a collection of auto-encoders) to move adversarial examples closer to the manifold of legitimate, or natural, examples. When using a collection of auto-encoders, one reformer network is chosen at random at test time, thus strengthening the defense. It was shown to be an effective defense against gray-box attacks where the attacker knows everything about the network and defense, except the parameters. MagNet is the closest defense to our approach, as it attempts to reform an adversarial sample using a learnt auto-encoder. The main differences between MagNet and our approach are: (1) we use

GANs instead of auto-encoders, and, most importantly, (2) we use GD minimization to find latent codes as opposed to a feedforward encoder network. This makes Defense-GAN more robust, especially against white-box attacks.

### 5.2.3 Generative Adversarial Networks

Generative Adversarial Networks, originally introduced by [Goodfellow et al., 2014a], consist of two neural networks,  $G$  and  $D$ .  $G : \mathbb{R}^k \rightarrow \mathbb{R}^n$  maps a low-dimensional latent space to the high dimensional sample space of  $\mathbf{x}$ .  $D$  is a binary neural network classifier. In the training phase,  $G$  and  $D$  are typically learned in an adversarial fashion using actual input data samples  $\mathbf{x}$  and random vectors  $\mathbf{z}$ . An isotropic Gaussian prior is usually assumed on  $\mathbf{z}$ . While  $G$  learns to generate outputs  $G(\mathbf{z})$  that have a distribution similar to that of  $\mathbf{x}$ ,  $D$  learns to discriminate between “real” samples  $\mathbf{x}$  and “fake” samples  $G(\mathbf{z})$ .  $D$  and  $G$  are trained in an alternating fashion to minimize the following min-max loss [Goodfellow et al., 2014a]:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (5.5)$$

It was shown that the optimal GAN is obtained when the resulting generator distribution  $p_g = p_{\text{data}}$  [Goodfellow et al., 2014a].

However, GANs turned out to be difficult to train in practice [Gulrajani et al., 2017], and alternative formulations have been proposed. [Arjovsky et al., 2017] introduced Wasserstein GANs (WGANs) which are a variant of GANs that use the

Wasserstein distance, resulting in a loss function with more desirable properties:

$$\min_G \max_D V_W(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[D(\mathbf{x})] - \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[D(G(\mathbf{z}))]. \quad (5.6)$$

In this work, we use WGANs as our generative model due to the stability of their training methods, especially using the approach in [Gulrajani et al., 2017].

### 5.3 Proposed Defense-GAN

We propose a new defense strategy which uses a WGAN trained on legitimate (un-perturbed) training samples to “denoise” adversarial examples. At test time, prior to feeding an image  $\mathbf{x}$  to the classifier, we project it onto the range of the generator by minimizing the reconstruction error  $\|G(\mathbf{z}) - \mathbf{x}\|_2^2$ , using  $L$  steps of GD. The resulting reconstruction  $G(\mathbf{z})$  is then given to the classifier. Since the generator was trained to model the unperturbed training data distribution, we expect this added step to result in a substantial reduction of any potential adversarial noise. We formally motivate this approach in the following section.

#### 5.3.1 Motivation

As mentioned in Section 5.2.3, the GAN min-max loss in (5.5) admits a global optimum when  $p_g = p_{\text{data}}$  [Goodfellow et al., 2014a]. It can be similarly shown that WGAN admits an optimum to its own min-max loss in (5.6), when the set  $\{\mathbf{x} \mid p_g(\mathbf{x}) \neq p_{\text{data}}(\mathbf{x})\}$  has zero Lebesgue-measure. Formally,

**Lemma 1** *A generator distribution  $p_g$  is a global optimum for the WGAN min-max game defined in (5.6), if and only if  $p_g(\mathbf{x}) = p_{data}(\mathbf{x})$  for all  $\mathbf{x} \in \mathbb{R}^n$ , potentially except on a set of zero Lebesgue-measure.*

A sketch of the proof can be found in Section 5.5.

Additionally, it was shown that, if  $G$  and  $D$  have enough capacity to represent the data, and if the training algorithm is such that  $p_g$  converges to  $p_{data}$ , then

$$\mathbb{E}_{\mathbf{x} \sim p_{data}} \left[ \min_{\mathbf{z}} \|G_t(\mathbf{z}) - \mathbf{x}\| \right] \longrightarrow 0 \quad (5.7)$$

where  $G_t$  is the generator of a GAN or WGAN<sup>1</sup> after  $t$  steps of its training algorithm [Kabkab et al., 2018].

This serves to show that, under ideal conditions, the addition of the GAN reconstruction loss minimization step should not affect the performance of the classifier on natural, legitimate samples, as such samples should be almost exactly recovered. Furthermore, we hypothesize that this step will help reduce the adversarial noise which follows a different distribution than that of the GAN training examples.

### 5.3.2 Defense-GAN algorithm

Defense-GAN is a defense strategy to combat both white-box and black-box adversarial attacks against classification networks. At inference time, given a trained GAN generator  $G$  and an image  $\mathbf{x}$  to be classified,  $\mathbf{z}^*$  is first found so as to minimize

---

<sup>1</sup>For simplicity, we will use GAN and WGAN interchangeably in the rest of this manuscript, with the understanding that our implementation follows the WGAN loss.

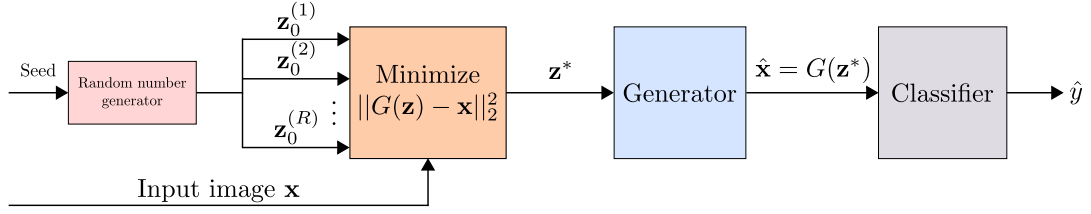


Figure 5.1: Overview of the Defense-GAN algorithm.

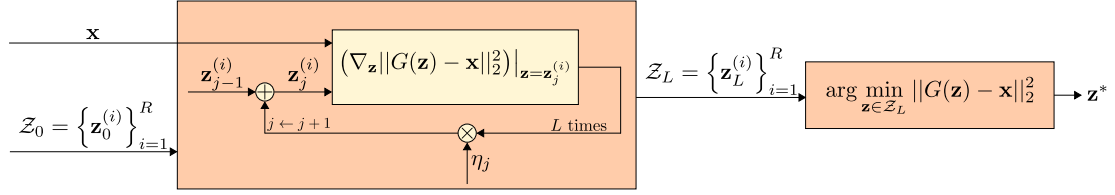


Figure 5.2:  $L$  steps of Gradient Descent are used to estimate the projection of the image onto the range of the generator.

$G(z^*)$  is then given as the input to the classifier. The algorithm is illustrated in Figure 5.1. As (5.8) is a highly non-convex minimization problem, we approximate it by doing a fixed number  $L$  of GD steps using  $R$  different random initializations of  $z$  (which we call random restarts), as shown in Figures 5.1 and 5.2.

The GAN is trained on the available classifier training dataset in an unsupervised manner. The classifier can be trained on the original training images, their reconstructions using the generator  $G$ , or a combination of the two. As was discussed in Section 5.3.1, as long as the GAN is appropriately trained and has enough capacity to represent the data, original clean images and their reconstructions should not differ much. Therefore, these two classifier training strategies should, at least theoretically, not differ in performance.

Compared to existing defense mechanisms, our approach is different in the

following aspects:

1. Defense-GAN can be used in conjunction with any classifier and does not modify the classifier structure itself. It can be seen as an add-on or pre-processing step prior to classification.
2. If the GAN is representative enough, re-training the classifier should not be necessary and any drop in performance due to the addition of Defense-GAN should not be significant.
3. Defense-GAN can be used as a defense to any attack: it does not assume an attack model, but simply leverages the generative power of GANs to reconstruct adversarial examples.
4. Defense-GAN is highly non-linear and white-box gradient-based attacks will be difficult to perform due to the GD loop. A detailed discussion about this can be found in Section [5.6](#).

## 5.4 Experiments

We assume three different attack threat levels:

1. Black-box attacks: the attacker does not have access to the details of the classifier and defense strategy. It therefore trains a substitute network to find adversarial examples.
2. White-box attacks: the attacker knows all the details of the classifier and defense strategy. It can compute gradients on the classifier and defense networks

in order to find adversarial examples.

3. White-box attacks, revisited: in addition to the details of the architectures and parameters of the classifier and defense, the attacker has access to the random seed and random number generator. In the case of Defense-GAN, this means that the attacker knows all the random initializations  $\{\mathbf{z}_0^{(i)}\}_{i=1}^R$ .

We compare our method to adversarial training [Goodfellow et al., 2014b] and MagNet [Meng and Chen, 2017] under the FGSM, RAND+FGSM, and CW (with  $\ell_2$  norm) white-box attacks, as well as the FGSM black-box attack. Details of all network architectures used in this chapter can be found in Section 5.7. When the classifier is trained using the reconstructed images ( $G(\mathbf{z}^*)$ ), we refer to our method as Defense-GAN-Rec, and we use Defense-GAN-Orig when the original images ( $\mathbf{x}$ ) are used to train the classifier. Our GAN follows the WGAN training procedure in [Gulrajani et al., 2017], and details of the generator and discriminator network architectures are given in Table 5.6. The reformer network (encoder) for the MagNet baseline is provided in Table 5.7. Our implementation is based on TensorFlow [Abadi et al., 2015b] and builds on open-source software: CleverHans by [Papernot et al., 2016a] and improved WGAN training by [Gulrajani et al., 2017]. We use machines equipped with NVIDIA GeForce GTX TITAN X GPUs.

In our experiments, we use two different image datasets: the MNIST handwritten digits dataset [LeCun et al., 1998] and the Fashion-MNIST (F-MNIST) clothing articles dataset [Xiao et al., 2017b]. Both datasets consist of 60,000 training images and 10,000 testing images. We split the training images into a training set of 50,000

images and hold-out a validation set containing 10,000 images. For white-box attacks, the testing set is kept the same (10,000 samples). For black-box attacks, the testing set is divided into a small hold-out set of 150 samples reserved for adversary substitute training, as was done in [Papernot et al., 2017], and the remaining 9,850 samples are used for testing the different methods.

#### 5.4.1 Results on black-box attacks

In this section, we present experimental results on FGSM black-box attacks. As previously mentioned, the attacker trains a substitute model, which could differ in architecture from the targeted model, using a limited dataset consisting of 150 legitimate images augmented with synthetic images labeled using the target classifier. The classifier and substitute model architectures used and referred to throughout this section are described in Table 5.5.

In Tables 5.1 and 5.2, we present our classification accuracy results and compare to other defense methods. As can be seen, FGSM black-box attacks were successful at reducing the classifier accuracy by up to 70%. All considered defense mechanisms are relatively successful at diminishing the effect of the attacks. We note that, as expected, the performance of Defense-GAN-Rec and that of Defense-GAN-Orig are very close. In addition, they both perform consistently well across different classifier and substitute model combinations. MagNet also performs in a consistent manner, but achieves lower accuracy than Defense-GAN. Two adversarial training defenses are presented: the first one obtains the adversarial examples



assuming the same attack  $\epsilon = 0.3$ , and the second assumes a different  $\epsilon = 0.15$ . With incorrect knowledge of  $\epsilon$ , the performance of adversarial training generally decreases. In addition, the classification performance of this defense method has very large variance across the different architectures. It is worth noting that adversarial training defense is only fit against FGSM attacks, because the adversarially augmented data, even with a different  $\epsilon$ , is generated using the same method as the black-box attack (FGSM). In contrast, Defense-GAN and MagNet are general defense mechanisms which do not assume a specific attack model.

The performances of defenses on the F-MNIST dataset, shown in Table 5.2, are noticeably lower than on MNIST. This is due to the large  $\epsilon = 0.3$  in the FGSM attack. Please see Appendix 5.8 for qualitative examples showing that  $\epsilon = 0.3$  represents very high noise, which makes F-MNIST images difficult to classify, even by a human.

In addition, the Defense-GAN parameters used in this experiment were kept the same for both Tables, in order to study the effect of dataset complexity, and can be further optimized as investigated in the next section.

### Effect of number of GD iterations $L$ and random restarts $R$

Figure 5.3 shows the effect of varying the number of GD iterations  $L$  as well as the random restarts  $R$  used to compute the GAN reconstructions of input images. Across different  $L$  and  $R$  values, Defense-GAN-Rec and Defense-GAN-Orig have comparable performance. Increasing  $L$  has the expected effect of improving

Table 5.1: Classification accuracies of different classifier and substitute model combinations using various defense strategies on the MNIST dataset, under FGSM black-box attacks with  $\epsilon = 0.3$ . Defense-GAN has  $L = 200$  and  $R = 10$ .

Classifier/ Substitute	No Attack	No Defense	<b>Defense- GAN-Rec</b>	<b>Defense- GAN-Orig</b>	MagNet	Adv. Tr. $\epsilon = 0.3$	Adv. Tr. $\epsilon = 0.15$
A/B	0.9970	0.6343	<u>0.9312</u>	0.9282	0.6937	<b>0.9654</b>	0.6223
A/E	0.9970	0.5432	0.9139	0.9221	0.6710	<b>0.9668</b>	<u>0.9327</u>
B/B	0.9618	0.2816	<u>0.9057</u>	<b>0.9105</b>	0.5687	0.2092	0.3441
B/E	0.9618	0.2128	<u>0.8841</u>	<b>0.8892</b>	0.4627	0.1120	0.3354
C/B	0.9959	0.6648	<u>0.9357</u>	0.9322	0.7571	<b>0.9834</b>	0.9208
C/E	0.9959	0.8050	0.9223	0.9182	0.6760	<b>0.9843</b>	<u>0.9755</u>
D/B	0.9920	0.4641	<u>0.9272</u>	<b>0.9323</b>	0.6817	0.7667	0.8514
D/E	0.9920	0.3931	<b>0.9164</b>	<u>0.9155</u>	0.6073	0.7676	0.7129

Table 5.2: Classification accuracies of different classifier and substitute model combinations using various defense strategies on the F-MNIST dataset, under FGSM black-box attacks with  $\epsilon = 0.3$ . Defense-GAN has  $L = 200$  and  $R = 10$ .

Classifier/ Substitute	No Attack	No Defense	<b>Defense- GAN-Rec</b>	<b>Defense- GAN-Orig</b>	MagNet	Adv. Tr. $\epsilon = 0.3$	Adv. Tr. $\epsilon = 0.15$
A/B	0.9346	0.5131	0.586	0.5803	0.5404	<b>0.7393</b>	<u>0.6600</u>
A/E	0.9346	0.3653	0.4790	0.4616	0.3311	<b>0.6945</b>	<u>0.5638</u>
B/B	0.7470	0.4017	<u>0.4940</u>	<b>0.5530</b>	0.3812	0.3177	0.3560
B/E	0.7470	0.3123	<u>0.3720</u>	<b>0.4187</b>	0.3119	0.2617	0.2453
C/B	0.9334	0.2635	0.5289	0.6079	0.4664	<b>0.7791</b>	<u>0.6838</u>
C/E	0.9334	0.2066	0.4871	0.4625	0.3016	<b>0.7504</b>	<u>0.6655</u>
D/B	0.8923	0.4541	0.5779	0.5853	0.5478	<u>0.6172</u>	<b>0.6395</b>
D/E	0.8923	0.2543	0.4007	0.4730	0.3396	<b>0.5093</b>	<u>0.4962</u>

performance when no attack is present. Interestingly, with an FGSM attack, the classification performance decreases after a certain  $L$  value. With too many GD iterations on the mean squared error (MSE)  $\|G(\mathbf{z}) - (\mathbf{x} + \boldsymbol{\delta})\|_2^2$ , some of the adversarial noise components are retained. In the right Figure, the effect of varying  $R$  is shown to be extremely pronounced. This is due to the non-convex nature of the MSE, and increasing  $R$  enables us to sample different local minima.

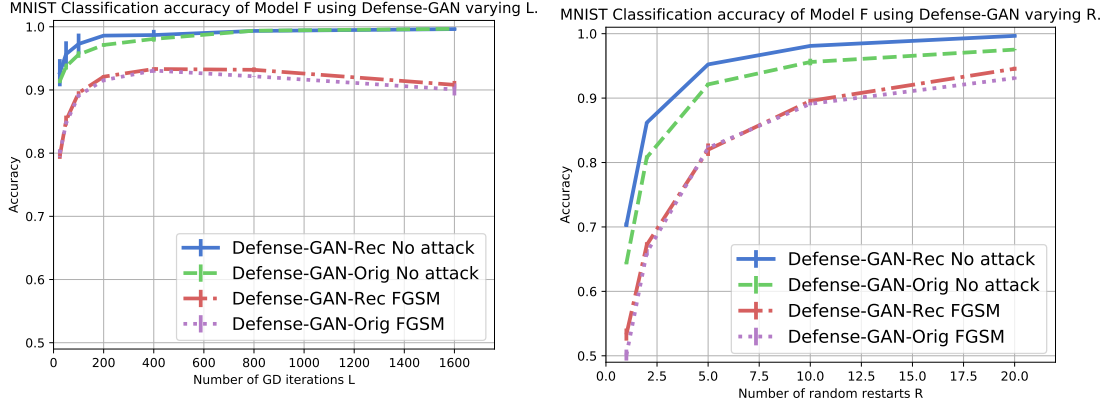


Figure 5.3: Classification accuracy of Model F using Defense-GAN on the MNIST dataset, under FGSM black-box attacks with  $\epsilon = 0.3$  and substitute Model E. Left: various number of iterations  $L$  are used ( $R = 10$ ). Right: various number of random restarts  $R$  are used ( $L = 100$ ).

### Effect of adversarial noise norm $\epsilon$

We now investigate the effect of changing the attack  $\epsilon$  in Table 5.3. As expected, with higher  $\epsilon$ , the FGSM attack is more successful, especially on the F-MNIST dataset where the noise norm seems to have a more pronounced effect with nearly 37% drop in performance between  $\epsilon = 0.1$  and 0.3. Figure 5.7 in Section 5.8 shows adversarial samples as well as their reconstructions with Defense-GAN at different values of  $\epsilon$ . We can see that for large  $\epsilon$ , the class is difficult to discern, even for the human eye.

Even though it seems that increasing  $\epsilon$  is a desirable strategy for the attacker, this increases the likelihood that the adversarial noise is discernible and therefore the attack is detected. It is trivial for the attacker to provide adversarial images at very high  $\epsilon$ , and a good measure of an attack’s strength is its ability to affect performance at low  $\epsilon$ . In fact, in the next section, we discuss how Defense-GAN can

be used to not only diminish the effect of attacks, but to also detect them.

Table 5.3: Classification accuracy of Model F using Defense-GAN ( $L = 400$ ,  $R = 10$ ), under FGSM black-box attacks for various noise norms  $\epsilon$  and substitute Model E.

$\epsilon$	Defense-GAN-Rec MNIST	Defense-GAN-Rec F-MNIST
0.10	$0.9864 \pm 0.0011$	$0.8844 \pm 0.0017$
0.15	$0.9836 \pm 0.0026$	$0.8267 \pm 0.0065$
0.20	$0.9772 \pm 0.0019$	$0.7492 \pm 0.0170$
0.25	$0.9641 \pm 0.0001$	$0.6384 \pm 0.0159$
0.30	$0.9307 \pm 0.0034$	$0.5126 \pm 0.0096$

## Attack detection

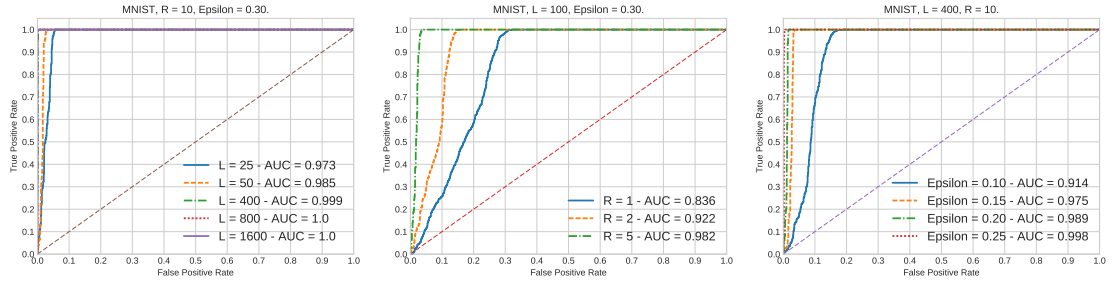


Figure 5.4: ROC Curves when using Defense-GAN MSE for FGSM attack detections on the MNIST dataset (Classifier Model F, Substitute Model E). Left: Results for various number of GD iterations are shown with  $R = 10$ ,  $\epsilon = 0.30$ . Middle: Results for various number of random restarts  $R$  are shown with  $L = 100$ ,  $\epsilon = 0.30$ . Right: Results for various  $\epsilon$  are shown with  $L = 400$ ,  $R = 10$ .

We intuitively expect that clean, unperturbed images will lie closer to the range of the Defense-GAN generator  $G$  than adversarial examples. This is due to the fact that  $G$  was trained to produce images which resemble the legitimate data. In light of this observation, we propose to use the MSE of an image with its reconstruction from (??) as a “metric” to decide whether or not the image

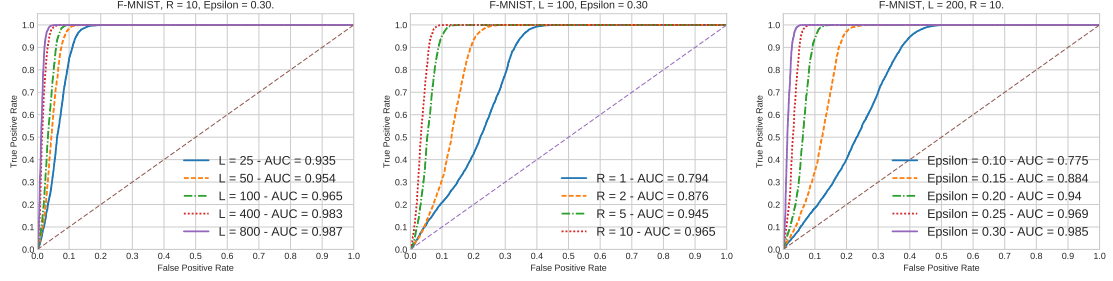


Figure 5.5: ROC Curves when using Defense-GAN MSE for FGSM attack detections on the F-MNIST dataset (Classifier Model F, Substitute Model E). Left: Results for various number of GD iterations are shown with  $R = 10$ ,  $\epsilon = 0.30$ . Middle: Results for various number of random restarts  $R$  are shown with  $L = 100$ ,  $\epsilon = 0.30$ . Right: Results for various  $\epsilon$  are shown with  $L = 200$ ,  $R = 10$ .

was adversarially manipulated. In order words, for a given threshold  $\theta > 0$ , the hypothesis test is:

$$\|G(\mathbf{z}^*) - \mathbf{x}\|_2^2 \underset{\text{no attack}}{\overset{\text{attack}}{\gtrless}} \theta. \quad (5.8)$$

We compute the reconstruction MSEs for every image from the test dataset, and its adversarially manipulated version using FGSM. We show the Receiver Operating Characteristic (ROC) curves as well as the Area Under the Curve (AUC) metric for different Defense-GAN parameters and  $\epsilon$  values in Figures 5.4 and 5.5. The results show that this attack detection strategy is effective especially when the number of GD iterations  $L$  and random restarts  $R$  are large. From the left and middle Figures, we can conclude that the number of random restarts plays a very important role in the detection false positive and true positive rates as was discussed in Section 5.4.1. Furthermore, when  $\epsilon$  is very small, it becomes difficult to detect attacks at low false positive rates.

## Results on white-box attacks

We now present results on white-box attacks using three different strategies: FGSM, RAND+FGSM, and CW. We perform the CW attack for 100 iterations of projected GD, with learning rate 10.0, and use  $c = 100$  in equation (5.4). Table 5.4 shows the classification performance of different classifier models across different attack and defense strategies. We note that Defense-GAN significantly outperforms the two other baseline defenses. We even give the adversarial attacker access to the random initializations of  $\mathbf{z}$ . However, we noticed that the performance does not change much when the attacker does not know the initialization. Adversarial training was done using FGSM to generate the adversarial samples. It is interesting to mention that when CW attack is used, adversarial training performs extremely poorly. As previously discussed, adversarial training does not generalize well against different attack methods.

Due to the loop of  $L$  steps of GD, Defense-GAN is resilient to GD-based white-box attacks, since the attacker needs to “un-roll” the GD loop and propagate the gradient of the loss all the way across  $L$  steps. In fact, from Table 5.4, the performance of classifier A with Defense-GAN on the MNIST dataset drops less than 1% from 0.997 to 0.988 under FGSM. In comparison, from Figure 5.8, when  $L = 25$ , the performance of the same network drops to 0.947 (more than 5% drop). This shows that using a larger  $L$  significantly increases the robustness of Defense-GAN against GD-based white-box attacks. This comes at the expense of increased inference time complexity. We present a more detailed discussion about the difficulty of GD-based

white-box attacks in Section 5.6 and time complexity in Section 5.11. Additional white-box experimental results on higher-dimensional images are reported in 5.10.

## 5.5 Optimality of $p_g = p_{\text{data}}$ for WGANs

Sketch of proof of Lemma 1: The WGAN min-max loss is given by:

$$V_W(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[D(\mathbf{x})] - \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[D(G(\mathbf{z}))] \quad (5.9)$$

$$= \int_{\mathbf{x}} p_{\text{data}}(\mathbf{x}) D(\mathbf{x}) d\mathbf{x} - \int_{\mathbf{z}} p_{\mathbf{z}}(\mathbf{z}) D(G(\mathbf{z})) d\mathbf{z} \quad (5.10)$$

$$= \int_{\mathbf{x}} (p_{\text{data}}(\mathbf{x}) - p_g(\mathbf{x})) D(\mathbf{x}) d\mathbf{x} \quad (5.11)$$

For a fixed  $G$ , the optimal discriminator  $D$  which maximizes  $V_W(D, G)$  is such that:

$$D_G^*(\mathbf{x}) = \begin{cases} 1 & \text{if } p_{\text{data}}(\mathbf{x}) \geq p_g(\mathbf{x}) \\ 0 & \text{otherwise} \end{cases} \quad (5.12)$$

Plugging  $D_G^*$  back into (5.11), we get:

$$V_W(D_G^*, G) = \int_{\mathbf{x}} (p_{\text{data}}(\mathbf{x}) - p_g(\mathbf{x})) D_G^*(\mathbf{x}) d\mathbf{x} \quad (5.13)$$

$$= \int_{\{\mathbf{x} \mid p_{\text{data}}(\mathbf{x}) \geq p_g(\mathbf{x})\}} (p_{\text{data}}(\mathbf{x}) - p_g(\mathbf{x})) d\mathbf{x} \quad (5.14)$$

Let  $\mathcal{X} = \{\mathbf{x} \mid p_{\text{data}}(\mathbf{x}) \geq p_g(\mathbf{x})\}$ . Clearly, to minimize (5.14), we need to set  $p_{\text{data}}(\mathbf{x}) = p_g(\mathbf{x})$  for  $\mathbf{x} \in \mathcal{X}$ . Then, since both pdfs should integrate to 1,

$$\int_{\mathcal{X}^c} p_g(\mathbf{x}) d\mathbf{x} = \int_{\mathcal{X}^c} p_{\text{data}}(\mathbf{x}) d\mathbf{x} \quad (5.15)$$

However, this is a contradiction since  $p_g(\mathbf{x}) < p_{\text{data}}(\mathbf{x})$  for  $\mathbf{x} \in \mathcal{X}^c$ , unless  $\mu(\mathcal{X}^c) = 0$  where  $\mu$  is the Lebesgue measure. This concludes the proof.

## 5.6 Difficulty of GD-based white-box attacks on Defense-GAN

In order to perform a GD-based white-box attack on models using Defense-GAN, an attacker needs to compute the gradient of the output of the classifier with respect to the input. From Figure 5.1, the generator and the classifier can be seen as one, combined, feedforward network, through which it is easy to propagate gradients. The difficulty lies in the orange box of the GD optimization detailed in Figure 5.2.

For the sake of simplicity, let's assume that  $R = 1$ . Define  $\mathcal{L}(\mathbf{x}, \mathbf{z}) = \|G(\mathbf{z}) - \mathbf{x}\|_2^2$ . Then  $\mathbf{z}^* = \mathbf{z}_L$ , which is computed recursively as follows:

$$\mathbf{z}_1 = \mathbf{z}_0 + \eta_0 \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{z}_0} \quad (5.16)$$

$$\mathbf{z}_2 = \mathbf{z}_1 + \eta_1 \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{z}_1} \quad (5.17)$$

$$= \mathbf{z}_0 + \eta_0 \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{z}_0} + \eta_1 \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{z}_0 + \eta_0 \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{z}_0}} \quad (5.18)$$

and so on. Therefore, computing the gradient of  $\mathbf{z}^*$  with respect to  $\mathbf{x}$  involves a



large number ( $L$ ) of recursive chain rules and high-dimensional Jacobian tensors. This computation gets increasingly prohibitive for large  $L$ .

## 5.7 Neural network architectures

We describe the neural network architectures used throughout the chapter. The detail of models A through F used for classifier and substitute networks can be found in Table 5.5. In Table 5.6, the GAN architectures are described, and in Table 5.7, the encoder architecture for the MagNet baseline is given. In what follows:

- $\text{Conv}(m, k \times k, s)$  refers to a convolutional layer with  $m$  feature maps, filter size  $k \times k$ , and stride  $s$
- $\text{ConvT}(m, k \times k)$  refers to the transpose (gradient) of  $\text{Conv}$  (sometimes referred to as “deconvolution”) with  $m$  feature maps, filter size  $k \times k$ , and stride  $s$
- $\text{FC}(m)$  refers to a fully-connected layer with  $m$  outputs
- $\text{Dropout}(p)$  refers to a dropout layer with probability  $p$
- $\text{ReLU}$  refers to the Rectified Linear Unit activation
- $\text{LeakyReLU}(\alpha)$  is the leaky version of the Rectified Linear Unit with parameter  $\alpha$

Table 5.5: Neural network architectures used for classifiers and substitute models.

A	B, F*	C	D, E*
Conv(64, $5 \times 5$ , 1)	Dropout(0.2)	Conv(128, $3 \times 3$ , 1)	FC(200)
ReLU	Conv(64, $8 \times 8$ , 2)	ReLU	ReLU
Conv(64, $5 \times 5$ , 2)	ReLU	Conv(64, $3 \times 3$ , 2)	Dropout(0.5)
ReLU	Conv(128, $6 \times 6$ , 2)	ReLU	FC(200)
Dropout(0.25)	ReLU	Dropout(0.25)	ReLU
FC(128)	Conv(128, $5 \times 5$ , 1)	FC(128)	Dropout(0.5)
ReLU	ReLU	ReLU	FC(10) + Softmax
Dropout(0.5)	Dropout(0.5)	Dropout(0.5)	
FC(10) + Softmax	FC(10) + Softmax	FC(10) + Softmax	

[ \* : F (resp. E) shares the same architecture as B (resp. D) with the dropout layers removed ]

## 5.8 Qualitative examples

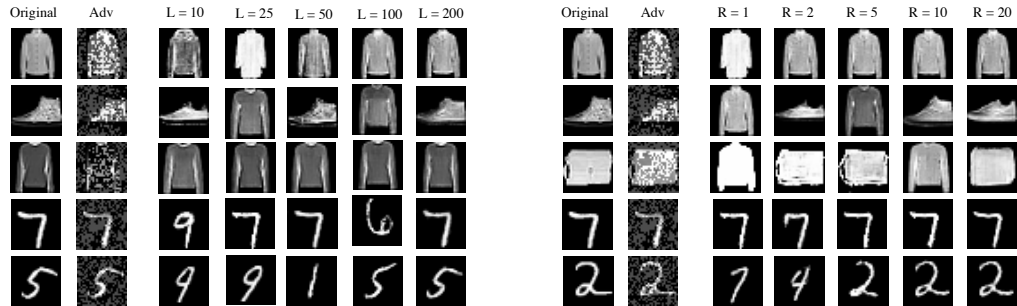


Figure 5.6: Examples from MNIST and F-MNIST. Left: Original, FGSM adversarial  $\epsilon = 0.3$ , and reconstruction images for  $R = 1$  and various  $L$  are shown. Right: Original, FGSM adversarial  $\epsilon = 0.3$ , and reconstruction images for  $L = 25$  and various  $R$  are shown.

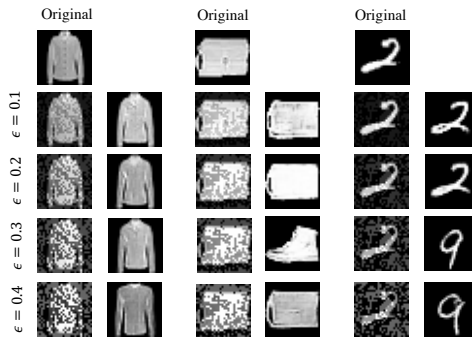


Figure 5.7: Examples from MNIST and F-MNIST: Original, FGSM adversarial and reconstruction images for  $L = 50$ ,  $R = 15$  and various  $\epsilon$  are shown.

## 5.9 Additional results on the effect of varying the number of GD iterations $L$ and random restarts $R$

Table 5.8: Classification accuracy of Model F using Defense-GAN with various number of iterations  $L$  ( $R = 10$ ), on the MNIST dataset, under FGSM black-box attack with  $\epsilon = 0.3$ .

$L$	Defense-GAN-Rec	Defense-GAN-Orig	Defense-GAN-Rec	Defense-GAN-Orig
	No attack	No attack	Adversarial	Adversarial
25	$0.9273 \pm 0.0215$	$0.9141 \pm 0.0033$	$0.7955 \pm 0.0045$	$0.7998 \pm 0.0063$
50	$0.9567 \pm 0.0203$	$0.9371 \pm 0.0048$	$0.8516 \pm 0.0078$	$0.8472 \pm 0.0026$
100	$0.9728 \pm 0.0164$	$0.9560 \pm 0.0051$	$0.8953 \pm 0.0027$	$0.8911 \pm 0.0024$
200	$0.9860 \pm 0.0010$	$0.9712 \pm 0.0028$	$0.9210 \pm 0.0023$	$0.9155 \pm 0.0032$
400	$0.9869 \pm 0.0082$	$0.9808 \pm 0.0044$	$0.9332 \pm 0.0027$	$0.9307 \pm 0.0034$
800	$0.9934 \pm 0.0009$	$0.9938 \pm 0.0004$	$0.9319 \pm 0.0038$	$0.9216 \pm 0.0005$
1600	$0.9963 \pm 0.0013$	$0.9967 \pm 0.0005$	$0.9081 \pm 0.0062$	$0.9008 \pm 0.0095$

Table 5.11: Classification accuracy of Model F using Defense-GAN with various number of random restarts  $R$  ( $L = 100$ ), on the F-MNIST dataset, under FGSM black-box attack with  $\epsilon = 0.3$ .

$R$	Defense-GAN-Rec	Defense-GAN-Orig	Defense-GAN-Rec	Defense-GAN-Orig
	No attack	No attack	Adversarial	Adversarial
1	$0.8425 \pm 0.0008$	$0.5597 \pm 0.0015$	$0.3504 \pm 0.0102$	$0.3380 \pm 0.0043$
2	$0.8994 \pm 0.0051$	$0.7793 \pm 0.0023$	$0.4050 \pm 0.0148$	$0.3508 \pm 0.0167$
5	$0.9260 \pm 0.0028$	$0.6726 \pm 0.0006$	$0.4521 \pm 0.0177$	$0.4024 \pm 0.0085$
10	$0.9101 \pm 0.0032$	$0.8190 \pm 0.0043$	$0.4808 \pm 0.0088$	$0.4221 \pm 0.0255$

Figure 5.8: Classification accuracy of different models using Defense-GAN on the MNIST dataset, under FGSM white-box attack with  $\epsilon = 0.3$ , for various number of iterations  $L$  and  $R = 10$ .

## 5.10 Additional results on white-box attacks

We report results on white-box attacks on the CelebFaces Attributes dataset (CelebA) [Liu et al., 2015] in Table 5.12. The CelebA dataset is a large-scale face dataset consisting of more than 200,000 face images, split into training, validation, and testing sets. The RGB images were center-cropped and resized to  $64 \times 64$ . We performed the task of gender classification on this dataset. The GAN architecture is the same as that in Table 5.6, except for an additional ConvT(128,  $5 \times 5$ , 1) layer in the generator network.

## 5.11 Time complexity

The computational complexity of reconstructing an image using Defense-GAN is on the order of the number of GD iterations performed to estimate  $\mathbf{z}^*$ , multiplied by the time to compute gradients. The number of random restarts  $R$  has less effect on the running time, since random restarts are independent and can run in parallel if enough resources are available. Table 5.13 shows the average running time, in seconds, to find the reconstructions of MNIST and F-MNIST images on one NVIDIA GeForce GTX TITAN X GPU. For most applications, these running times are not prohibitive. We can see a tradeoff between running time and defense robustness as well as accuracy.

Table 5.13: Average time, in seconds, to compute reconstructions of MNIST/F-MNIST images for various values of  $L$  and  $R$ .

	$L = 10$	$L = 25$	$L = 50$	$L = 100$	$L = 200$
$R = 1$	$0.043 \pm 0.027$	$0.070 \pm 0.003$	$0.137 \pm 0.004$	$0.273 \pm 0.006$	$L = 0.543 \pm 0.017$
$R = 2$	$0.042 \pm 0.026$	$0.067 \pm 0.002$	$0.131 \pm 0.003$	$0.261 \pm 0.006$	$L = 0.510 \pm 0.006$
$R = 5$	$0.043 \pm 0.029$	$0.070 \pm 0.002$	$0.136 \pm 0.004$	$0.270 \pm 0.004$	$L = 0.535 \pm 0.008$
$R = 10$	$0.051 \pm 0.032$	$0.086 \pm 0.001$	$0.170 \pm 0.002$	$0.338 \pm 0.008$	$L = 0.675 \pm 0.016$
$R = 20$	$0.060 \pm 0.035$	$0.105 \pm 0.003$	$0.209 \pm 0.006$	$0.414 \pm 0.012$	$L = 0.825 \pm 0.022$

## 5.12 An unsuccessful attempt to attack Defense-GAN

Among seven defense methods that are proposed in 2018 International Conference on Representation Learning (ICLR), this work is the only method that is reported as not broken in the best paper award winner of 2018 International Conference on Machine Learning (ICML) [Athalye et al., 2018]. In there, they first approximate Defense-GAN with a differentiable function  $g(x)$  and use the gradients of  $g(x)$  as an approximation to the gradients of Defense-GAN at test time. This method is the same as the black-box attack methods mentioned in Section 5.2.1 and they are not effective for attacking Defense-GAN. Their partial success is due to the non-optimal setting of hyperparameters that they used at test time.

In general, the attacks that try to approximate Defense-GAN fail because of the non-convex nature of the optimization that is solved at test time. Defense-GAN does not reach the solutions that are found from the approximation methods since the initialization, learning rate, and the number of steps are essential parameters of the optimization which are usually discarded by such methods.

Table 5.4: Classification accuracies of different classifier models using various defense strategies on the MNIST (top) and F-MNIST (bottom) datasets, under FGSM, RAND+FGSM, and CW white-box attacks. Defense-GAN has  $L = 200$  and  $R = 10$ .

Attack	Classifier Model	No Attack	No Defense	Defense-GAN-Rec	MagNet	Adv. Tr. $\epsilon = 0.3$
FGSM $\epsilon = 0.3$	A	0.997	0.217	<b>0.988</b>	0.191	<u>0.651</u>
	B	0.962	0.022	<b>0.956</b>	<u>0.082</u>	0.060
	C	0.996	0.331	<b>0.989</b>	0.163	<u>0.786</u>
	D	0.992	0.038	<b>0.980</b>	0.094	<u>0.732</u>
RAND+FGSM $\epsilon = 0.3, \alpha = 0.05$	A	0.997	0.179	<b>0.988</b>	0.171	<u>0.774</u>
	B	0.962	0.017	<b>0.944</b>	0.091	<u>0.138</u>
	C	0.996	0.103	<b>0.985</b>	0.151	<u>0.907</u>
	D	0.992	0.050	<b>0.980</b>	0.115	<u>0.539</u>
CW $\ell_2$ norm	A	0.997	<u>0.141</u>	<b>0.989</b>	0.038	0.077
	B	0.962	0.032	<b>0.916</b>	0.034	<u>0.280</u>
	C	0.996	<u>0.126</u>	<b>0.989</b>	0.025	0.031
	D	0.992	<u>0.032</u>	<b>0.983</b>	0.021	0.010
Attack	Classifier Model	No Attack	No Defense	Defense-GAN-Rec	MagNet	Adv. Tr. $\epsilon = 0.3$
FGSM $\epsilon = 0.3$	A	0.934	0.102	<b>0.879</b>	0.089	<u>0.797</u>
	B	0.747	0.102	<b>0.629</b>	<u>0.168</u>	0.136
	C	0.933	0.139	<b>0.896</b>	0.110	<u>0.804</u>
	D	0.892	0.082	<b>0.875</b>	0.099	<u>0.698</u>
RAND+FGSM $\epsilon = 0.3, \alpha = 0.05$	A	0.934	0.102	<b>0.888</b>	0.096	<u>0.447</u>
	B	0.747	0.131	<b>0.661</b>	<u>0.161</u>	0.119
	C	0.933	0.105	<b>0.893</b>	0.112	<u>0.699</u>
	D	0.892	0.091	<b>0.862</b>	0.104	<u>0.626</u>
CW $\ell_2$ norm	A	0.934	0.076	<b>0.896</b>	0.060	<u>0.157</u>
	B	0.747	<u>0.172</u>	<b>0.656</b>	0.131	<u>0.118</u>
	C	0.933	0.063	<b>0.896</b>	0.084	<u>0.107</u>
	D	0.892	0.090	<b>0.875</b>	0.069	<u>0.149</u>



Table 5.6: Neural network architectures used for GANs.

Generator	Discriminator
FC(4096)	Conv(64, $5 \times 5$ , 2)
ReLU	LeakyReLU(0.2)
ConvT(256, $5 \times 5$ , 1)	Conv(128, $5 \times 5$ , 2)
ReLU	LeakyReLU(0.2)
ConvT(128, $5 \times 5$ , 1)	Conv(256, $5 \times 5$ , 2)
ReLU	LeakyReLU(0.2)
ConvT(1, $5 \times 5$ , 1)	FC(1)
Sigmoid	Sigmoid

Table 5.7: Neural network architecture used for the MagNet encoder.

Encoder
Conv(64, $5 \times 5$ , 2)
LeakyReLU(0.2)
Conv(128, $5 \times 5$ , 2)
LeakyReLU(0.2)
Conv(256, $5 \times 5$ , 2)
LeakyReLU(0.2)
FC(128) + tanh

Table 5.9: Classification accuracy of Model F using Defense-GAN with various number of iterations  $L$  ( $R = 10$ ), on the F-MNIST dataset, under FGSM black-box attack with  $\epsilon = 0.3$ .

$L$	Defense-GAN-Rec No attack	Defense-GAN-Orig No attack	Defense-GAN-Rec Adversarial	Defense-GAN-Orig Adversarial
25	$0.8037 \pm 0.0050$	$0.7595 \pm 0.0009$	$0.4040 \pm 0.0149$	$0.3910 \pm 0.0119$
50	$0.8676 \pm 0.0018$	$0.7898 \pm 0.0016$	$0.4412 \pm 0.0023$	$0.3980 \pm 0.0114$
100	$0.9101 \pm 0.0032$	$0.8190 \pm 0.0043$	$0.4808 \pm 0.0088$	$0.4221 \pm 0.0255$
200	$0.9145 \pm 0.0014$	$0.8373 \pm 0.0054$	$0.5119 \pm 0.0038$	$0.4594 \pm 0.0056$
400	$0.9490 \pm 0.0013$	$0.8557 \pm 0.0049$	$0.5126 \pm 0.0096$	$0.4754 \pm 0.0102$
800	$0.9588 \pm 0.0065$	$0.8832 \pm 0.0042$	$0.5520 \pm 0.0098$	$0.4644 \pm 0.0092$
1600	$0.9640 \pm 0.0010$	$0.9125 \pm 0.0040$	$0.5335 \pm 0.0226$	$0.4952 \pm 0.0155$

Table 5.10: Classification accuracy of Model F using Defense-GAN with various number of random restarts  $R$  ( $L = 100$ ), on the MNIST dataset, under FGSM black-box attack with  $\epsilon = 0.3$ .

$R$	Defense-GAN-Rec No attack	Defense-GAN-Orig No attack	Defense-GAN-Rec Adversarial	Defense-GAN-Orig Adversarial
1	$0.7035 \pm 0.0035$	$0.6436 \pm 0.0017$	$0.5329 \pm 0.0094$	$0.5011 \pm 0.0085$
2	$0.8619 \pm 0.0010$	$0.8080 \pm 0.0029$	$0.6722 \pm 0.0041$	$0.6605 \pm 0.0050$
5	$0.9523 \pm 0.0006$	$0.9213 \pm 0.0024$	$0.8199 \pm 0.0097$	$0.8228 \pm 0.0038$
10	$0.9810 \pm 0.0015$	$0.9560 \pm 0.0051$	$0.8956 \pm 0.0032$	$0.8911 \pm 0.0024$
20	$0.9966 \pm 0.0009$	$0.9753 \pm 0.0010$	$0.9456 \pm 0.0031$	$0.9310 \pm 0.0023$

Table 5.12: Classification accuracies of different classifier models using various defense strategies on the CelebA gender classification task, under FGSM, RAND+FGSM, and CW white-box attacks. Defense-GAN has  $L = 200$  and  $R = 2$ .

Attack	Classifier Model	No Attack	No Defense	Defense- GAN-Rec	MagNet	Adv. Tr. $\epsilon = 0.3$
FGSM $\epsilon = 0.3$	A	0.9652	0.0870	<b>0.9255</b>	0.0985	<u>0.1225</u>
	B	0.9468	0.0995	<b>0.9140</b>	0.0920	<u>0.2345</u>
	C	0.9459	0.0460	<b>0.9255</b>	0.1085	<u>0.1130</u>
	D	0.9476	0.0605	<b>0.9205</b>	0.0975	<u>0.7755</u>
RAND+FGSM $\epsilon = 0.3, \alpha = 0.05$	A	0.9652	0.0560	<b>0.9280</b>	<u>0.1105</u>	0.0700
	B	0.9468	0.1785	<b>0.9030</b>	0.1015	<u>0.4515</u>
	C	0.9459	0.0470	<b>0.9200</b>	0.1045	<u>0.1055</u>
	D	0.9476	0.0665	<b>0.9165</b>	0.1105	<u>0.696</u>
CW $\ell_2$ norm	A	0.9652	0.0460	<b>0.8210</b>	0.0985	<u>0.5690</u>
	B	0.9468	0.0575	<b>0.7465</b>	<u>0.0955</u>	0.0725
	C	0.9459	0.0435	<b>0.7985</b>	0.0985	<u>0.2635</u>
	D	0.9476	0.0660	<b>0.7740</b>	0.1040	<u>0.5010</u>

## Chapter 6: Summary and Future Directions

### 6.1 Summaries

In Chapter 2, we presented a continuous face-based authentication method using facial attributes for mobile devices. We trained binary attribute classifiers and showed their effectiveness as feature vectors for active authentication with extensive experiments. We showed that attribute-based scores alone could improve the verification results. Furthermore, we showed that in situations where the low-level features such as LBP are reliable, verification results could be further improved by fusing the resulting scores with the attribute-based scores. We also evaluated the different realizations of our method on an actual cell phone and showed that the authentication algorithm could be implemented with low memory usage, power consumption at more than four frames per second. We also proposed a feasible multi-task DCNN architecture to extract accurate describable facial attributes on mobile devices. Each network predicted multi facial attributes from a given face component by mapping it to a shared embedding space. We showed that our attribute prediction performance is comparable to state-of-the-art. We explored the embedding space and illustrated that we could extract new attributes by looking at subspace clusters of this space. We also have shown that our networks perform

attribute-based authentication better than the previously proposed method. Finally, we analyze the feasibility of our method by performing battery usage and prediction speed experiments on an actual mobile device.

In Chapter 3, we presented a new attribute manipulation model that offers unprecedented flexibility. Unlike previous models that represent attributes as discrete classes, our multi-dimensional, continuous attribute representation allows for a much richer set of attribute manipulations. These include Diverse Swaps, in which an attribute can be changed but realized via a diverse set of choices, and Borrows, in which an attribute realization can be taken from a reference image and applied to a query image. Qualitative evaluation of our approach illustrates its efficacy for use in a variety of applications.

In Chapter 4, we introduced ExplainGAN to interpret black-box classifiers by visualizing boundary-crossing transformations. These transformations are designed to be interpretable by humans and provide a high-level, conceptual intuition underlying a classifier’s decisions. This style of visualization can overcome limitations of attribution and example-by-nearest-neighbor methods by making spatially localized changes along with visual examples. While not explicitly trained to act as a saliency map, ExplainGAN’s maps are very competitive at demonstrating saliency. We also introduced a new metric, Substitutability, that evaluates how much label-capturing information is retained when performing boundary-crossing image transformations.

In Chapter 5, we proposed Defense-GAN, a novel defense strategy utilizing GANs to enhance the robustness of classification models against black-box and white-box adversarial attacks. Our method did not assume a particular attack

model and was shown to be effective against most commonly considered attack strategies. We empirically show that Defense-GAN consistently provides adequate defense on two benchmark computer vision datasets, whereas other methods had many shortcomings on at least one type of attack.

It is worth mentioning that, although Defense-GAN was shown to be a possible defense mechanism against adversarial attacks, one might come across practical difficulties while implementing and deploying this method. The success of Defense-GAN relies on the expressiveness and generative power of the GAN. However, training GANs is still a challenging task and an active area of research, and if the GAN is not adequately trained and tuned, the performance of Defense-GAN will suffer on both original and adversarial examples. Moreover, the choice of hyper-parameters  $L$  and  $R$  is also critical to the effectiveness of the defense, and it may be challenging to tune them without knowing the attack model.

## 6.2 Future Directions

As seen in Chapter 2, attributes are potent features for designing real-world ML systems efficiently and interactively. We have shown that raw attribute scores are useful for active authentication; however, a better way of coming up with the score is possible. Although we have presented a sampling scheme to see rare labels more frequently, it is worthwhile to see how active sampling methods can be used to sample data more efficiently. For example, if the network is performing “well” for an attribute, it might be better to sample data points in a way that is “beneficial”

for training the other attributes.

Besides training the attribute classifier, given that a single CNN can be trained for multiple tasks, an extension of our work may include loss functions for face verification. More specifically, we can have a network that is trained with metric learning loss or identity classification loss to get both attribute predictions and identity-based similarity scores. This approach, however, requires a dataset labeled with attributes and identities. In the case of not having such a dataset but having multiple datasets with identity labels and multiple datasets with attribute labels, it will be interesting to see how one can train different parts of the network with the label at hand and not train for losses that we do not have labels for.

Another aspect of using attributes for active authentication is the phenomena of domain shift. The reason is that the attributes are learned from datasets that are usually gathered from the web, but the models are tested on mobile phone images. These two domains are different in many ways. First of all, the images that are acquired by the front camera of a cell phone are usually of low quality compared to the ones that are available online. The phone cameras have lens distortion which matters because we tend to hold the phone close to our face. The illumination is different since usually, the cell phone images are acquired with indoor lighting or low light, but the web images are usually outdoors, or if captured indoor, they are taken in well lit conditions. All of these domain differences can be either dealt with using data-driven domain adaptation methods.

In Chapter 3 we looked at the problem of conditional image generation. The model that was used in this chapter had an encoder-decoder structure. As a future

research direction, it will be interesting to see how we can extend the idea to just a single generator. Conditional GANs have been around and are working with binary attribute labels. One possible way of exploring this is to use an alternative optimization method with one step reconstructing a given image and constraining the latent codes, and the other step for minimax optimization.

In Chapter 4 we filled the semantic gap with attribute detectors. The proposed method works for binary classifiers; thus the next step is to extend the work for multiple classes. The primary issue in explaining a multi-class classifier is the quadratic number of pairwise relationships between classes. More specifically in an  $n$  class problem, having an input of one class, we want to get  $n - 1$  transformations and explanations for the same input. One way would be to embed the solution into the network architecture. For example, one can put  $n$  transformation heads and mask heads. Then for a given input of class  $A$ , all other heads will explain the transfer from one class to the other. Several issues arise in this case. First of all, the complexity of the network goes up linearly as  $n$  increases. This results in inefficient training and inference speed. The second issue is that each head will be trained with a subset of the data for that specific class, which means that we need a large and balanced dataset regarding the labels.

Besides dataset size and label imbalance, there is a conceptual issue with considering all of the pairwise relationships. It might not be obvious apriori that all the pairwise relationships are meaningful. Therefore approaches that consider all of them together might fail at training time for this reason. One possible way to overcome this issue is to selectively train different parts of the network for the mean-

ingful relationships. However, this approach will still have the same computation complexity. Another way would be to add target labels as inputs to the network and keep the same network architecture. This way the network complexity will stay almost the same, and we can efficiently train the network for meaningful class pairs.

In Chapter 5 we looked at how to protect attribute detectors from induced semantic gap caused by adversarial perturbations using GANs. The primary challenge with the current approach is the inference speed. This issue can be addressed by approximating the latent codes with a feedforward CNN and continuing the gradient descent steps from there.

Another future research direction for Chapter 5 is attacking a classifier that is protected by the proposed method. As mentioned in Section 5.6, gradient-based attacks, which are the dominant type of attacks, cannot work for the proposed framework. One possible way is to attack each step of the unrolled gradient descent steps. Such approaches have been successful for attacking RNNs [Papernot et al., 2016d] and can possibly be extended to Defense-GAN.



## Bibliography

- [Abadi et al., 2015a] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015a). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.
- [Abadi et al., 2015b] Abadi, M. et al. (2015b). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.
- [Ahonen et al., 2006] Ahonen, T., Hadid, A., and Pietikainen, M. (2006). Face description with local binary patterns: Application to face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(12):2037–2041.
- [Altinok and Turk, 2003] Altinok, A. and Turk, M. (2003). Temporal integration for continuous multimodal biometrics. In *Proceedings of the Workshop on Multimodal User Authentication*. Citeseer.
- [Antipov et al., 2017] Antipov, G., Baccouche, M., and Dugelay, J.-L. (2017). Face aging with conditional generative adversarial networks. *arXiv preprint arXiv:1702.01983*.
- [Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.
- [Asthana et al., 2013] Asthana, A., Zafeiriou, S., Cheng, S., and Pantic, M. (2013). Robust discriminative response map fitting with constrained local models. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3444–3451. IEEE.

- [Athalye et al., 2018] Athalye, A., Carlini, N., and Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*.
- [Bach et al., 2015] Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140.
- [Berg and Belhumeur, 2013] Berg, T. and Belhumeur, P. N. (2013). Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 955–962. IEEE.
- [Bourdev et al., 2011] Bourdev, L., Maji, S., and Malik, J. (2011). Describing people: A poselet-based approach to attribute classification. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1543–1550. IEEE.
- [Bousmalis et al., 2017] Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., and Krishnan, D. (2017). Unsupervised pixel-level domain adaptation with generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 7.
- [Bradski, 2000] Bradski, G. (2000). Opencv toolbox. *Dr. Dobbs’s Journal of Software Tools*.
- [Brock et al., 2016] Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2016). Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093*.
- [Carlini and Wagner, 2017] Carlini, N. and Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE.
- [Carrillo, 2003] Carrillo, C. M. (2003). Continuous biometric authentication for authorized aircraft personnel: A proposed design. Technical report, DTIC Document.
- [Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.
- [Chen et al., 2016] Chen, X., Duan, Y., Houthoofd, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180.
- [Choi et al., 2017] Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. (2017). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *arXiv preprint arXiv:1711.09020*.

- [Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.
- [Cox and Pinto, 2011] Cox, D. and Pinto, N. (2011). Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 8–15. IEEE.
- [Crouse et al., 2015] Crouse, D., Han, H., Chandra, D., Barbellio, B., and Jain, A. K. (2015). Continuous authentication of mobile user: Fusion of face image and inertial measurement unit data. In *International Conference on Biometrics*.
- [Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.
- [Datta et al., 2005] Datta, R., Li, J., and Wang, J. Z. (2005). Content-based image retrieval: approaches and trends of the new age. In *Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval*, pages 253–262. ACM.
- [Derawi et al., 2010] Derawi, M., Nickel, C., Bours, P., and Busch, C. (2010). Unobtrusive user-authentication on mobile phones using biometric gait recognition. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 306–311.
- [Elhamifar and Vidal, 2009] Elhamifar, E. and Vidal, R. (2009). Sparse subspace clustering. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2790–2797. IEEE.
- [Everingham et al., 2008] Everingham, M., Van Gool, L., Williams, C., Winn, J., and Zisserman, A. (2008). The pascal visual object classes challenge 2008.
- [Farhadi et al., 2009] Farhadi, A., Endres, I., Hoiem, D., and Forsyth, D. (2009). Describing objects by their attributes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1778–1785. IEEE.
- [Fathy et al., 2015] Fathy, M. E., Patel, V. M., and Chellappa, R. (2015). Face-based active authentication on mobile devices. In *IEEE International Conference on Acoustics, Speech and Signal Processing*.
- [Feng et al., 2012] Feng, T., Liu, Z., Kwon, K.-A., Shi, W., Carburnar, B., Jiang, Y., and Nguyen, N. (2012). Continuous mobile authentication using touchscreen gestures. In *IEEE Conference on Technologies for Homeland Security*, pages 451–456.
- [Ferrari and Zisserman, 2007] Ferrari, V. and Zisserman, A. (2007). Learning visual attributes. In *Advances in Neural Information Processing Systems*, pages 433–440.

- [Fong and Vedaldi, 2017] Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. *arXiv preprint arXiv:1704.03296*.
- [Frank et al., 2013] Frank, M., Biedert, R., Ma, E., Martinovic, I., and Song, D. (2013). Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Transactions on Information Forensics and Security*, 8(1):136–148.
- [Fridman et al., 2015] Fridman, L., Weber, S., Greenstadt, R., and Kam, M. (2015). Active authentication on mobile devices via stylometry, gps location, web browsing behavior, and application usage patterns. *IEEE Systems Journal*.
- [Ganin et al., 2016] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.
- [Gatys et al., 2015] Gatys, L. A., Ecker, A. S., and Bethge, M. (2015). A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*.
- [Gatys et al., 2016] Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). Image style transfer using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2414–2423. IEEE.
- [Goodfellow et al., 2014a] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.
- [Goodfellow et al., 2014b] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- [Goodman and Flaxman, 2016] Goodman, B. and Flaxman, S. (2016). European union regulations on algorithmic decision-making and a” right to explanation”. *arXiv preprint arXiv:1606.08813*.
- [Gulrajani et al., 2017] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*.
- [Gunther et al., 2013] Gunther, M., Costa-Pazo, A., Ding, C., Boutellaa, E., Chiacchia, G., Zhang, H., de Assis Angeloni, M., Struc, V., Khoury, E., Vazquez-Fernandez, E., et al. (2013). The 2013 face recognition evaluation in mobile environment. In *Biometrics (ICB), 2013 International Conference on*, pages 1–7. IEEE.

- [Hadid et al., 2007] Hadid, A., Heikkila, J., Silven, O., and Pietikainen, M. (2007). Face and eye detection for person authentication in mobile phones. In *ACM/IEEE International Conference on Distributed Smart Cameras*, pages 101–108.
- [Hendrycks and Gimpel, 2017] Hendrycks, D. and Gimpel, K. (2017). Early methods for detecting adversarial images.
- [Hinton et al., 2015] Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- [Huang et al., 2007] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. (2007). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst.
- [Inc, 2013] Inc, N. M. (2013). Nearly one in three consumers who have lost their mobile devices still do not lock them, new survey shows. ”<http://www.prnewswire.com/news-releases/nearly-one-in-three-consumers-who-have-lost-their-mobile-devices-still-do-not-lock-them-new-survey-shows-200410151.html>”.
- [Isola et al., 2017] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *arXiv preprint*.
- [Jain et al., 2004] Jain, A. K., Dass, S. C., and Nandakumar, K. (2004). Soft biometric traits for personal recognition systems. In *Biometric Authentication*, pages 731–738. Springer.
- [Janakiraman et al., 2005] Janakiraman, R., Kumar, S., Zhang, S., and Sim, T. (2005). Using continuous face verification to improve desktop security. In *Application of Computer Vision, 2005. WACV/MOTIONS’05 Volume 1. Seventh IEEE Workshops on*, volume 1, pages 501–507. IEEE.
- [Johnson et al., 2016] Johnson, J., Alahi, A., and Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer.
- [Kabkab et al., 2018] Kabkab, M., Samangouei, P., and Chellappa, R. (2018). Task-aware compressed sensing with generative adversarial networks. *arXiv preprint arXiv:1802.01284*.
- [Kazemi and Sullivan, 2014] Kazemi, V. and Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1867–1874. IEEE.
- [King, 2009] King, D. E. (2009). Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758.

- [Kingma and Ba, 2014] Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [Klare et al., 2014] Klare, B. F., Klum, S., Klontz, J. C., Taborsky, E., Akgul, T., and Jain, A. K. (2014). Suspect identification based on descriptive facial attributes. In *Biometrics (IJCB), 2014 IEEE International Joint Conference on*, pages 1–8. IEEE.
- [Klosterman and Ganger, 2000] Klosterman, A. J. and Ganger, G. R. (2000). Secure continuous biometric-enhanced authentication.
- [Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- [Kulkarni et al., 2015] Kulkarni, T. D., Whitney, W. F., Kohli, P., and Tenenbaum, J. (2015). Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547.
- [Kumar et al., 2008] Kumar, N., Belhumeur, P. N., and Nayar, S. K. (2008). Face-Tracer: A Search Engine for Large Collections of Images with Faces. In *European Conference on Computer Vision (ECCV)*, pages 340–353.
- [Kumar et al., 2009] Kumar, N., Berg, A. C., Belhumeur, P. N., and Nayar, S. K. (2009). Attribute and Simile Classifiers for Face Verification. In *IEEE International Conference on Computer Vision (ICCV)*.
- [Kurakin et al., 2016] Kurakin, A., Goodfellow, I., and Bengio, S. (2016). Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.
- [Lampert et al., 2014] Lampert, C. H., Nickisch, H., and Harmeling, S. (2014). Attribute-based classification for zero-shot visual object categorization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(3):453–465.
- [Lample et al., 2017] Lample, G., Zeghidour, N., Usunier, N., Bordes, A., Denoyer, L., et al. (2017). Fader networks: Manipulating images by sliding attributes. In *Advances in Neural Information Processing Systems*, pages 5969–5978.
- [Larsen et al., 2015] Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*.
- [Layne et al., 2012] Layne, R., Hospedales, T. M., Gong, S., and Mary, Q. (2012). Person re-identification by attributes. In *BMVC*, volume 2, page 8.
- [LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

- [Levi and Hassner, 2015] Levi, G. and Hassner, T. (2015). Age and gender classification using convolutional neural networks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) workshops*.
- [Liu et al., 2011] Liu, J., Kuipers, B., and Savarese, S. (2011). Recognizing human actions by attributes. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3337–3344. IEEE.
- [Liu et al., 2014] Liu, M., Zhang, D., and Chen, S. (2014). Attribute relation learning for zero-shot classification. *Neurocomputing*, 139:34–46.
- [Liu et al., 2017] Liu, M.-Y., Breuel, T., and Kautz, J. (2017). Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708.
- [Liu and Tuzel, 2016] Liu, M.-Y. and Tuzel, O. (2016). Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477.
- [Liu et al., 2016] Liu, Y., Chen, X., Liu, C., and Song, D. (2016). Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*.
- [Liu et al., 2007] Liu, Y., Zhang, D., Lu, G., and Ma, W.-Y. (2007). A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282.
- [Liu et al., 2015] Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.
- [Maaten and Hinton, 2008] Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.
- [Mahbub et al., 2016] Mahbub, U., Patel, V. M., Chandra, D., Barbello, B., and Chellappa, R. (2016). Partial face detection for continuous authentication. *arXiv preprint arXiv:1603.09364*.
- [Mathieu et al., 2016] Mathieu, M. F., Zhao, J. J., Zhao, J., Ramesh, A., Sprechmann, P., and LeCun, Y. (2016). Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5040–5048.
- [MATLAB, 2014] MATLAB (2014). *version 8.4.0.150421 (R2014b)*. The Math-Works Inc., Natick, Massachusetts.
- [McCool et al., 2012] McCool, C., Marcel, S., Hadid, A., Pietikainen, M., Matejka, P., Cernocky, J., Poh, N., Kittler, J., Larcher, A., Levy, C., Matrouf, D., Bonastre, J.-F., Tresadern, P., and Cootes, T. (2012). Bi-modal person recognition on

- a mobile phone: using mobile phone data. In *IEEE ICME Workshop on Hot Topics in Mobile Multimedia*.
- [Meng and Chen, 2017] Meng, D. and Chen, H. (2017). Magnet: a two-pronged defense against adversarial examples. *arXiv preprint arXiv:1705.09064*.
- [Monrose et al., 2002] Monrose, F., Reiter, M. K., and Wetzels, S. (2002). Password hardening based on keystroke dynamics. *International Journal of Information Security*, 1(2):69–83.
- [Montavon et al., 2017] Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K.-R. (2017). Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 65:211–222.
- [Moosavi-Dezfooli et al., 2016] Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582.
- [Niinuma and Jain, 2010] Niinuma, K. and Jain, A. K. (2010). Continuous user authentication using temporal information. In *SPIE Defense, Security, and Sensing*, pages 76670L–76670L. International Society for Optics and Photonics.
- [Niinuma et al., 2010] Niinuma, K., Park, U., and Jain, A. K. (2010). Soft biometric traits for continuous user authentication. *Information Forensics and Security, IEEE Transactions on*, 5(4):771–780.
- [Obeid et al., 2001] Obeid, M., Jedynek, B., and Daoudi, M. (2001). Image indexing & retrieval using intermediate features. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 531–533. ACM.
- [Papernot et al., 2016a] Papernot, N., Goodfellow, I., Sheatsley, R., Feinman, R., and McDaniel, P. (2016a). cleverhans v1. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*.
- [Papernot et al., 2017] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. (2017). Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM.
- [Papernot et al., 2016b] Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016b). The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE.
- [Papernot et al., 2016c] Papernot, N., McDaniel, P., Sinha, A., and Wellman, M. (2016c). Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*.



- [Papernot et al., 2016d] Papernot, N., McDaniel, P., Swami, A., and Harang, R. (2016d). Crafting adversarial input sequences for recurrent neural networks. In *Military Communications Conference, MILCOM 2016-2016 IEEE*, pages 49–54. IEEE.
- [Papernot et al., 2016e] Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. (2016e). Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 582–597. IEEE.
- [Parkhi et al., 2015] Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2015). Deep face recognition. In *British Machine Vision Conference*.
- [Patterson and Hays, 2012] Patterson, G. and Hays, J. (2012). Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2751–2758. IEEE.
- [Perarnau et al., 2016] Perarnau, G., van de Weijer, J., Raducanu, B., and Álvarez, J. M. (2016). Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*.
- [Prechelt, 1998] Prechelt, L. (1998). Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks*, 11(4):761–767.
- [Primo et al., 2014] Primo, A., Phoha, V., Kumar, R., and Serwadda, A. (2014). Context-aware active authentication using smartphone accelerometer measurements. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 98–105.
- [Radford et al., 2015] Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.
- [Rudin et al., 1992] Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268.
- [Samangouei and Chellappa, 2016] Samangouei, P. and Chellappa, R. (2016). Convolutional neural networks for attribute-based active authentication on mobile devices. In *Biometrics Theory, Applications and Systems (BTAS), 2016 IEEE 8th International Conference on*, pages 1–8. IEEE.
- [Samangouei et al., 2017a] Samangouei, P., Hand, E., Patel, V. M., and Chellappa, R. (2017a). Active authentication using facial attributes. *Mobile Biometrics*, 3:131.

- [Samangouei et al., 2018a] Samangouei, P., Kabkab, M., and Chellappa, R. (2018a). Defense-gan: Protecting classifiers against adversarial attacks using generative models. *International Conference on Learning Representations*.
- [Samangouei et al., 2015] Samangouei, P., Patel, V. M., and Chellappa, R. (2015). Attribute-based continuous user authentication on mobile devices. In *Biometrics Theory, Applications and Systems (BTAS), 2015 IEEE 7th International Conference on*, pages 1–8.
- [Samangouei et al., 2017b] Samangouei, P., Patel, V. M., and Chellappa, R. (2017b). Facial attributes for active authentication on mobile devices. *Image and Vision Computing*, 58:181–192.
- [Samangouei et al., 2018b] Samangouei, P., Saeidi, A., Nakagiwa, L., and Silberman, N. (2018b). Explaingan: Model explanation via decision boundary crossing transformations. *Proceedings of European Conference on Computer Vision (ECCV)*.
- [Samek et al., 2017] Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. (2017). Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673.
- [Sarkar et al., 2016] Sarkar, S., Patel, V. M., and Chellappa, R. (2016). Deep feature-based face detection on mobile devices. *arXiv preprint arXiv:1602.04868*.
- [Schmidhuber, 1992] Schmidhuber, J. (1992). Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879.
- [Schroff et al., 2015] Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823.
- [Selvaraju et al., 2016a] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2016a). Grad-cam: Visual explanations from deep networks via gradient-based localization. See <https://arxiv.org/abs/1610.02391v3>, 7(8).
- [Selvaraju et al., 2016b] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2016b). Grad-cam: Visual explanations from deep networks via gradient-based localization. See <https://arxiv.org/abs/1610.02391v3>, 7(8).
- [Shrikumar et al., 2017] Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685*.

- [Shrikumar et al., 2016] Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. (2016). Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*.
- [Sim et al., 2007] Sim, T., Zhang, S., Janakiraman, R., and Kumar, S. (2007). Continuous verification using multimodal biometrics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(4):687–700.
- [Simonyan et al., 2013] Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: visualising image classification models and saliency maps (2014). *arXiv preprint arXiv:1312.6034*.
- [Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- [Spillane, 1975] Spillane, R. (1975). Keyboard apparatus for personal identification. *IBM Technical Disclosure Bulletin*, 17(3346):3346.
- [Springenberg et al., 2014] Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.
- [Sun et al., 2014a] Sun, Y., Chen, Y., Wang, X., and Tang, X. (2014a). Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996.
- [Sun et al., 2014b] Sun, Y., Chen, Y., Wang, X., and Tang, X. (2014b). Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996.
- [Sundararajan et al., 2017] Sundararajan, M., Taly, A., and Yan, Q. (2017). Ax-  
iomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*.
- [Szegedy et al., 2013] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.
- [Taigman et al., 2014] Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). 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.
- [Takacs et al., 2008] Takacs, G., Chandrasekhar, V., Gelfand, N., Xiong, Y., Chen, W.-C., Bimpigiannis, T., Grzeszczuk, R., Pulli, K., and Girod, B. (2008). Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, pages 427–434. ACM.

- [Tramèr et al., 2017] Tramèr, F., Kurakin, A., Papernot, N., Boneh, D., and McDaniel, P. (2017). Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.
- [Tropp and Gilbert, 2007] Tropp, J. A. and Gilbert, A. C. (2007). Signal recovery from random measurements via orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 53(12):4655–4666.
- [Vaquero et al., 2009] Vaquero, D., Feris, R. S., Tran, D., Brown, L., Hampapur, A., Turk, M., et al. (2009). Attribute-based people search in surveillance environments. In *Applications of Computer Vision (WACV), 2009 Workshop on*, pages 1–8. IEEE.
- [Vedaldi and Fulkerson, 2008] Vedaldi, A. and Fulkerson, B. (2008). VLFeat: An open and portable library of computer vision algorithms.
- [Viola et al., 2005] Viola, P., Jones, M. J., and Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161.
- [Wattenberg et al., 2016] Wattenberg, M., Viñe, F., and Johnson, I. (2016). How to use t-sne effectively. *Distill*.
- [Wen et al., 2016] Wen, Y., Zhang, K., Li, Z., and Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer.
- [Wiskott et al., 1997] Wiskott, L., Fellous, J.-M., Kuiger, N., and Von Der Malsburg, C. (1997). Face recognition by elastic bunch graph matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):775–779.
- [Wright et al., 2009] Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., and Ma, Y. (2009). Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227.
- [Xiao et al., 2017a] Xiao, H., Rasul, K., and Vollgraf, R. (2017a). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.
- [Xiao et al., 2017b] Xiao, H., Rasul, K., and Vollgraf, R. (2017b). Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.
- [Yan et al., 2016] Yan, X., Yang, J., Sohn, K., and Lee, H. (2016). Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, pages 776–791. Springer.
- [Zeiler and Fergus, 2014] Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

- [Zhang et al., 2015a] Zhang, H., Patel, V. M., and Chellappa, R. (2015a). Robust multimodal recognition via multitask multivariate low-rank representations. In *IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE.
- [Zhang et al., 2015b] Zhang, H., Patel, V. M., Fathy, M. E., and Chellappa, R. (2015b). Touch gesture-based active user authentication using dictionaries. In *IEEE Winter conference on Applications of Computer Vision*. IEEE.
- [Zhang et al., 2015c] Zhang, H., Patel, V. M., Shekhar, S., and Chellappa, R. (2015c). Domain adaptive sparse representation-based classification. In *IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE.
- [Zhang et al., 2014a] Zhang, L., Kalashnikov, D. V., and Mehrotra, S. (2014a). Context-assisted face clustering framework with human-in-the-loop. *International Journal of Multimedia Information Retrieval*, 3(2):69–88.
- [Zhang et al., 2010] Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R. P., Mao, Z. M., and Yang, L. (2010). Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pages 105–114. ACM.
- [Zhang et al., 2014b] Zhang, N., Paluri, M., Ranzato, M., Darrell, T., and Bourdev, L. (2014b). Panda: Pose aligned networks for deep attribute modeling. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1637–1644. IEEE.
- [Zhang et al., 2005] Zhang, W., Shan, S., Gao, W., Chen, X., and Zhang, H. (2005). Local gabor binary pattern histogram sequence (lgbphs): A novel non-statistical model for face representation and recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 786–791. IEEE.
- [Zhao et al., 1998] Zhao, W., Krishnaswamy, A., Chellappa, R., Swets, D. L., and Weng, J. (1998). Discriminant analysis of principal components for face recognition. In *Face Recognition*, pages 73–85. Springer.
- [Zhou et al., 2016] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929.
- [Zhu et al., 2017] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*.