SYNTHETIC FINGERPRINT GENERATION WITH GANS

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by
Vahdettin Onur Kılınç

December 2021
İZMİR

ACKNOWLEDGMENTS

First and foremost, I am eternally grateful for the support of my supervisor *Nesli ERDOĞMUŞ*. Without her guidance and involvement in every step of the process, I could never accomplish this. I am thankful for your understanding, patience, and support.

I want to thank my family, especially my mom, for always being there for me. Like everything I have achieved in life, I owe finishing this thesis to their unconditional love and support.

Finally, I would like to thank my professors and colleagues for helping me and being understanding over the years.

ABSTRACT

SYNTHETIC FINGERPRINT GENERATION WITH GANS

Fingerprints are regarded as the most reliable form of human identification for thousands of years. Even though the fingerprint acquisiton process has became more convenient with technological advancements, privacy concerns hindered the data collection and, hence advancement of research on fingerprint biometrics. Like many other problem solved with deep learning, biometrics also requires a sizable database to succeed. This study focuses on synthetic fingerprint generation to tackle bottlenecks created by data scarcity. First, a preprocessing pipeline is designed tp enhance images from a small publicly available fingerprint dataset. Next the new enhanced dataset is given as an input to a generative network to create candidate synthetic fingerprints. Lastly Fingerprint image quality models filter low-quality fingerprint images from the candidate set to form the synthetic fingerprint dataset.

Numerous experiments were conducted to show the usability of the generated synthetics fingerprints using both real and synthetic fingerprint datasets available for network trainig. Experimental results show that enhancing fingerprint images from real-life datasets helps models trained with synthetic fingerprint images classify enhanced versions of the real-life fingerprint samples.Synthetic fingerprints generated using the proposed pipeline can establish a good training set which can imporove deep neural network performance as substantially as their real- life counterparts, but without introducing any privacy concerns.

ÖZET

SENTETİK PARMAK İZLERİNİN GANLAR İLE ÜRETİMİ

Parmak izleri binlerce yıldır en güvenilir insan tanıma yöntemi olarak kabul edilmiştir. Teknolojik gelişmeler ile parmak izi toplama süreci daha pratik hale gelmiş olsa da; kişilerin gizlilik endişeleri parmak izi biyometrisi üzerinde çalışan araştırmacıları yavaşlattı. Derin öğrenme ile çözülen bütün problemler gibi biyometri de başarılı olabilmek için büyük veritabanlarına ihtiyaç duyar. Kişilerin gizliliği ile oluştulan yasaların getirdiği darboğaz problemini çözmek için bu çalışma sentetik parmak izleri üretmektedir. Tasarlanan üretim hattı, öncelikle halka açık küçük bir parmak izi veri kümesindeki resimleri iyileştirir. Yeni oluşturulan gelişmis parmak izi veri kümesi, aday sentetik parmak izleri üretmek için bir ağa girdi olarak verilir. Parmak izi kalitesi ölçen modeller aday kümeden yüksek kaliteli parmak izlerini, sentetik parmak izi veri kümesi oluşturmak için seçerler.

Bir çok sayıda yapılan deney, üretilen sentetik parmak izlerinin kalitesini göstermek için hem gerçek hem de sentetik veri kümeleri kullanılarak yapıldı. Deneyler sonucunda gerçek parmak izi resimlerimlerinin iyileştirilmesi, sentetik parmak izlerinin gerçek hayata entegrasyonunda yardımcı olduğu gözlemlenmiştir. Geliştirilen üretim hattı kullanılarak üretilen sentetik parmak izleri gerçek hayattaki benzerleri ile yakın temsil kabiliyetleri olduğu ve kişilerin gizliliği sorunları içermediği gözlemlenmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

Figure **Page**

LIST OF TABLES

Table **Page**

CHAPTER 1

INTRODUCTION

Biometrics are getting more integrated into our day-to-day lives with technological advancements; thus, the need for quality fingerprint data is increasing. From small tasks like unlocking phones to entering the most secure buildings globally, fingerprint recognition is still the most used biometric recognition type in the biometric security field. Fingerprints are still the most reliable form of biometric data for recognition. Researchers are trying to create more robust algorithms to improve fingerprint recognition with an ever-growing world population and security concerns.

Today, most fingerprint recognition algorithms are being tested with small datasets like NIST SD4[1] and NIST SD14[10]. These datasets have been discontinued due to their lack of documentation. With the discontinuation of the old datasets, researchers cannot compare the new algorithms with the already existing ones. Regardless of the discontinuation, small datasets tend not to generalize well on training. Obtaining large fingerprint datasets is hard for researchers. Corporates and governments have to obey strict privacy-protecting laws, so acquiring existing large datasets becomes impossible for most.

Researchers tried to synthesize fake fingerprints to solve the lack of data issue. Early researchers created mathematical methods to generate synthetic fingerprints.Recent technological advancements paved the way for researchers to use generative deep learning models to create more realistic synthetic fingerprints. Generative models have been proven useful in different problems, like creating human faces to generating realistic-looking fake bedroom images. Using state-of-the-art generative approaches like IWGAN[8] to generate synthetic fingerprints has gained popularity in recent years.

Synthetic fingerprint generation offers a solution to the problems stated above. Synthetic datasets can be as large as the research requires, and they do not contain any actual personal data. Synthetic fingerprints can be created with different parameters, such as age, ethnicity, and fingerprint class.

High-quality synthetic fingerprint datasets can be used with real-life datasets to

solve data scarcity. This thesis aims to generate realistic synthetic fingerprint images to generate a large dataset to help improve fingerprint biometrics. A synthetic fingerprint generation pipeline was designed. This pipeline takes real-life fingerprint dataset images as an input. These images firstly go through an enhancement process. Enhanced images are given to the GAN model designed. Quality assessment is conducted for the generated fingerprint images, and images with higher quality are added to the new dataset. These new synthetic fingerprints are used in fingerprint classification. Even though fingerprint classification can not identify a single person, it can narrow down the search space. Each finger can have different classes. Classification of all ten fingerprints can narrow the search space even more. The synthetic fingerprint generated in this thesis can be used as a training dataset to train these classifiers.

The thesis continues with background information about fingerprints and generative models. Chapter 3 is about related works where both mathematical and deep learning based approaches on generating synthetic fingerprints are explained briefly. Chapter 4 shows the proposed methodology. The chapter starts with deep learning based approaches and their performances on generating synthetic fingerprints. Pipeline's enhancement, generation processes are explained afterward. Chapter 5 presents the experimental results where representative qualities of synthetic and real-life datasets are compared using classification methods.

CHAPTER 2

BACKGROUND

2.1. Fingerprints

Epidermal ridges on our fingertips create a pattern that we call fingerprints. Fingerprints start to form in the embryonic genome activation stage of embryonic development. At the further parts of this stage, volar pads begin to develop[11]. Volar pads have a significant impact on fingerprint generation[11]. Volar pads are swellings of tissues on the surface of hands and feet. Since their growth rate is slower than the rest of the hand, these pads get slowly less distinct. Since other epidermis cells grow faster, volar epidermis cells create small ledges on the skin. These are ledges that form the first ridges on the skin.

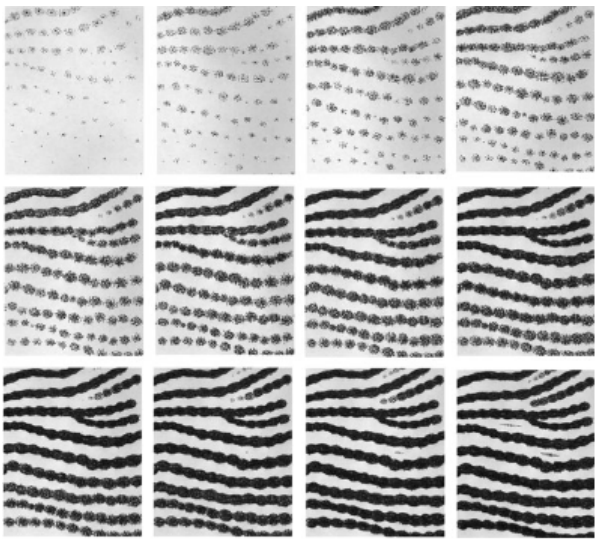


Figure 2.1. Fingerprint ridge formation process.

Most minor changes in the process of the rapid growth changes can alter the whole fingerprints formation. This characteristic makes the fingerprint one of the most reliable

biometric identifiers; thus, it is one of the oldest identifiers humans use. Authorities from ancient China and Babylon signed documents using fingerprints. Starting from the 16th century, scientific studies of fingerprints gained popularity. Mid 17th century scientific studies began to accept the uniqueness of each fingerprint. Henry Faulds and Francis Galton all made efforts to recognize fingerprints as tangible evidence of a crime. After the first arrest using fingerprints as evidence in Argentina[12], all countries started implementing fingerprints into their justice systems. Nowadays, all countries have their fingerprint databases where millions of criminal and civil fingerprints are recorded. Modern methods like using scanners to capture fingerprints helped both official and commercial use of fingerprints. Fast and clean acquisition of fingerprints increased the efficiency and acceptability of the collection process. All databases are digitally stored, and this enables researchers to work on fingerprints on computers, giving them endless opportunities to develop more robust systems on both identification and storage of these fingerprints.

2.2. Fingerprint Details

Each fingerprint is unique, but almost all fingerprints share some characteristics to identify them. All fingerprints consist of some collection of ridges and valleys. Ridges' width and their spacing usually lie between 200µm to 850µm[13]. Details on fingerprints have generally three levels going from Level 1 to Level 3.

2.2.1. Level 1 Details

Ridges are usually straight parallel lines, but they can abruptly end or curve to form some patterns. These pattern regions are called singularities, and these regions make up the Level 1 details. One central region these patterns can develop is the core of the fingerprint. In this region, they tend to create loops, whorls, and arches, and these three types are called the main classes of fingerprints. But in literature, loops and arches are separated into smaller classes to form the major five classes we use today[14]. Loops are divided into left and right loops depending on their overall slant. Arches are separated

into two classes called arches and tented arches, where normal arches tend to consist of continuous ridges while tented arches look like disconnected ridges in the middle with a sharper edge resembling a tent image. These five classes can be seen in Figure 2.2.

(a) Arch (b) Tented Arch (c) Right Loop (d) Left Loop

(e) Whorl

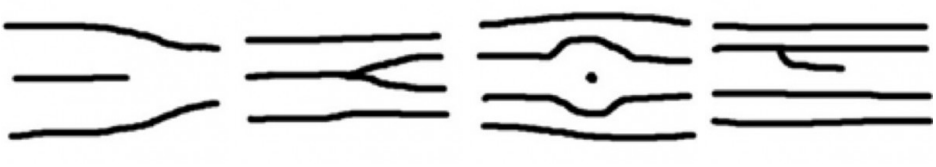
Figure 2.2. Five main fingerprint classes from NIST SD4[1] .

2.2.2. Level 2 Details

Level 2 details, more known as minutiae points, are abrupt ridge endings and points where two ridges combine into one; these minutiae points are called ridge endings and bifurcations, respectively. Example minutiae points can be seen in 2.3

2.3. Master Fingerprints

The master fingerprint is the perfect impression of the fingerprint. Human skin is oily and prone to getting dirty, especially on the fingertips. People cant place their fingers in the same way they did before, and they also cant put the same amount of pressure on the paper or the scanner. Human skin has an elastic structure, and slight pressure changes can distort the fingerprint image. Ridges can look thicker than they are, or they can overlap

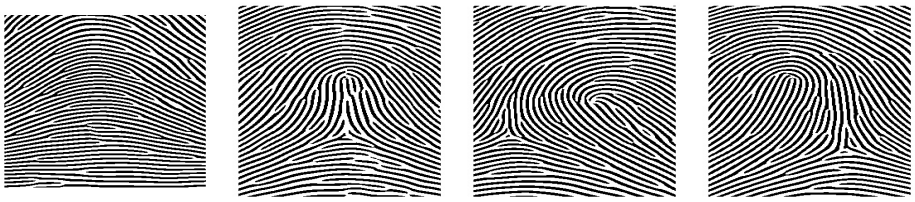


(a) Ridge ending (b) Bifurcation (c) Dot (d) Hook

(e) Enclosure

Figure 2.3. Minutiae point examples.

each other. Fingerprint collection can be messy, too; old methods like using ink rely on excellent ink coverage on the finger. All these variables can make the fingerprint look different. Master fingerprints can only be generated as synthetic fingerprints since there is no way to collect perfect fingerprints yet. The best way to imitate the master fingerprints in real life is to enhance the fingerprints to cleanse them from the noises on images. Master fingerprint images can be seen in 2.4



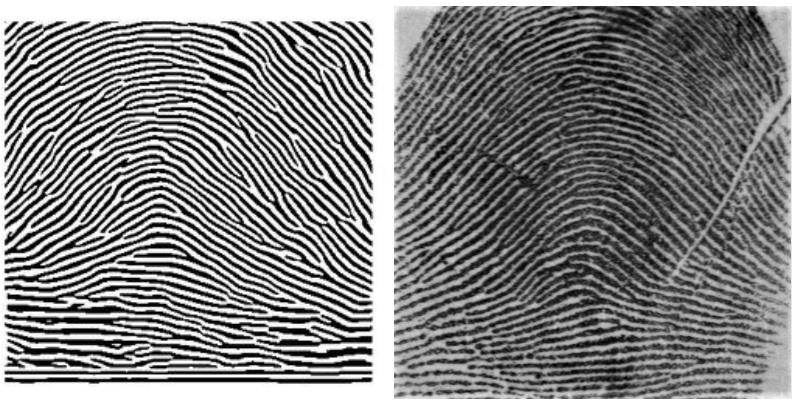
(a) Arch (b) Tented Arch (c) Right Loop (d) Left Loop

(e) Whorl

Figure 2.4. Master fingerprints of five main fingerprint classes. Generated with SFinGe [2].

2.4. Synthetic Fingerprints

Synthetic Fingerprint images are approximations of real-life fingerprint images. These images are usually generated with two different approaches. The older approach is called model-based approach. The model-based approach relies on mathematical models to generate master fingerprints. Using these master fingerprints, researchers generate different impressions of the same fingerprint since fingerprints can look different from themselves in real life. With the creation of GAN[15], researchers started to lean towards machine learning to generate fingerprints. The learning-based approach uses generative networks such as GANs and VAEs[16] to create realistic fingerprint images.



(a) Synthetic fingerprint (b) Real fingerprint

Figure 2.5. Comparison of real and synthetic fingerprints.

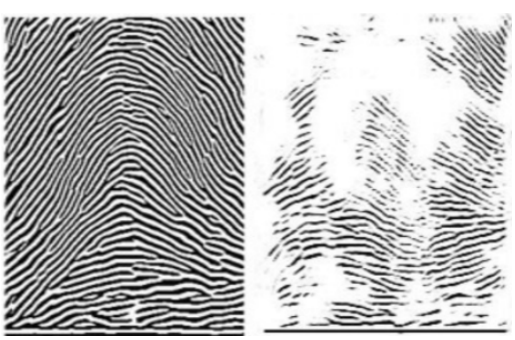
2.5. Fingerprint Image Quality

Fingerprint biometric applications all rely on high-quality fingerprint images. All governments mandates reporting fingerprint quality with all collected fingerprint databases. Most of the fingerprint verification software includes fingerprint image quality evaluation tools. These softwares are not open to public access making the quality assessment hard to improve. Different quality measurement information creates a vague environment where one system might deem the quality high while others consider it low. Some open source fingerprint image quality assesment tools are available to public and

they perform better than the most.

2.5.1. NIST Fingerprint Image Quality

NIST released the first open-source fingerprint image quality tool called NIST Fingerprint Image Quality (NFIQ)[17]. NFIQ tool classifies the given image in five classes, one through five, using eleven different features. A three-layered fully-connected feed-forward network takes all eleven features and outputs one of the five classes. Minutiae numbers, minutiae map, the total number of pixels fingerprint occupies in the image are all used as input for the network. Class one is considered as the highest quality, while class five is regarded as the worst. The network trained for NFIQ used 3900 fingerprint images from a variety of databases. Half of the total fingerprint images are from unique individuals, and fingerprints are mixed with rolled and plain impressions.



(a) Synthetic fingerprints from this study

(b) Synthetic fingerprints generated with SFinGe[2]

(c) Real fingerprints from NIST SD4[1]

Figure 2.6. High quality (left) and poor quality (right) images according to their NFIQ scores.

2.5.2. NIST Fingerprint Image Quality 2

In March 2010, "The Future of NFIQ" workshop participating researchers decided to develop the new version of the NFIQ[18]. The original version of NFIQ has been successful, and both governments and commercial businesses used the NFIQ extensively. The project's open-source nature enabled NIST to invite organizations and individuals to work on a better version. The new design standardized the input images type. Captured images have to come from either optical sensors or scanned ink, and images must be in 500 dpi. NFIQ 2[19] increased the number of features from 11 to 14. These features are:

- Regions with fingerprint images
- Frequency domain
- Local clarity score
- Orientation certainty level
- Orientation Flow
- Ridge valley uniformity
- The arithmetic mean of the grayscaled input image
- Block-based arithmetic mean of the grayscaled input image
- Minutiae count
- Minutiae quality
- ROI-based features (area mean, orientation map coherence sum, relative orientation map coherence sum)
- Local quality measures(mean, standard deviation)
- NFIQ 2 feature vector (concatenation of all quality features)
- The predictive power of NFIQ 2 features

6629 images were trained with random forest binary classification. Images classified in Class 0 have low utility(poor quality), images in Class 1 represent high utility(good quality) images. The new scoring system is based on the probability of an image belonging to Class 1. Each probability is multiplied by 100 and rounded up to the closest integer.

2.6. Generative Models

2.6.1. Generative Adversarial Networks

Generative adversarial networks (GANs)[15] are able to learn from data without needing too much explanation of the data by using two competitive networks. GANs usually consist of generator and discriminator networks. The generator's duty is to learn the real data distribution and apply it to random noise to create real data. The discriminator must figure out if the presented data is fake or real. These two players try to reach the Nash equilibrium to capture the real data distribution ultimately. There are virtually no limits to what types of architecture can be used as discriminator and generator networks. GANs used fully connected layers first; these architectures were used for simple generations like MNIST[20]. CNN's popularity and success enabled researchers to implement the convolutional layered approach to the GAN. Original GAN paper formulated the adversarial approach with a two-player minimax game function.

$$min_G max_D\ VG,D\ \mathbb{E}_{x\sim p_{data}}\log Dx\ \mathbb{E}_{z\sim p_{z_z}}\log 1-DGz \qquad (2.6.1)$$

As long as generator and discriminator networks are given an equal chance to learn, they can achieve a near-optimal solution. The function given is a binary cross-entropy function. The binary classification used has a small nuance where input data consists of real and fake parts. A Discriminator network is a binary classifier where when it encounters a real sample, it tries to maximize the output, but if the sample is provided from the generator, it does the opposite; this part is implemented as $\log 1 - DGz$ in the function. The generator cannot be trained to minimize the $\log 1 - DGz$ part. Regardless of the data to be generated, initial generator results are almost always are insufficient to fool the discriminator. A discriminator who receives these poor results can easily classify these inputs, hindering the generator's learning process. Generators try something different than minimizing $\log 1 - DGz$; they are trained to maximize the $\log DGz$; this way, both networks have the exact opposite goals, and they improve each other.

2.6.1.1. Deep Convolutional Generative Adversarial Networks

After the massive success of GANs [15] and CNNs[21], Researchers created a network called Deep Convolutional Generative Adversarial Networks (DCGAN)[3]. DCGAN's contribution to the literature opened the door for many different GAN approaches. Almost all the advanced GAN models use convolutional layers because of the success of DCGAN. Instead of using linear layers like the original GANs, DCGAN uses convolutional layers. The model removes all pooling functions and instead uses strided convolutions and follows it up with a batch normalization layer. They remove fully connected layers on deeper parts of the model. The generator uses ReLU function except for all last layers where it uses the Tanh function, and the discriminator uses Leaky ReLU function as their activation functions. DCGAN generates great results when trained on simple datasets like MNIST[20], but results for complex images like human faces and LSUN[22] are easily distinguishable as computer-generated. DCGAN suffers from model collapse on these datasets when trained for a long time.

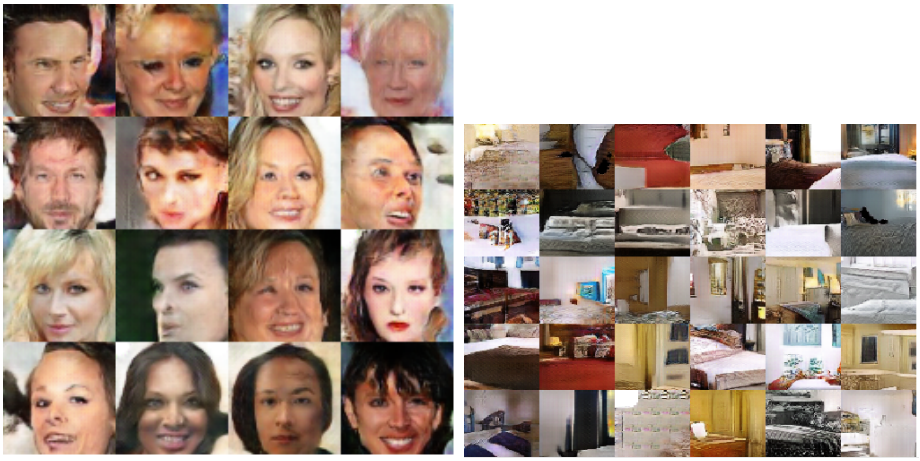
(a) Synthetic face images generated with DCGAN[3].

(b) Synthetic bedroom images generated with DCGAN[3].

Figure 2.7. Synthetic images generated with DCGAN[3].

2.6.1.2. Least Squared Generative Adversarial Networks

Following DCGAN, new, more stable GANs have been created. Least Squared Generative Adversarial Networks(LSGAN)[4] came out in 2017 following DCGAN. This approach generates higher quality images than prior GANs with a more stable training process. The least-square function penalizes the outlier samples more, even if they are correctly classified. This approach helps prevent vanishing gradient problems since regular GAN does not punish these outlying samples; the generator might create more gradients. With a lower risk of vanishing gradients, LSGAN's learning process becomes more stable. Linear least means squares are highly susceptible to these outlier data points, and these data points can easily skew the results[23].



(a) Synthetic face images generated with LSGAN[4].

(b) Synthetic bedroom images generated with LSGAN[4].

Figure 2.8. Synthetic images generated with LSGAN[4].

2.6.1.3. Relativistic Generative Adversarial Networks

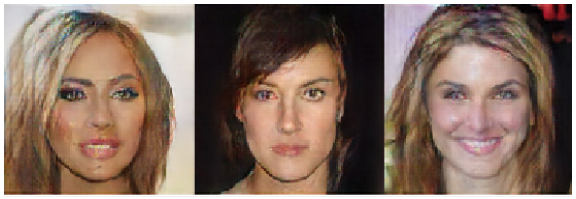
Relativistic GANs (RELGAN) [24] modify discriminators to turn into relativistic ones to produce higher quality images with stable training. After some training generator gets better at producing realistic data, the discriminator's ability to separate real data from the fake data reduces. Before a batch is given to the discriminator, there is prior knowledge

that no models use making the model relativistic. Discriminators can use the distribution percentage of real and fake samples on a batch to make better decisions. Any GAN can be turned into a relativistic version of itself by adding this knowledge to the discriminator.

2.6.1.4. Wasserstein Generative Adversarial Networks

Discriminators are more easily optimized than generators. If the generator is doing poorly while the discriminator is improving, generators will face vanishing gradients. Wasserstein GAN(WGAN)[5] uses a different cost function than the original GAN using Wasserstein distance, also called the Earth Mover's distance. WGAN's discriminator does not work like a conventional discriminator; it instead learns the K-Lipschitz function to calculate Wasserstein distance. Weights are clamped to a small range to enforce Lipschitz continuity. They note that weight clipping is not a good way to enforce Lipschitz constraints since picking an optimal range is challenging. Smaller ranges can lead to a vanishing gradient problem, or if it is too large, it can take too long to reach the optimum weights. Even though using Wasserstein distance significantly improves GANs finding the optimal clipping value and finding the number of training steps for discriminator per iteration is a time-consuming process.

(a) Synthetic face images generated with WGAN[5].



(b) Synthetic bedroom images generated with WGAN[5].

Figure 2.9. Synthetic images generated with WGAN[5].

2.6.1.5. Improved Training of Wasserstein Generative Adversarial Networks

Improved Training of Wasserstein Generative Adversarial Networks (WGAN-GP)[8] tries to solve issues of WGAN, mainly the problems that come with weight clipping. Research shows that any form of weight clipping can lead to optimization problems. Even with the best scenario, WGAN fails to converge. Rather than weight clipping, they added a gradient penalty to original critic loss. Arjovsky et al. suggest using batch normalization in WGAN to prevent vanishing gradients, but WGAN-GP penalizes critic's gradient for each input.

2.6.1.6. Progressive Growing of General Adversarial Networks

Progressive Growing of GANs (PGAN)[6] uses a different method than any other GAN. Models generator and discriminator grows progressively. Networks start with low resolutions and learn more subtle details as it grows. This approach can generate very detailed images at the end of its training. Small starting resolution helps with the memory constraints of GPUs, but as the network grows, it requires more memory, leading to a bottleneck when working with cheaper GPUs.

2.6.2. Variational Autoencoders

Autoencoders are bottleneck architectures. They try to learn how to represent data efficiently in an unsupervised environment[25]. Similar to GANs, autoencoders have two parts called encoder and decoder; however, their job is different from the two networks in GAN. Encoders compress the data and map it into code. The decoder reconstructs the data into its original form from the compressed representation. After the reconstruction process, its loss is calculated; this loss measures how close the reconstruction is to the original input. Autoencoders do not possess generative properties; however, autoencoders can

(a) Synthetic face images generated with PGAN[6].

(b) Synthetic bedroom images generated with PGAN[6].

Figure 2.10. Synthetic images generated with PGAN[6].

generate new samples by modifying the encoder. To generate new samples encoder takes the input and encodes it into a latent distribution rather than latent representation[16]. Representation is sampled from the latent space to be reconstructed. Since this representation is different from any input, the decoder will generate new samples.

2.6.3. Upsampling Methods

While training, GANs upsampling is used in the generator network to create the desired output size. Upsampling function increases the image's shape by a scale. Upsampling tries to fill the gap created by the increased size of the matrix. When matrix's size is increased new fields are added between existing fields. There are a couple of approaches to how to fill these gaps. Few methods are tested for fingerprint data.

2.6.3.1. Nearest Neighbor Interpolation

One of the simplest interpolation methods is the nearest neighbor interpolation. Points with no values are filled with the nearest point with a value. On a 2D image matrix, every value on a pixel is copied to its neighbor to increase the size of the image. This method can create a sharper, more artificial look on resized images. Advanced

interpolation methods like linear or cubic interpolation methods can create better results when an image is upsampled.

2.6.3.2. Bilinear Interpolation

This interpolation method creates new data points by looking at the existing points and tries to fit new points between the range set by existing data points. Existing points create straight-line linear polynomial, and new data is selected from this line. The new point can be calculated with a slope equation. This method is augmented to work in two directions, making the interpolation line a quadratic one.

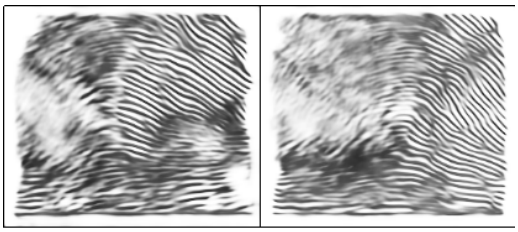


Figure 2.11. Fingerprint images generated with the GAN. Bilinear interpolation method used in upsampling.

2.6.3.3. Bicubic Interpolation

Bicubic interpolation is the extended form of Cubic Hermite Spline. Like linear interpolation, Cubic Hermet Spline produces continuous function, but the function created does not have to be linear. The function is applied to each interval, and its derivative is also continuous. This method creates one of the smoothest transitions between known data points and produces little artifact. Results are similar to Figure: 2.11.

CHAPTER 3

RELATED WORK

Synthetic fingerprint generation techniques can be classified into two classes. The first technique is model-based, and it uses mathematical models to approximate real fingerprints. Even though this technique is older than the second one, it generates excellent results. Fingerprints generated can look highly realistic to an untrained eye. Since the technique is model-based, images tend to look more uniform, while real-life samples look different than each other. The second technique is getting more popular due to advancements in deep learning. Using generative models to create synthetic fingerprints is becoming the new norm. Both GAN and CNN are used to generate synthetic fingerprints. These fingerprints can look more realistic since they learn the irregularities of fingerprints in training. The biggest problem with this technique is training stability. GANs get less stable at the later stages of the training resulting worse results. Also, to train a complex deep learning model, one needs an expensive graphic card which can deter a researcher from using a deep learning approach.

One of the most famous approaches to creating synthetic fingerprints was developed by Capelli et al. , and it is called SFinGe (Synthetic Fingerprint Generator)[2]. SFinGe was designed to imitate electronically scanned fingerprints. In a sense, SFinGe tries to invert the classic fingerprint matching algorithm to create realistic samples. Fingerprint matching algorithms remove the background from the image, generate orientation maps, and enhance the ridges to find minutiae points. SFinGe model generates a segmented fingerprint, adds noise, and places it on a background. The SFinGe model starts with creating a master fingerprint. First, the model creates a fingerprint area. The shape and size of the area are dependent on variables such as pressure from the finger, the position of the finger, and the size of the fingertip. After generating the fingerprint area the approach creates an orientation image of an imaginary fingerprint. This image again depends on variables like the class of the fingerprint and the position of the minutieas. However, the model cannot be used for arch types without any minutieas. The model generates a feasible random ridge line frequency image.Ridge frequency images represent the number

of ridges in a given length. These images are created by inspecting a large number of fingerprints. Ridge frequencies on human fingerprints tend to drop above the loops and below the deltas. With two images created, they are given as an input for ridge pattern generation. The model starts by placing dots on an empty image and enhances the image by using Gabor filters. Master fingerprints are then used to create different impressions of themselves to create intra-fingerprint variability. Different placements of the finger, dampness of the skin, cuts on the fingerprint, scanner noise, and external objects can all be used to create variabilities. These variabilities help improve recognition. Against human subjects, SFinGe's outputs are proven to be reasonably realistic. Only 23 percent of the subjects correctly classify genuine and synthetic fingerprints. The model-based approach makes it hard to create images with varying ridge thicknesses, and SFinGe is no exception. Noises on SFinGe samples are uniformly distributed, unlike real examples noises and artifacts tend to cluster over small areas of the images.

Another related work [26] attempts to create synthetic fingerprints with prespecified features. Research focuses on generating synthetic fingerprints which contact-based sensors have collected. The model-based approach takes inputs like fingerprint image size, minutiae samples, and orientation maps from their respective statistical models to synthesize the fingerprints. Level-1 and level-2 features are extracted from real fingerprint images using their statistical distribution models. These features are dependent on each other due to their nature. Singular points on fingerprints can be roughly approximated for each fingerprint type with Gaussians[27] . For each type of fingerprint, class images get aligned at their centroid points. The assumption has been made that each class can be represented with Gaussian distributions. The model generates master fingerprints with AM-FM based methods. Fingerprint ridges can be represented with AM-FM functions where FM can represent variations and AM can represent intensity values in ridge. Impressions are generated by using different types of distortions on master fingerprints. Different regions of the fingerprint are subjected to different levels of non-linear plastic distortion to emulate a real-life finger. They further render the images to create a more realistic look. This realistic look is achieved by simulating the dryness of the skin and adding noise to the image. Lastly, they apply a smoothing filter to achieve the final impression. The number of minutiae points on the synthetic images is more in line with what SFinGe produces. Research claims that this approach gives more flexibility and control over what the SFinGe method can offer.

A similar mathematical model approach was created by Imdahl et al. [28] in 2015. Their Realistic Fingerprint Creator (RFC) model starts at all foreground pixels instead of the random point start of SFinGe[2]. The orientation field from a database and five associated parameters are randomly generated. The field is then shaped according to these parameters. These parameters are picked in a plausible range not to create unrealistic orientation fields. Each pixel gets selected randomly and assigned a black or white color. This initial image was then transformed into a fingerprint image by applying Gabor filters for 50 iterations. Finally, they apply threshold and thinning operations to the generated images. They extract the minutiae points to test the similarity of real and synthetic samples. To calculate the similarity, researchers generate minutiae histograms of both real and synthetic images. Afterward, earth mover's distance metric is used to compare the histograms. With this metric, they claim that the real and synthetic images are not distinguishable from each other.

FingerGAN[29] is based on deep learning approaches. Their framework is a DCGAN with increased convolutional layers. The framework adds total variation to the loss function to prevent a dashed line look on the ridges. The model was trained on PolyU High-Resolution-Fingerprint Database and FVC2006's DB2-A dataset. The only metric the researchers show is Fréchet Inception Distance, where the generated samples scored 70.5. They claim the score is comparable to state-of-the-art models, but it does not include any of these scores. Results look noisy, and even an untrained person can easily separate the synthetic samples from the real ones.

SYNFI model [30] is a deep learning based approach to generating synthetic fingerprints. They rely on GANs and super-resolution techniques. In order to generate these images, they split these techniques into a two-phase approach. Before the phases start, they pre-process the images. They use NIST biometric image software(NBIS) to segment and centrally align the fingerprint images. They scale the images into two sizes: the first 64x64 pixels and the second is 256x256 for the GAN and Super Resolution(SR) model, respectively. In phase 1, Wasserstein GAN[*] was trained with the lower quality database. With only GANs, they could only generate 64x64 pixel images; anything higher resolution yielded unsatisfactory images. They transformed these low-resolution images into high-resolution ones in the second phase. ESRGAN[31] with Residual-in-Residual Dense blocks were used to create the higher resolution version of these synthetic images. They analyzed the indistinguishability of the real and synthetic fingerprints with different

classifier models, and all of them failed to classify since the best accuracy was only 50.43 percent.

One of the most advanced deep learning based approaches in synthetic fingerprint generation was made by Jain et al. in 2018[32]. The architecture proposed consists of Convolutional Autoencoder (CAE) and Improved-WGAN[8] (IWGAN). They acquired a database with 250 thousand fingerprint images from unnamed law enforcement to train the models. All fingerprint images from this database are resized to 512x512 pixels, and all have 500 dpi. These images did not go through any data augmentation process other than normalization. CAE extracts a compressed representation of an image given and tries to reconstruct the input image with a chosen cost function. They take advantage of the similarity between the decoder of CAE and the IWGAN generator. Initially, CAE is trained in unsupervised mode. Trained CAE's decoder then used as a starting point for the IWGAN's generator. Their IWGAN implementation has seven convolutional layers in both generator and discriminator, and the LeakyReLU activation function is used in all layers. Model is able to generate an image in 12ms on a reasonably cheap computer. Produced images achieve similar NFIQ 2 scores with a higher probability of occurrence to real fingerprint datasets like CASIA and synthetic fingerprint datasets like IBG Novetta. To find the fingerprint diversity, they calculate imposter comparision score using VeriFinger SDK 6.3. Scores shows that the proposed approach can generate more diverse looking fingerprints than IBG Novetta and SFinGe.

Wyzyowski et al.[33] created a hybrid model for fingerprint generation. They incorporated SFinGe[2] approach to create seed images. Seed images are different versions of the same fingerprints. The first stage of creating seed images is generating master fingerprints from SFinGe implementation, and then they dynamically change the ridge thicknesses of these fingerprints. To avoid sudden changes in the thickness of the ridges, they use the sine function. Pore and scratch distributions are learned from real fingerprint images to add to the master fingerprints. They use the pore-ridge reconstruction method[*] to implement pores to the master fingerprint. They also count scratches on all the fingerprints in a real fingerprint image database and use the normalized cumulative density function to choose the number of scratches to add to the master fingerprint. From the generated fingerprints, they take different positions and angles to create multiple instances. These instances are given as input to the CycleGAN[34] alongside real fingerprint images. Output images and real fingerprints are given to 60 human participants, and they failed to

discriminate real from synthetic images. The model can make synthetic fingerprints look real, but generating new fingerprint images relies on SFinGe and deep learning solely used to translate synthetic images to more realistic ones.

CHAPTER 4

METHOD

With the development of new generative networks, synthetic images started to look more realistic. One of the best examples of this is how researchers started to generate realistic human faces. The model proposed in this thesis consist of three phases. First, fingerprints from real databases are collected, and they are enhanced to look like master fingerprints. The second phase is the training and generation phase. This phase takes enhanced fingerprint images as input to train the proposed model and generate the fingerprints. Generated fingerprints are then fed to a quality checking pipeline where they are evaluated, and based on their quality, they are selected as final synthetic images.

4.1. Choosing The Best Approach

Generative models were tested with a sample subset from NIST SD4 to determine which approach yielded better results for fingerprint generation. A small subset of 100 images per class was randomly selected as the subset. Images did go through a small enhancement process using a non-local means denoising algorithm[35].

4.1.1. Variational Autoencoders

A couple of different VAE approaches were used to check whether VAE's could outperform GAN's. The first approach tested was Hyperspherical Variational Autoencoders[36]. This approach swaps the Gaussian distribution used in the classic approach to von Mises-Fischer distribution. Results achieved were lackluster compared to well-optimized GANs for the dataset given. The second model used is called Vector Quantised Variational AutoEnconder (VQ-VAE) [7]. Vector Quantised approach makes encoder outputs to be

discrete rather than being continuous. This approach also prevents posterior collapse caused by the decoder training is being faster and more robust than the encoder. Both encoder and decoder use convolutional layer structure from PixelCNN[37]. The results generated from VQ-VAE were impressive in terms of quality, but they lacked the required diversity. Hyperspherical Variational Autoencoder approach generated results that were competitive with GANs in terms of quality but lacked the expected diverse results for better training of classification models. VQ-VAE performed poorly on the raw NIST SD4 dataset, but when trained with the enhanced NIST SD4, it generated high quality and diverse images, but the images came with undesired artifacts.

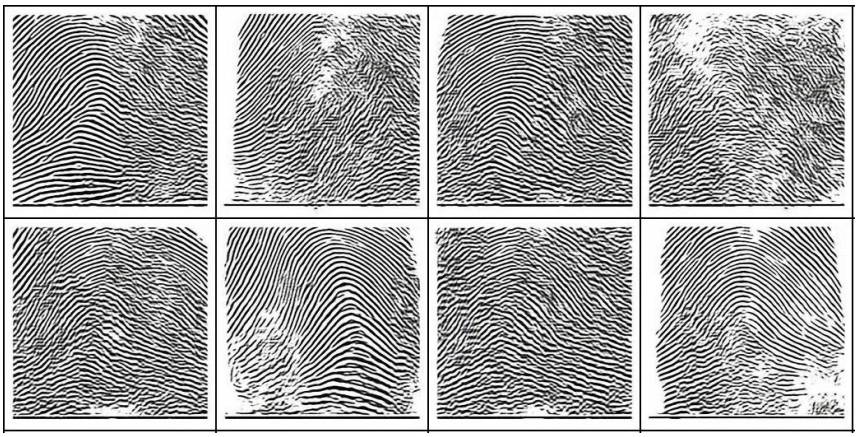


Figure 4.1. Generated results from VQ-VAE[7]. Enhanced NIST SD4 dataset were given as an input to the model.

4.1.2. Generative Adversarial Networks

Different GANs perform differently on each dataset. Fingerprint datasets are no exception. A small portion of the NIST SD4[1] dataset and its enhanced version is used as a test set to decide whether a particular GAN approach is good enough to generate fingerprints. All approaches are tuned to generate the best results from the test set, and they all have given the same amount of time and resources.

4.1.2.1. DCGAN

Fingerprint images are detailed because of their changing ridge thickness, the number of minutiae points, and their classes. These problems make DCGAN[3] an inadequate candidate for generating synthetic fingerprints.

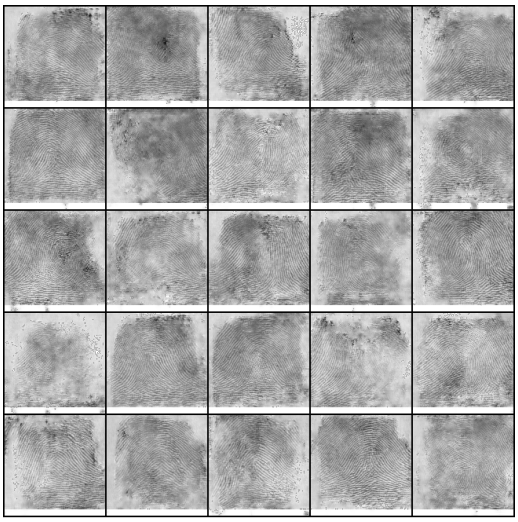


Figure 4.2. Generated results from DCGAN[3]. NIST SD4 dataset[1] were given as an input to the model.

4.1.2.2. LSGAN

Results of the LSGAN[4] are significantly better than DCGAN[3] but not enough to be chosen as the approach for fingerprint generation.

4.1.2.3. RELGAN

Results generated with RELGAN[24] were similar to the original DCGANs.

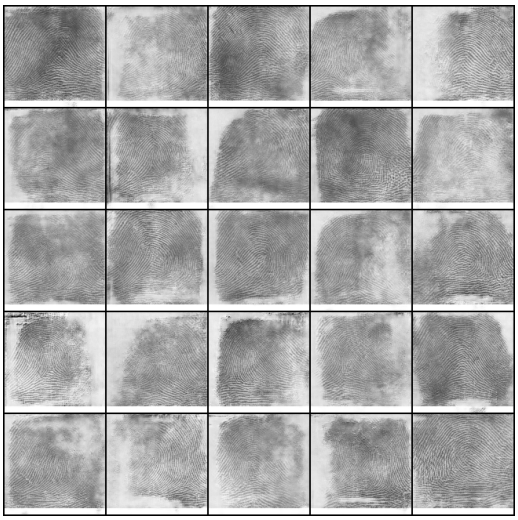


Figure 4.3. Generated results from LSGAN. NIST SD4 dataset were given as an input to the model.

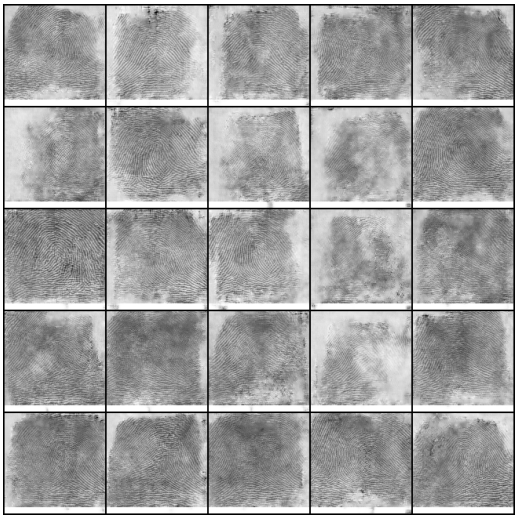


Figure 4.4. Generated results from RELGAN. NIST SD4 dataset were given as an input to the model.

4.1.2.4. WGAN

Even though using Wasserstein distance[5] significantly improves GANs finding the optimal clipping value and finding the number of training steps for discriminator per iteration is a time-consuming process. Results were clearly better than the rest of the results.

4.1.2.5. WGAN-GP

WGAN-GP[8] results had the highest quality results with more stable and relatively faster training than any other approach.

Figure 4.5. Results generated from WGAN-GP[8].

4.1.2.6. PGAN

Results generated with PGAN[6] comparable with WGAN-GP, but with Fingerprint images, WGAN-GP can achieve good-looking images faster. With a good GPU, substituting this method with WGAN-GP can lead to more realistic images.

4.2. Fingerprint Enhancement

Fingerprint enhancement is an essential step before training. Protecting the stability of the training process of GANs is essential and small noises can alter the results. Fingerprint images are noisy, and using them without any enhancement can generate bad results since the training process of GANs are not stable. Jain et al. [38] created an image enhancement algorithm for fingerprints. The thesis stays true to this enhancement algorithm but is tuned to perform better on the database used. The database used for the thesis consists of relatively high-quality images; thus, one part of the algorithm was excluded from the enhancement process. Excluded part of the algorithm finds unrecoverable parts and deletes them. The following subsections explain the fingerprint enhancement

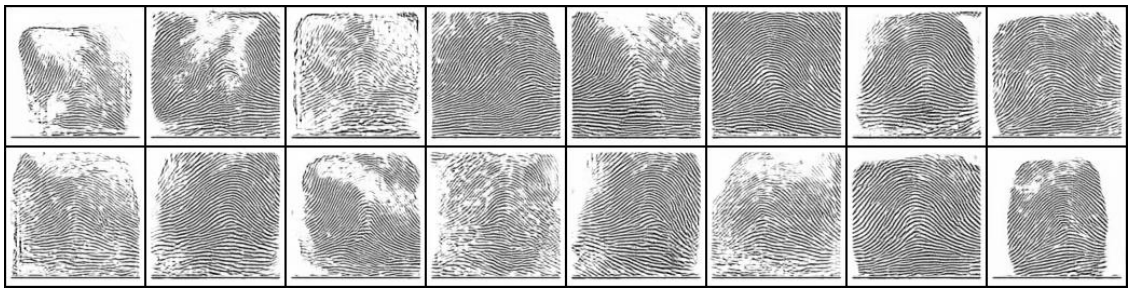


Figure 4.6. Results generated from PGAN[6]. PGAN trained with only one class because of the resource constraints.

algorithm used.

4.2.1. Normalization

Image normalization is done by calculating each pixels z score and adjusting image pixel intensities accordingly. Normalization is a pixel-wise operation, and each pixels value is recalculated using desired mean and standard deviation. Using the formula:

$$\frac{x-\mu}{\sigma} \qquad (4.2.1.)$$

where:

- μ is the mean of the pixels values
- σ is the standard deviation of the pixels values

This step is essential to almost all image enhancement processes. Real fingerprint images are all considered gray-level images where only one color channel is present, and each pixel value denotes the intensity of the gray level. Normalization aims to reduce the intensity differences on ridges. All ridges can have the similar gray levels since the model is working on Level-1 details and, the pores on the ridges and other small details are not crucial to Level-1 details. After acquiring the normalized image, this image is used as a mask to find ridges where dark areas are considered ridges.

(a) Original fingerprint image. (b) Normalized fingerprint image.

Figure 4.7. Fingerprint image before and after the normalization process.

4.2.2. Fingerprint Segmentation

The fingerprint ridge segmentation function takes a fingerprint image and returns a mask that can identify where ridges are located. Given a block size function takes the image and splits it into block-sized regions. Each region's standard deviation gets calculated. If the value lies above the predicted threshold value, that region is deemed part of the fingerprint. This method can create chunks in the mask if the ridge distance gets too low due to poor image quality or poor finger placement. After the mask is generated, the mask gets normalized again to remove any intensity differences further.

(a) Normalized fingerprint. (b) Segmented fingerprint image.

Figure 4.8. Fingerprint image before and after the segmentation process.

4.2.3. Ridge Orientation

Ridge orientation creates an estimation map of where the local orientation of the ridges is. Image is again dived into the same block-sized regions. Sobel-Feldman operator is generated in these blocks. Even though this operation can get inaccurate, it is adequate enough to approximate. Local orientation gets centered at the middle of each block. Some windows might calculate orientations wrong. Ridges tend to look continuous with no extreme changes in their orientation.

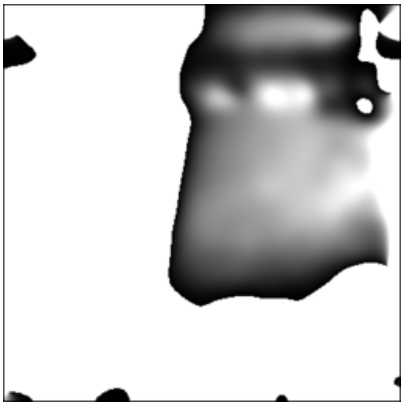


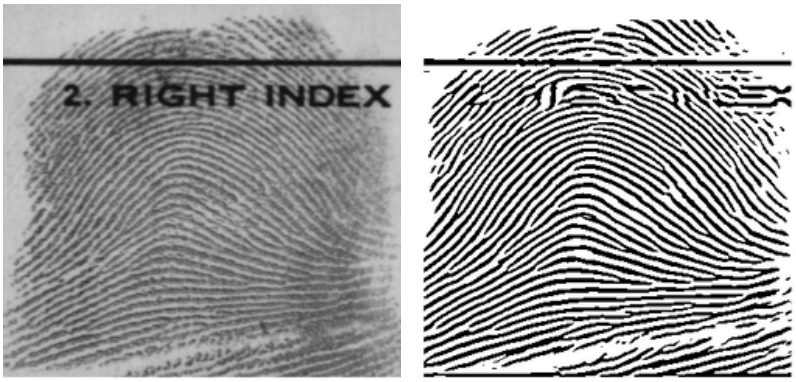
Figure 4.9. Orientation map of the fingerprint image.

4.2.4. Ridge Frequency

Uninterrupted ridges and valleys gray levels can be used to model a sinusoidal wave along the direction of the orientation. This wave is a great tool to differentiate actual ridges from undesired noises. Normalized mask images and orientation field maps can be used together to generate these waves. Same resolution fingerprints can be used as a reference point to find the frequency range of ridges and valleys. For the blocks with minutiae, they need to be interpolated using neighboring blocks.

4.2.5. Ridge Filter

The Ridge filter is the last function used in the enhancement process. This filter enhances fingerprints with oriented filters. A band-pass filter can be created using sinusoidal waves to remove the noise. Properties of Gabor filters make them an excellent band-pass filter candidate to use. The filter's frequency is taken from ridge frequency, and orientation is taken from the orientation map. Standard deviations of the Gaussian envelope are chosen based on the dataset from where the fingerprint originates from.



(a) Original fingerprint image. (b) Enhanced fingerprint image.

Figure 4.10. Fingerprint image before and after the enhancement process.

4.3. Generative Model

The model used for training is an implementation of Wasserstein loss[5] with gradient penalty into DCGAN[3]. The generator and discriminator consist of four convolutional layers and one fully-connected layer at the top. Each convolutional layer has a 3x3 pixel kernel with a one-pixel stride. All but the last layer use LeakyReLU as an activation function. The last layer uses the Tanh function instead. The generator takes random noise shaped latent space with 256 dimensions. Discriminator takes 256x256 images from both real samples and generators output.

4.3.1. Generator and Discriminator

The generator takes a 512-dimensional latent vector of random noise as an input. This input goes into a fully connected layer sized 1048576. After this, four convolutional layers follow. These layers consist of upsampling function with a scale of 2, a convolutional layer with a 3x3 kernel, a 1x1 stride, one padding, batch normalization function with 0.8 epsilon, and LeakyReLU activation function with a negative slope value of 0.2.

Generator	
Layer	Output shape
Linear	1048576
Upsample	[56, 128, 128]
Conv2d	[256, 128, 128]
BatchNorm2d	[256, 128, 128]
LeakyReLU	[256, 128, 128]
Upsample	[256, 256, 256]
Conv2d	[128, 256, 256]
BatchNorm2d	[128, 256, 256]
LeakyReLU	[128, 256, 256]
Upsample	[128, 512, 512]
Conv2d	[64, 512, 512]
BatchNorm2d	[64, 512, 512]
LeakyReLU	[64, 512, 512]
Conv2d	[1, 512, 512]
Tanh	[1, 512, 512]

Table 4.1. Generator's architecture.

The discriminator network takes either real or fake image samples as input. This image goes through deconvolutional layers to form a validation score. Discriminator layers primarily consist of reverse ordered generator layers after the feature learning discriminator's fully connected layers take a 512-dimensional feature vector. The fully connected neural network generates output after the Sigmoid function produces a score that classifies the image as either real or fake. The fully connected network also uses 0.25 dropout regularization to help with generalization.

4.4. Fingerprint Quality Assessment

The generative model can generate an infinite amount of images, but all of them are not high-quality results. After each image is generated, they should go through a quality assessment. NFIQ 1[17] and NFIQ 2[19] were used to judge the quality of the images. NFIQ 1 is older and less powerful than NFIQ 2 in quality assessment, but it is faster than NFIQ 2. NFIQ 1 is used as a preliminary to NFIQ 2. Each image that scores three or below was selected and sent to NFIQ 2. Images that score 70 or more in NFIQ 2 are selected and added to the dataset. This process provides researchers to work with high-quality images. This process can easily be turned off the get raw results as well.

CHAPTER 5

EXPERIMENTS

The generative pipeline is used to create a sizeable synthetic fingerprint database. Conducted experiments show how much the generated fingerprints are usable in the real world. Real fingerprints images are from the NIST SD4 dataset. The images in this dataset have low noise and high quality. Most importantly, this database comes with class labels for each fingerprint image. Five class labels (Arch, Tented Arch, Right Loop, Left Loop, Whorl) were used in classification experiments. Designed classification experiments use a combination of real and synthetic fingerprints to show to what extend the synthetic fingerprints can be relied on in real-life. The last part of the generation pipeline uses NFIQ and NFIQ 2 scores to generate high-quality images. Average scores of the synthetic fingerprints were compared against real fingerprints to check if the synthetic fingerprints can at least achieve the same quality as the sensor image. Minutiae points have a significant role in fingerprint biometrics; thus, having a realistic minutiae distribution and quantity is extremely important. Real and synthetic fingerprint minutiae maps, minutiae histograms, and total minutiae points were compared to show their similarity. Each experiment and its variations are shown in detail.

5.1. Classification

Picking the right architecture is essential in most classification jobs. The experiments are focused on data quality rather than architecture quality; that is why one of the state-of-the-art models was chosen rather than creating a new one. The classifier has five different output classes. These classes comes from the Level-1 features of human fingerprints. Using this classification on the ten fingers can narrow the search space and help other tools and experts identify people. The representational quality of the data is tested using the same classification architecture and same validation data with different training data. The chosen network is an improved version of ResNeXt[] developed

by NVIDIA. ResNeXt is the second-place winner in ILSVRC 2016 classification task. The network uses repeating layers just like its predecessor, ResNet. The model aims to exploit the split-transform-merge strategy introduced in Inception models. NVIDIA added a squeeze-and-extraction module[9] to the ResNeXt[39] to create SE-ResNeXt. The squeeze-and-extraction model help determines how much each channel in the convolutional block matters. Each channel in the convolutional block gets squeezed into a single value by global pooling, and all channels go into a fully connected layer followed by an activation function that creates non-linearity. These steps add slight complexity to the whole network; no performance loss is detected. Almost all the networks benefit from the squeeze-and-extraction module, which drops the top-1 error on average one percent. SE-ResNeXt101-32x4d model was used in every classification task to show the usability of the synthetic prints. The chosen model is pre-trained with ImageNet's[40] one thousand labels. Image classes chosen are part of the ILSVRC 2012 classification task still used in today's classification tasks. Every classification task used the same hyper-parameters and 10-fold cross-validation. These parameters are:

- Batch size: 8
- Learning rate: 0.08 (Scaled with batch size)
- Momentum: 0.875
- Label smoothing: 0.1
- Weight decay: $6.10e^{-}5$
- Cosine annealing learning rate scheduler
- Drop-out rate: 0.5

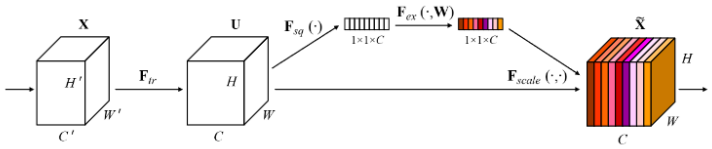


Figure 5.1. Example squeeze and extraction block from the original paper[9].

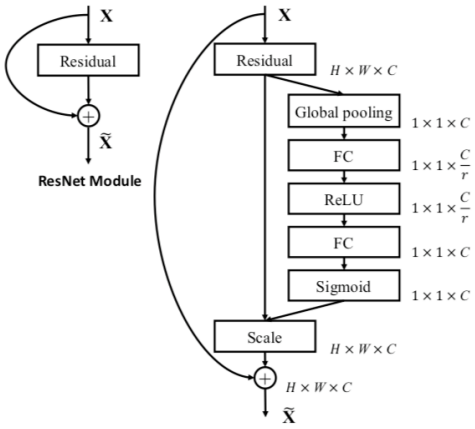


Figure 5.2. Implementation of squeeze and extraction module to the ResNet[9] model.

5.2. Experiment 1

This experiment focuses on the classification task of NIST SD4 data. Four different training datasets were given to the same classification architecture. The model was trained with these four training datasets using 10-fold cross-validation. These datasets are:

- NIST SD4
- Synthetic fingerprint generated in this thesis
- Synthetic fingerprint generated with SFinGe model
- The enhanced version of NIST SD4

All training datasets have the same amount of fingerprint images per class. 720 images were selected randomly for each class for all datasets resulting in 3600 images per dataset. The Figure shows that the model trained with NIST SD4 was able to classify the test dataset. Other models that trained with the remaining three datasets had low test accuracies. All of them had around % 20 test accuracy on average and failed to represent the features of the NIST SD4 dataset. This experiment shows that using an enhanced version of the same dataset or the synthetic fingerprint datasets chosen are not good training sets for the classification task of raw fingerprint images.

Training Set	Highest Acc %	Lowest Acc %	Average Acc %
Enhanced NIST SD4	%24	%19	%22
Thesis	%22	%21	%22
SFinGe	%24	%22	%23
NIST SD4	%89	%91	%90

Table 5.1. Test accuracy statistics of models and their training sets for experiment 1.

5.3. Experiment 2

The previous experiment showed that models trained with synthetic fingerprints could not classify Level-1 details on the NIST SD4 dataset. This experiment tries to solve this issue by enhancing the NIST SD4 dataset images. The same algorithm[38] used before the generation process was used to enhance the NIST SD4 dataset images. Three datasets were used to train the SE-ResNeXt model with 10-fold cross-validation. These datasets are:

- Synthetic fingerprint generated in this thesis
- Synthetic fingerprint generated with SFinGe model
- The enhanced version of NIST SD4

Results tell a different story than experiment 1. Training datasets were a closer representation of the enhanced dataset than the raw one. This conclusion can be seen in the figures where the test accuracies are significantly higher than in Experiment 1. One difference is that synthetic fingerprints generated in this thesis had a better representational power than the SFinGe images. Comparing the results shows that enhancing the training and test dataset does not reduce the test accuracy and can be used instead of the raw data.

Training Set	Lowest Acc %	Highest Acc %	Average Acc %
Enhanced NIST SD4	%87	%93	%91
Thesis	%69	%87	%86
SFinGe	%43	%50	%47

Table 5.2. Test accuracy statistics of models and their training sets for experiment 2.

5.4. Experiment 3

The third experiment uses different training and test dataset pairs. Three datasets were used in this experiment. With using three datasets, three training and test dataset pairs were created. These are: Thesis - SFinGe SFinGe - Thesis Thesis - SFinGe Impressions

- Synthetic fingerprint generated in this thesis - Synthetic fingerprint generated with SFinGe model
- Synthetic fingerprint generated with SFinGe model - Synthetic fingerprint generated in this thesis
- Synthetic fingerprint generated in this thesis - Synthetic fingerprint impressions generated with SFinGe model

Each dataset consisted of 1000 images. The classification model trained with the synthetic fingerprint dataset generated in this thesis and its validation accuracies against both SFinGe datasets was recorded. The model's test accuracy was high in both cases, achieving %94 on average for SFinGe and %81 on average for SFinGe impressions. The model trained with SFinGe had a low test accuracy with only %69 on average. This experiment shows that when a model is trained with the synthetic fingerprint images dataset, it generalizes better than SFinGe fingerprints.

Training Set - Test set	Highest Acc. %	Lowest Acc. %	Average Acc. %
Thesis - SFinGe	%96	%84	%94
Thesis - SFinGe Impressions	%82	%59	%81
SFinGe - Thesis	%72	%52	%69

Table 5.3. Test accuracy statistics of models and their training sets for experiment 3.

5.5. Experiment 4

The last experiment focuses on if synthetically generated fingerprints help with the classification when used with the real fingerprint data. For this, both SFinGe and generated dataset were added to the enhanced NIST SD4 dataset and enhanced NIST SD4

were used as the test dataset. The experiment was conducted in two parts. For the first part of the experiment, equal parts of both enhanced NIST SD4 and synthetic fingerprints were combined into one training set. The second part adds more synthetic data to the training set to see if results get better with more data or worse with adding more synthetic data to the training set. Results show that adding the same amount of synthetic data generated in this thesis as NIST SD4 to the training set improves models test accuracy slightly. However, with adding the SFinGe dataset, that slight improvement was not captured. Adding even more SFinGe data did not improve the test accuracy. Even though synthetic fingerprints generated in this thesis increased the test accuracy when used as equal parts with enhanced NIST SD4, adding more synthetic data dropped the accuracy back to the original enhanced NIST SD4 data. These results show that the representative power of synthetic fingerprints is close to the real-life fingerprint images but not enough to improve real-life datasets.

Training Set	Highest Acc. %	Lowest Acc. %	Average Acc. %
%50 Thesis+%50 Enh. SD4	%93	%85	%92
%66 Thesis+%33 Enh. SD4	%91	%86	%90
%50 SFinGe+%50 Enh. SD4	%91	%85	%90
%66 SFinGe+%33 Enh. SD4	%90	%87	%90

Table 5.4. Test accuracy statistics of models and their training sets for experiment 4.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1. Conlusion

The lack of publicly available datasets hindered the researchers. Synthetic fingerprints can not be matched with any human's fingerprint, and a model can generate an infinite number of them. A deep-learning-based pipeline was designed for this study with these opportunities in mind. The pipeline enhances the publicly available dataset to help with the generation process. Generative models take the enhanced dataset to generate candidate synthetic fingerprints. These candidate fingerprints go through an elimination process where models trained to assess fingerprint quality eliminate those with low scores. After these steps, a final synthetic database is generated. Results show that the generated models trained with synthetic fingerprints can classify enhanced real data with %86 accuracy, which is a significant improvement from SFinGe master fingerprints. Results also show that enhancing real fingerprint images helps models trained with synthetic fingerprints classify. This is crucial since the aim is to use models only trained with synthetic fingerprints in real-life scenarios. Study shows that synthetic fingerprints can replace real fingerprint datasets in the near future.

6.2. Future Work

The realism of the data is not yet explored since all of the state-of-the-art methods are unavailable to public access. Even though the experiments show that enhancing the fingerprint data is acceptable new parts can be added to the pipeline to make the images look more like real-life samples. The progressive GAN approach can be added to the model to generate better results.

REFERENCES

[1] Sherena.johnson@nist.gov, "Nist special database 4," Jul 2018.

[2] R. Cappelli, D. Maio, and D. Maltoni, "Synthetic fingerprint-database generation," in *Object recognition supported by user interaction for service robots*, vol. 3, pp. 744–747, IEEE, 2002.

[3] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[4] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802, 2017.

[5] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, PMLR, 2017.

[6] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[7] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," *arXiv preprint arXiv:1711.00937*, 2017.

[8] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *arXiv preprint arXiv:1704.00028*, 2017.

[9] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[10] Sherena.johnson@nist.gov, "Nist special database 14," Jul 2018.

[11] K. Wertheim, "Embryology and morphology of friction ridge skin," *The fingerprint sourcebook*, pp. 103–126, 2011.

[12] E. H. Holder, L. O. Robinson, and J. H. Laub, *The fingerprint sourcebook*. US Department. of Justice, Office of Justice Programs, 2011.

[13] C. W. Schultz, J. X. Wong, and H.-Z. Yu, "Fabrication of 3d fingerprint phantoms via unconventional polycarbonate molding," *Scientific reports*, vol. 8, no. 1, pp. 1–9, 2018.

[14] E. Henry, "Classification and uses of finger prints.[sl]: George routledge and sons," 1900.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Advances in neural information processing systems," *Curran Associates, Inc*, pp. 2672–2680, 2014.

[16] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[17] K. Ko *et al.*, "User's guide to nist biometric image software (nbis)," 2007.

[18] O. Bausinger and E. Tabassi, "Fingerprint sample quality metric nfiq 2.0," *BIOSIG 2011–Proceedings of the Biometrics Special Interest Group*, 2011.

[19] E. Tabassi, M. Olsen, O. Bausinger, C. Busch, A. Figlarz, G. Fiumara, O. Henniger, J. Merkle, T. Ruhland, C. Schiel, *et al.*, "Nfiq 2 nist fingerprint image quality," 2021.

[20] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[21] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision*, pp. 319–345, Springer, 1999.

[22] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[23] N. A. Heckert, J. J. Filliben, C. M. Croarkin, B. Hembree, W. F. Guthrie, P. Tobias, J. Prinz, *et al.*, "Handbook 151: Nist/sematech e-handbook of statistical methods," 2002.

[24] W. Nie, N. Narodytska, and A. Patel, "Relgan: Relational generative adversarial networks for text generation," in *International conference on learning representations*, 2018.

[25] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 37–49, JMLR Workshop and Conference Proceedings, 2012.

[26] Q. Zhao, A. K. Jain, N. G. Paulter, and M. Taylor, "Fingerprint image synthesis based on statistical feature models," in *2012 IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pp. 23–30, IEEE, 2012.

[27] R. Cappelli and D. Maltoni, "On the spatial distribution of fingerprint singularities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 4, pp. 742–448, 2008.

[28] C. Imdahl, S. F. Huckemann, and C. Gottschlich, "Towards generating realistic synthetic fingerprint images," *2015 9th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pp. 78–82, 2015.

[29] S. Minaee and A. Abdolrashidi, "Finger-gan: Generating realistic fingerprint images using connectivity imposed gan," *arXiv preprint arXiv:1812.10482*, 2018.

[30] M. S. Riazi, S. M. Chavoshian, and F. Koushanfar, "Synfi: Automatic synthetic fingerprint generation," *arXiv preprint arXiv:2002.08900*, 2020.

[31] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *Proceedings of the European conference on computer vision (ECCV) workshops*, pp. 0–0, 2018.

[32] K. Cao and A. Jain, "Fingerprint synthesis: Evaluating fingerprint search at scale," in *2018 International Conference on Biometrics (ICB)*, pp. 31–38, IEEE, 2018.

[33] A. B. V. Wyzykowski, M. P. Segundo, and R. de Paula Lemes, "Level three synthetic fingerprint generation," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 9250–9257, IEEE, 2021.

[34] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[35] A. Buades, B. Coll, and J.-M. Morel, “Non-local means denoising,” *Image Processing On Line*, vol. 1, pp. 208–212, 2011.

[36] T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak, “Hyperspherical variational auto-encoders,” *arXiv preprint arXiv:1804.00891*, 2018.

[37] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, “Conditional image generation with pixelcnn decoders,” *arXiv preprint arXiv:1606.05328*, 2016.

[38] L. Hong, Y. Wan, and A. Jain, “Fingerprint image enhancement: algorithm and performance evaluation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 8, pp. 777–789, 1998.

[39] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

[40] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.