# Generative Models for Handwriting Synthesis and Imitation

1st Rifumo Mzimba
*School of Computer Science & Applied Mathematics*
*University of the Witwatersrand*
Johannesburg, South Africa
1619542@students.wits.ac.za

2nd Richard Klein
*School of Computer Science & Applied Mathematics*
*University of the Witwatersrand*
Johannesburg, South Africa
richard.klein@wits.ac.za

***Abstract*—Generative Advesarial Networks (GANs) have enabled computers to imitate style and generate data it has it has never seen before. We investigate the ability of GANs to synthesize and imitate a person's handwriting style given just a few sample images of their handwriting. We extend the GAN-Writing model to imitate handwriting styles without the predefined writer styles which limits its imitation ability for new handwriting styles. We show that GANs can imitate a person's handwriting and passes the Turing test. We also show that the combination of the triplet loss and cycle consistency make the GAN-Writing model to imitate handwriting styles in a few-short learning paradigm and also converges faster.**

***Index Terms*—handwriting imitation, handwriting synthesis, GANs, triplet loss, cycle consistency, handwriting generation**

## I. Introduction

Handwriting synthesis is the process of training a machine to produce text that looks handwritten by a human. Different from script fonts, the generated characters need to show randomness that exists in real handwriting. Handwriting imitation constrains the synthesized text to the unique traits of a given person's handwriting style. Having a computer do this can allow us to send mass personalized invitations, emails, social media posts, and gift messages. Authors can write books using their handwriting or the handwriting of historical authors. The response rate of handwritten text has been shown to be more than double that of typed text [1].

Researchers have used complex mathematical methods to generate handwriting but the results look synthetic [2]. A key influence came from Graves [3] who proposed a recurrent neural network (RNN) based generator where they treat the problem as a sequence generation. The model managed to generate natural-looking text but the styles of the model are limited to samples in the training set. It cannot mimic a specific handwriting. State-of-the-art are inspired by Generative Adversarial Networks [4] which have been shown to have great generative capabilities in tasks such as cartoon imaging, shoe design, handwriting profiling, generation of images using text description, missing data imputation, enhancement and denoising of speech, to list but a few [5]. Most of the models are designed to synthesize handwriting acquired from online sources. [6] proposed a GAN derivative model called GANwriting that synthesizes offline handwriting. They use a writer-classifier network to force the generator to imitate a writer's style of handwriting. As a result, their model struggles to generate handwriting styles not in the writer-classifier. We propose incorporating cycle consistency and triplet loss to train the model.

The following sections are structured as follows: We discuss the background of GANs and their evaluation metrics in II, as well as writer identification, triplet loss and cycle consistency. In III we discuss related work in handwriting synthesis. IV describes our baseline model followed by our experiments, results and evaluation in V. We then provide a summary and conclusion in VI.

## II. Background

### A. Generative Adversarial Networks

*a) Generative Adversarial Networks overview:* GANs [4] are based on min-max game theory, posed as a zero-sum game where two networks, a discriminator ($D$) and a generator ($G$) compete to achieve nash equilibrium [7]. $G$ takes stochastic noise and generates data while $D$ classifies it as real or generated. $G$ learns from $D$'s feedback with no access to the real data input. For random noise $z$ and real data $x$ the objective function of the vanilla GAN can be stated using minimax value function as:

$$\min_G \max_D (\mathbb{E}_{x\sim p_x}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z))] \quad (1)$$

where $p_x$ and $p_z$ are the training distribution and the prior on input noise distribution, respectively. Since the discriminator can learn to distinguish generated images with high confidence during early training while the generator is poor, it causes the gradients to diminish, i.e., $\log(1 - D(G(z)))$ saturates, and the gradients no longer provide sufficient information for the generator to learn. Hence, the generator often maximizes $\log D(G(z))$ rather than minimizing $\log(1 - D(G(z)))$ in practice [4].

*b) Conditional Generative Adversarial Networks:* [8] introduced a conditional GAN (cGAN) which adds a conditional variable $c$ as input to $G$ and $D$ to stabilize training and

generate samples of a specific type. The objective function then becomes:

$$\min_G \max_D (\mathbb{E}_{x\sim p_x}[\log D(x|c)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z|c))] \quad (2)$$

*c) Training:* The GANs are trained jointly by the alternating gradient descent whereby one network is fixed while the other takes a single gradient descent iteration. These steps are repeated until the generator can produce plausible images to fool the discriminator, i.e., the discriminator outputs a probability of 0.5 for both the the real and generated data consistently. The discriminator feedback becomes random and less meaningful. If the GAN continues training past this point the model may destabilize and never converge.

*d) Challenges:* As already discussed, GANs suffer from diminishing gradients and convergence. GANs also show difficultly to reach Nash equilibrium during training [9]. They can get wedged in a bad local minimum [10]. Mode collapse and mode dropping are some of the other challenges for GANs. Mode collapse occurs when generator learns to map several different input vectors $z$ to the same output $\bar{x}$. Hence, it only produces a limited variety of samples that can fool the discriminator. Mode dropping is when some hard-to-represent modes of $p_x$ are simply "ignored'" by $p_z$. [9] – [13] propose several remedies for these challenges.

### B. Evaluation Metrics

The lack of a universal evaluation metric makes it difficult to measure the performance of GANs for different tasks. [14] provides several possible metrics to use. We discuss the Inception Score (IS) [15] and Frèchet Inception Distance (FID) [18].

*a) Inception Score:* IS measures the quality and diversity of the generated samples using the Inception Net [16] pre-trained on the ImageNet [17] dataset to acquire the conditional label distribution $p(y|x)$. The entropy of the condition label distribution $p(y|x)$ is expected to be low and high for the marginal distribution $p(y) = \int_z p(y|x = G(x))d_z$, depicting the samples are highly classifiable and diverse. IS is computed as the exponent of the average KL divergence between these two distributions:

$$IS(G) = \exp(\mathbb{E}_{x\sim G(z)}[KL(p(y|x)||p(y))]) \quad (3)$$

This has been shown to correlate to human perception of quality and diversity [15]. However, the IS cannot detect mode collapse or overfitting.

*b) Frèchet Inception Distance (FID):* [18] The FID uses the pre-trained Inception Net [16] layers as a feature space for the generated samples and training samples respectively, where the feature space is a continuous Gaussian distribution. It then computes the distance between their means and covariances:

$$FID(x, \bar{x}) = ||\mu_x = \mu_{\bar{x}}||_2^2 + Tr(\Sigma_x + \Sigma_{\bar{x}} - 2(\Sigma_x\Sigma_{\bar{x}})^{\frac{1}{2}}) \quad (4)$$

where $(\mu_x, \Sigma_x)$, and $(\mu_{\bar{x}}, \Sigma_{\bar{x}})$ are the mean and covariance of the real data and model generated data distributions, respectively. Lower FID means that the two groups of images are more similar, with 0 implying that they are identical. Lower FID has been shown to correlate to higher image quality and diversity. Unlike IS, FID can detect intra-class mode dropping and is less sensitive to noise.

### C. Optical Character Recognition

The inverse problem of handwriting generation is handwriting recognition. Optical character recognition (OCR) is the process that converts input images into editable data [19]. Depending on data acquisition, OCR can be categorized into offline and online recognition. Offline refers to recognizing data that is in rasterized format [20], whereas online recognition recognizes data stored as a pen-tip ordered location sequence [21]. Online data provides dynamic information such as locus point and projection angles which is useful to distinguish overlapping characters from each other. Offline data is static and lacks this information, hence, more difficult to handle [22]. [19], [31], and [22] provide literature surveys on techniques used to solve OCR.

### D. Writer Identification

Writer identification (WI) finds the writer of a document based on similarities to a stored reference list with documents of which the writers are known. While handwriting recognition pays no attention to the writer's handwriting features, writer recognition uses these features to identify the writer. [23], [24], and [25] provide literature surveys on techniques used in WI.

### E. Triplet Loss

The triplet loss is a distance based loss function that computed between an anchor $a$ input, a positive $p$ input and a negative $n$ input:

$$\mathcal{L}(a, p, n) = max(d(a, p) - d(a, n) + margin, 0) \quad (5)$$

where $d$ is some distance on the embedding space. When we minimize this function, the anchor and positive become close, $d(a, p) \to 0$ while the anchor and negative are pushed further away, i.e., the distance between the anchor and negative increases, $d(a, n)$ becomes greater than $d(a, n) + margin$.

Triplets can be categorized as easy, semi-hard, and hard based on on their embedding distances $d$ from each other. If $d(a, p) + margin < d(a, n)$, then the triplet is classified as easy, and if $d(a, n) < d(a, p)$ the triplet is classified as hard. In the case where $d(a, p) < d(a, n) < d(a, p) + margin$, the triplet is classified as semi-hard.

These triplets can be obtained offline where the triplets are generated at the beginning and then fit the data to the network, or online where the triplets are computed from the batch. The online model proposed by [26] is more faster and more efficient.

[27] discusses two methods for mining online triplets. These are the batch all and batch hard. Batch all averages the loss of all hard and semi-hard valid triplets in the batch while batch hard uses the hardest positive and the hardest negative in the batch to compute $d$. [27] found that batch hard yields better results than batch all. But this is also dependent on the dataset.

### F. Cycle Consistency

Cycle Consistency Loss [28] performs unpaired image-to-image translation where it enforces the mapping $G : X \rightarrow Y$ and $F : Y \rightarrow X$ should be bijections. Hence, the cycle consistency loss is given by

$$\mathcal{L}_c(G, F) = \mathbb{E}_{x \sim p_x}[||F(G(x)) - x||_1] + \mathbb{E}_{||y \sim p_y}[G(F(x)) - y||_1] \quad (6)$$

## III. Related Work

[29] and [30] reviewed state-of-the-art handwriting synthesis techniques before the popularization of Deep Learning algorithms in handwriting generation. The following papers came after the reviews and have shown prominent enhancements from previously used methods. They have been divided into online and offline synthesis and highly focused on the Latin script, which the English and a majority of languages fall. Different techniques work better in different scripts as the scripts can be inherently discrete, cursive or both [30], [31]. There's been more work and success for online handwriting but people's online handwriting is substandard, hence the results as well. Offline synthesis can have more practical use than online synthesis.

*1) Online:* [3] proposed a novel idea that revolutionized handwriting synthesis. They approached handwriting synthesis as a sequence generation problem hence proposed the use of RNNs to approximate the probability distribution of the handwriting sequence. They added Long-Short Term Memory (LSTM) networks to increase the information remembered by the RNN. This, in turn, enabled the current values of the sequence and the hidden state of RNN to be able to predict the next probability density function of the next value of the sequence. A Mixture Density Networks (MDN) and Attention Mechanism were added to condition the network's predictions to a specific text sequence. The results produced look realistic. However, their model generates a hallucinated handwriting style rather than imitating a specified handwriting sample.

[21] proposed training the top layer of the LSTM cell of the model proposed by [3] to imitate a sample handwriting style, but the model fell short to the limitations of RNNs and LSTMs. The model could not generate longer strings and retraining the top layer of the model was cumbersome, limiting the model for practical use. They did not add a huge improvement to [3]'s model.

A proposal from [32] was to use a Deep Convolutional GAN (DCGAN). A third parameter that had the handwriting images with incorrect labels (ASCII character values) was added to the discriminator, in addition to the generated and real input images. As the discriminator learned to score them as fake, it learned not to only judge input images as real or fake, but to also match the character embedding. Reinforcement learning (RL) was used to join letters and form words. With this advancement, the generator learned to space characters and to make strokes from one letter to another better.

The model proposed by [33] is a modified GAN, where the discriminator has an integrated CNN-LSTM feature extraction and a Feedforward Neural Network classifier. The handwriting strokes are encoded following Path Signature Features (PSF). They used the generator proposed by [3]. The model generates handwritten texts which are neat and have miscellaneous styles and a uniform spatial distribution more than the model proposed by [3]. [34] combine CPPN (Compositional Pattern Producing Networks), VAE, and GAN models to generate high-resolution handwriting images. The new model (VAE/CPGAN) is shown to produce high-resolution images that outperform images generated from VAE [35], [36], VAE/GAN [37], DCGAN [32], [38] and CPPN-GAN-VAE [39] evaluated using the Inception Score metric. Furthermore, VAE/CPGAN converges faster than the compared models.

The idea of [40] is to predict single pen positions, a model that is independent of the internal memory of the network to store style, in contrast to the attention mechanism model proposed by [3]. The proposal uses CVRNN to disentangle the input into two latent variables, style and content. This improved the control for style generation in [3]'s model better than [33]'s proposal. In a successive proposal, [41] replaced the CVRNN with a Stochastic Temporal CNN (STCNN). The handwriting generation became more consistent.

*2) Offline:* [42] use labeled segmented glyphs and structured texture synthesis to synthesize and imitate handwriting. They managed to produce realistic handwritten text that looks like that of the author. The challenge with the model is that it requires intensive human intervention to select glyphs and label the ligatures. An additional challenge is that the model cannot reproduce letters that are not in the input text.

Another novel idea which state-of-the-art models are built on was proposed by [43]. They use a modified GAN with an auxiliary network to assist in recognizing offline text. The generator is fed an encoded sequence of characters to be generated through a bidirectional LSTM recurrent layer. They integrate the generated images into the training data in contrast to all the papers above and below, except for [6], and showed that it can increase recognition accuracy. The model cannot output varying sizes of words and it suffers from style collapse.

[20]'s work was inspired by [43]. Their model improves on the lack of varying length in words and images, and the need to annotate words at a character level. They proposed a semi-supervised fully convolutional handwriting text generator. A tweaked BigGAN was used. The handwriting images produced by this model were shown to be clearer, more versatile, and to have fewer artifacts under FID and GS metrics. This work, however, cannot generate characters with different receptive field widths.

[44] propose a fully automated spatial-temporal style transfer to imitate handwriting. They compute the skeleton of the input text using a proposed iterative knowledge transfer skeletonization algorithm. Afterward, they approximate the skeletonized sequence to an online sequence by converting the bitmap skeleton representations to strokes, and they obtain temporal information using maximum acceleration resampling and ordering. [3]'s model was used as the generator for handwriting synthesis. A modified *pix2pix* [45] was used to imitate ink and style of the original image from the online

handwriting skeleton produced, hence producing realistic offline handwritten text. The produced text does not always look like that of the writer and is unrealistic when the skeletonization does not construct complete skeletons. Furthermore, the model has difficulties synthesizing punctuation marks.

[6]'s proposal is similar to [42] and [43]'s work. They proposed a GAN with the generative process conditioned with textual content and calligraphic style features. The model is non-recurrent to produce the final word image, removing the need for pen-tip position sequences.

## IV. GANWriting Architecture

### A. Problem formulation

The multi-writer handwritten dataset is denoted by $\{\mathcal{X}, \mathcal{Y}, \mathcal{W}\}$, where $\mathcal{X}$ represents the handwritten word images, $\mathcal{Y}$ is the corresponding handwritten text, and $\mathcal{W} = \{w_i\}_i^N$ represents the writer identities of the writers. Let $\mathcal{A}$ be the alphabet consisting of every possible text string of the length $l$. The handwriting model $G$ is defined as:

$$\bar{x} = G(t, X_i) = G(t, \{x_1, ..., x_K\}) \tag{7}$$

where $X_i = \{x_{w,j}\}_k^K = 1 \subset \mathcal{X}$ is a few-shot example of $K$ images samples from $w_i \in \mathcal{W}$, $t \in A^l$, and $\bar{x}$ is the generated word with the textual content $t$ and the handwriting style of writer $w_i$. The model overview is presented in Fig. 1.

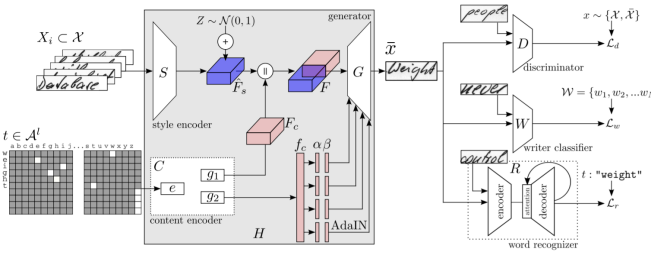


Fig. 1. Architecture of the GANWriting model [6].

### B. Generative Network

The generative architecture of $G$ is composed of a style encoder $S$, a textual content encoder $C$ and a conditioned image generator $G$.

*1) Style encoder:* The style encoder $S$ takes in $X_i$ and disentangle the calligraphic style attributes (stroke width, glyph shapes, ligatures, character roundness, slant, etc.) from their individual textual content. All the images $x_j \in X_i$ are resized to height $h$ and padded to width $w$ resulting in a single tensor $h \times w \times K$, where $w = 216, h = 64$, and $K = 15$. This tensor then gets fed into $S$ which uses VGG-19-BN [46] to learn a style latent space mapping $F_s = S(X_i)$. To imitate the variance in human handwriting for the same word written multiple times, $X_i$ is randomly chosen from the permutations of each writer $w_i$. Noise $Z \sim \mathcal{N}(0, 1)$ is also added to $F_s$ resulting in a subtle distorted feature representation $\hat{F}_s = F_s + Z$.

*2) Textual content encoder:* Let $e : \mathcal{A} \to \mathbb{R}^n$ be a character-wise embedding function. The textual content encoder $C$ is made up of two encoders: one performs low-level character-wise encoding $g_1 : \mathbb{R}^n \to \mathbb{R}^m$ and the other performs a global text encoding $g_2 : \mathbb{R}^{l \cdot n} \to \mathbb{R}^{2p \cdot q}$ to the given text $t$. Both $g_1$ and $g_2$ are Multi-Layer Perceptrons (MLPs) composed of three fully connected layers activation functions and batch normalization [47]. The input text $t$ is padded with an empty string $\epsilon$ if shorter than $l$. $F = [\hat{F}_s || F_c]$ is the concatenation of the content feature map $F_c$ from $g_1$ and $\hat{F}_s$ which gets fed into the the generator $G$.

*3) Generator:* The generator consists of two residual blocks [48], whereby the normalization layer is the AdaIN [49], defined as:

$$AdaIN(z, \alpha, \beta) = \alpha \left( \frac{z - \mu(z)}{\sigma(z)} + \beta \right) \tag{8}$$

where $z \in F$, $\mu$ and $\sigma$ are channel-wise mean and standard deviations, $\alpha = p$ and $\beta = 2q$ from the global textual encoding $f_c$.

Overall, the generative network can be defined as:

$$\bar{x} = G(t, X_i) = G(g_1(\hat{t}), g_2(\hat{t}), S(X_i)) \tag{9}$$

where $\hat{t} = [e(c); \forall c \in t]$.

### C. Learning Objectives

The learning objective is guided by the discriminator $D$, writer identifier $W$, and the word recognizer/OCR $R$. The first two networks, $D$ and $W$ are constructed using one convolutional layer at the beginning then six residual blocks with LeakyReLU activations and average poolings. They only differ at the last layer, where $D$ uses another convolutional layer as a binary classifier while $W$ uses a MLP to classify the input images to one of the writers in the training set. The OCR $R$ on the other hand uses the attention-based sequence-to-sequence model [50]. The discriminate loss $\mathcal{L}_d$ is the original loss defined in (1). The writer classifier $W$ uses the cross entropy loss:

$$\mathcal{L}_w(G, W) = -\mathbb{E}_{x \sim \{\mathcal{X}, \hat{\mathcal{X}}\}} \left[ \sum_{i=1}^{|\mathcal{W}|} w_i \log(\hat{w}_i) \right] \tag{10}$$

where $\hat{\mathcal{X}}$ are the generated images, $w_i$ and $\hat{w} = W(x)$ are the real writer distribution and the predicted probability distribution over writers in $W$ respectively. The OCR $R$ loss function is the Kullback-Leibler divergence loss:

$$\mathcal{L}_r(G, R) = -\mathbb{E}_{x \sim \{\mathcal{X}, \hat{\mathcal{X}}\}} \left[ \sum_{i=1}^{l} \sum_{j=1}^{|\mathcal{A}|} t_{i,j} \log \left( \frac{t_{i,j}}{\hat{t}_{i,j}} \right) \right] \tag{11}$$

where $t_i$ and $\hat{t}_i$ are the $i^{th}$ character of $t$ and decoded $i^{th}$ decoded character respectively, $t_{i,j}$ and $\hat{t}_{i,j}$ denote the real probability and $W$'s probability of the $j^{th}$ character in $\mathcal{A}$.

The total loss becomes the losses of the three learning objectives:

$$\mathcal{L}(G, D, W, R) = \mathcal{L}_d + \mathcal{L}_w + \mathcal{L}_r \tag{12}$$

$$\min_{G,W,R} \max_{D} \mathcal{L}(G, D, W, R) \tag{13}$$

## V. Experimental Results

### A. Dataset

Several publicly available datasets are available for research use. For the purpose of this research, we will be using the IAM Handwriting Database [51] similarly to our baseline model [6]. The IAM Handwriting Database [51] has 657 writes and 115320 words separated and labeled. It has been used for writer identification and verification, recognition tasks, and generation tasks. Their data is obtained from scanned documents at 300dpi resolution. There are several text corpora that can be used for out-of-vocabulary (OOV) words. Similar to [6] we use the Brown [52] corpus. [6] uses a subset of 22500 unique English words from it. They set aside 160 writers from the IAM Database for testing. They also set aside 400 unique words from the IAM transcript for evaluation during training. We follow a similar setup.

### B. Experimental Setup

We first $(a)$ setup the GANWriting [6] model and run it as our baseline model. We then $(b)$ add cycle consistency, $(c)$ triplet loss, then $(d)$ both cycle consistency and triplet loss to the the loss function $\mathcal{L}(\mathcal{G}, \mathcal{D}, \mathcal{W}, \mathcal{R})$ in (12) to become:

$$\mathcal{L}(G, D, W, R) = \mathcal{L}_d + \mathcal{L}_w + \mathcal{L}_r + \mathcal{L}_s \tag{14}$$

After these we remove the writer identifier in [6] for all experiments, $(a) - (d)$. Hence, the loss function becomes:

$$\mathcal{L}(G, D, R) = \mathcal{L}_d + \mathcal{L}_w + \mathcal{L}_s \tag{15}$$

We present our algorithm adapted from [6] in (1), where $w_{\mathcal{L}_*}$ represents the weight of the loss. It is set to 0 to exclude a certain loss.

**Algorithm 1** Algorithm for GANWriting

**Input:** $\{\mathcal{X}, \mathcal{Y}, \mathcal{W}\}$; alphabet $\mathcal{A}$; max training iterations $T$
**Output:** Networks parameters $\{\Theta_G, \Theta_D, \Theta_W, \Theta_R\}$

**for** $i = 0$ to $T$ **do**
  $\mathcal{L}_d \leftarrow$ (1)
  $\mathcal{L}_{r,w} \leftarrow$ (10) + (11)
  $\Theta_D = \Theta_D - \Gamma(\Delta_{\Theta_D} \mathcal{L}_d)$
  $\Theta_{R,W} = \Theta_{R,W} - \Gamma(\Delta_{\Theta_{R,W}} \mathcal{L}_{r,w})$
  $\mathcal{L}_s \leftarrow$ (5) **or** (6) **or** ((5) + (6))
  $\mathcal{L} \leftarrow w_{\mathcal{L}_d} \cdot \mathcal{L}_d + w_{\mathcal{L}_{r,w}} \cdot \mathcal{L}_{r,w} + w_{\mathcal{L}_s} \cdot \mathcal{L}_s$
  $\Theta_G = \Theta_G - \Gamma(\Delta_{\Theta_G} \mathcal{L})$
**end for**
**return** $\Theta_G, \Theta_D, \Theta_W, \Theta_R$

### C. Results

In Tab. I we display some of the images generated by the models, where IV means in vocubary words, OOV means out of vocabulary words, S refers to known style in training set and U refers to styles in the testing set.

| Type. | IV-S | IV-U | OOV-S | OOV-U |
|---|---|---|---|---|
| $\mathcal{L}_w$ |  |  |  |  |
| $\mathcal{L}_{w,c}$ |  |  |  |  |
| $\mathcal{L}_{w,t}$ |  |  |  |  |
| $\mathcal{L}_{w,c,t}$ |  |  |  |  |
| $\mathcal{L}_c$ |  |  |  |  |
| $\mathcal{L}_t$ |  |  |  |  |
| $\mathcal{L}_{c,t}$ |  |  |  |  |

TABLE I
IMAGES GENERATED BY DIFFERENT MODELS

### D. Quantitative Evaluation

[6] uses the FID as a quantitative metric. They acknowledge that it is not ideally designed for handwritten images, it was used for comparison purposes.

Tab. II shows the FID for the models. Similar to [6] we notice that the OOV-S and OOV-U models perform worse than the IV-S and IV-U models. However, incorporating the triplet loss with the cycle consistency the model performance is better than just using the writer identifier. Removing the writer identifier caused the models to suffer from mode collapse. The writer classifier is structurally at advantage to to utilize the style embedding, hence removing it causes the model to under-perform. However, adding both the triplet loss and cycle consistency gave us the best results, but some of the resulting images had artifacts, yet the style is maintained more compared to the baseline model.

| Model | IV-S | IV-U | OOV-S | OOV-U |
|---|---|---|---|---|
| $\mathcal{L}_{d,r}$+$\mathcal{L}_w$ | 120.07 | 124.30 | 125.87 | 130.68 |
| $\mathcal{L}_{d,r}$+$\mathcal{L}_w$+$\mathcal{L}_t$ | 119.72 | 122.55 | 123.20 | 128.39 |
| $\mathcal{L}_{d,r}$+$\mathcal{L}_w$+$\mathcal{L}_c$ | 119.02 | 121.31 | 124.11 | 129.92 |
| $\mathcal{L}_{d,r}$+$\mathcal{L}_w$+$\mathcal{L}_t$+$\mathcal{L}_c$ | 116.03 | 120.45 | 124.94 | 129.34 |
| $\mathcal{L}_{d,r}$ | 138.80 | 143.53 | 145.07 | 146.16 |
| $\mathcal{L}_{d,r}$+$\mathcal{L}_t$ | 135.02 | 140.22 | 143.84 | 144.13 |
| $\mathcal{L}_{d,r}$+$\mathcal{L}_c$ | 136.23 | 139.12 | 140.08 | 142.56 |
| $\mathcal{L}_{d,r}$+$\mathcal{L}_t$+$\mathcal{L}_c$ | 130.74 | 135.95 | 137.33 | 140.01 |

TABLE II
FID SCORES FOR THE MODELS, WHERE THE FID FOR REAL IMAGES IS 90.43.

### E. Qualitative Evaluation

*1) Setup:* We perform two experiments, the first one has two images, one real and one fake, where users have to choose which one is real and which one is fake. This imitates the discriminator model. We set five of these questions.

For the second experiment we give raters ten images to classify as real or fake, where five images are real and five are synthetic.

The raters had to rate images from three models: the original model and two of our best models, i.e., the one with the writer identifier, consistency and the triplet loss, and the one with the writer identifier and the cycle consistency. One image per

TABLE III
ORIGINAL MODEL: EXPERIMENT A

| Actual | Predicted | | |
|---|---|---|---|
| | Real | Fake | |
| Genuine | 29.81 | 20.19 | R: |
| Generated | 20.19 | 29.81 | FPR: |
| | P: | FOR | ACC: |

TABLE IV
ORIGINAL MODEL: EXPERIMENT B

| Actual | Predicted | | |
|---|---|---|---|
| | Real | Fake | |
| Genuine | 21.13 | 28.82 | R: |
| Generated | 28.49 | 21.51 | FPR: |
| | P: | FOR | ACC: |

writer was selected at random and no two words were repeated. We received 53 responses. The results are summarised by the confusion matrices in Tab. III-VIII.

From the results we see that the images all models are convincing, with the triplet loss + cycle consistency being the most convincing, followed by the cycle consistency, then the orignal GANWriting model.

## VI. CONCLUSION

In this paper we have shown that cycle consistency and triplet loss can work together in a few-short learning paradigm for handwriting synthesis.

TABLE V
CYCLE CONSISTENCY + TRIPLET LOSS: EXPERIMENT A

| Actual | Predicted | | |
|---|---|---|---|
| | Real | Fake | |
| Genuine | 24.15 | 25.85 | R: |
| Generated | 28.11 | 21.89 | FPR: |
| | P: | FOR | ACC: |

TABLE VI
CYCLE CONSISTENCY + TRIPLET LOSS: EXPERIMENT B

| Actual | Predicted | | |
|---|---|---|---|
| | Real | Fake | |
| Genuine | 29.25 | 20.75 | R: |
| Generated | 22.45 | 27.55 | FPR: |
| | P: | FOR | ACC: |

TABLE VII
CYCLE CONSISTENCY: EXPERIMENT A

| Actual | Predicted | | |
|---|---|---|---|
| | Real | Fake | |
| Genuine | 21.51 | 28.49 | R: |
| Generated | 27.17 | 29.83 | FPR: |
| | P: | FOR | ACC: |

TABLE VIII
CYCLE CONSISTENCY: EXPERIMENT B

| Actual | Predicted | | |
|---|---|---|---|
| | Real | Fake | |
| Genuine | 26.42 | 23.58 | R: |
| Generated | 23.40 | 26.60 | FPR: |
| | P: | FOR | ACC: |

## REFERENCES


[1] T. Haines, O. Aodha, and G. Brostow. "My text in yourhandwriting". ACM Transactions on Graphics, 35:1–18, 05 2016.

[2] K.M. Kumar, H. Kandala, and N.S. Reddy. "Synthesizing and imitating handwriting using deep recurrent neural networks andmixture density networks." In2018 9th International Conference on Computing,Communication and Networking Technologies (ICCCNT), pages 1–6. IEEE, 2018.

[3] A. Graves "Generating sequences with recurrent neural networks." arXiv preprint arXiv:1308.0850. 2013 Aug 4.

[4] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative Adversarial Networks", In Advances in neural information processing systems, pp. 2672-2680, 2014.

[5] L. Gonog and Y. Zhou. "A review: Generative adversarial net-works". In2019 14th IEEE Conference on Industrial Electronics and Applications(ICIEA), pages 505–510, 2019.

[6] L. Kang, P. Riba, Y. Wang, M. Rusiñol, A. Fornés and M. Villegas, "GANwriting: Content-Conditioned Generation of Styled Hand-written Word Images", Springer International Publishing, pages= 273–289, 2020.

[7] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Z. Yuhui. "Recent progress on generative adversarial networks (gans): A sur- vey". IEEE Access, PP:1–1, 03 2019.

[8] M. Mirza and S. Osindero. "Conditional GenerativeAdversarial Nets", ArXiv, 2014.

[9] M. Arjovsky, S. Chintala, and L. Bottou. "Wasserstein generative adversarial networks." In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 214–223, International Con- vention Centre, Sydney, Australia, 2017.

[10] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks." arXiv preprint arXiv:1701.00160, 2016.

[11] M., Luke, B. Poole, D. Pfau, and J. Sohl-Dickstein. "Unrolled generative adversarial networks." arXiv preprint arXiv:1611.02163, 2016.

[12] M. Arjovsky, and L. Bottou. "Towards principled methods for training generative adversarial networks." arXiv preprint arXiv:1701.04862, 2017.

[13] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann "Stabilizing training of generative adversarial networks through regularization". In Advances in neural information processing systems (pp. 2018-2028), 2017.

[14] A. Borji. "Pros and Cons of GAN Evaluation Measures." Comput. Vis. Image Underst. 179: 41-65, 2019.

[15] T. Salimans, I. Goodfellow, W. Zaremba, V. Che-ung, A. Radford, and X. Chen. "Improved techniques for training gans". In Advances in neural information processing systems, pages 2234–2242, 2016.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, "Imagenet: Alarge-scale hierarchical image database', in: Computer Vision and PatternRecognition, 2009. CVPR 2009. IEEE Conference on, IEEE, pp.248–255, 2009.

[18] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. "GANs Trained by a Two Time-Scale Update RuleConverge to a Local Nash Equilibrium". NIPS, 2017.

[19] A Purohit and S. S. Chauhan. A literaturesurvey on handwritten character recognition.IJCSIT) International Journal ofComputer Science and Information Technologies, 7(1):1–5, 2016.

[20] M. Agarwal, Shalika, V. Tomar, and P. Gupta. Handwritten character recognition using neural network and tensor flow. InIn-ternational Journal of Innovative Technology and Exploring Engineering (IJITEE),volume 8, pages 1445–1448, April 2019.

[21] K. M. Kumar, H. Kandala, and N S. Reddy. Syn-thesizing and imitating handwriting using deep recurrent neural networks andmixture density networks. In2018 9th International Conference on Comput-

ing,Communication and Networking Technologies (ICCCNT), pages 1–6. IEEE, 2018.

[22] A. Priya, S. Mishra, S. Raj, S. Mandal, and S. Datta. Online andoffline character recognition: A survey. In2016 International Conference on Com-munication and Signal Processing (ICCSP), pages 0967–0970, 2016.

[23] A. Rehman, S. Naz, and M. I. Razzak. Writer identification using machine learning approaches: a comprehensive re-view.Multimedia Tools and Applications, 78:10889–10931, 2018.

[24] I. Siddiqi and N. Vincent. Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features. Pattern Recognition, 43(11):3853 – 3865,.

[25] S and M. I. Sumam. A survey on writer identification schemes. International Journal of Computer Applications, 26:23–33, 2011.

[26] B. Amos, B. Ludwiczuk, and M. Satyanarayanan. "Openface: A general-purpose face recognition library with mobile applications." CMU School of Computer Science 6, no. 2, 2016.

[27] A. Hermans, B. Lucas and B. Leibe. "In Defense of the Triplet Loss for Person Re-Identification." ArXiv abs/1703.07737, 2017.

[28] J. Zhu, A. Park, T. Isola, and P. Efros Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. 2242-2251. 10.1109/ICCV.2017.244, 2017.

[29] R. Elanwar. "The state of the art in handwriting synthesis." In 2nd International Conference on New Paradigms in Electronics and information Technology (peit'013), Luxor, Egypt, 2013.

[30] Y. Elarian, R Abdel-Aal, I. Ahmad, M. Parvez, and A. Zidouri. "Handwriting synthesis: Classifications and techniques." INTERNATIONAL JOURNAL OF DOCUMENT ANALYSIS AND RECOGNITION, 17, 09 2014.

[31] J. Memon, M. Sami, R. A. Khan and M. Uddin, "Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)," in IEEE Access, vol. 8, pp. 142642-142668, 2020, doi: 10.1109/ACCESS.2020.3012542.

[32] A. Ghosh, B. Bhattacharya, and S. B. RoyChowdhury. "Handwriting profiling using generative adversarial networks." In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17,page 4927–4928. AAAI Press, 2017.

[33] B. Ji and T. Chen. "Generative adversarial network for hand-written text." ArXiv, abs/1907.11845, 2019

[34] C. G. Turhan and H. S. Bilge. Variational autoencoded compo-sitional pattern generative adversarial network for handwritten super resolution-image generation. In2018 3rd International Conference on Computer Science andEngineering (UBMK), pages 564–568, Sep. 2018.

[35] D. P. Kingma, and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).

[36] T. Salimans, D. K., and M. Welling. "Markov chain monte carlo and variational inference: Bridging the gap." In International Conference on Machine Learning, pp. 1218-1226. 2015.

[37] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. "Autoencoding beyond pixels using a learned similarity metric." In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16). JMLR.org, 1558–1566, 2016.

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

[39] H, David. "Generating Large Images from Latent Vectors." Unpublished, 2016.

[40] E. Aksan, F. Pece, and O. Hilliges. "Deepwriting: Making digital ink editable via deep generative modeling." Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18, 2018.

[41] E. Aksan and O. Hilliges. "Stcn: Stochastic temporalconvolutional networks." arXiv preprint arXiv:1902.06568, 2019.

[42] T. Haines, O. Aodha, and G. Brostow. "My text in yourhandwriting". ACM Transactions on Graphics, 35:1–18, 05 2016.

[43] E. Alonso, B. Moysset, and R. Messina. "Adversarial generation of handwritten text images conditioned on sequences." In2019 International Conference on Document Analysis and Recognition (ICDAR), pages 481–486, Sep. 2019.

[44] M. Mayr, M. Stumpf, A. Nicolaou, M. Seuret, A. Maier and V. Christlein. "Spatio-Temporal Handwriting Imitation." ArXiv abs/2003.10593, 2020.

[45] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. "Image-to-image translation with conditional adversarial networks". IEEE Conferenceon Computer Vision and Pattern Recognition (CVPR), 2017.

[46] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition." In Proceedings of the International Conference on Learning Representations, 2015.

[47] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, 2015.

[48] X. Huang, M. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In Proceedings of the European Conference on Computer Vision, 2018.

[49] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE International Conference on Computer Vision, 2017.

[50] L. Kang, J. I. Toledo, P. Riba, M. Villegas, A. Fornés, and M. Rusiñol. Convolve, attend and spell: An attention-based sequence-to-sequence model for handwritten word recognition. In Proceedings of the German Conference on Pattern Recognition, 2018.

[51] U-V Marti and Horst Bunke. The IAM-database: an English sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition, 5(1):39–46, 2002.

[52] S. Bird, E. Klein, and E. Loper. Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media, Inc., 2009.