Handwriting Transformers

Ankan Kumar Bhunia[1] Salman Khan[1,2] Hisham Cholakkal[1] Rao Muhammad Anwer[1]
Fahad Shahbaz Khan[1,3] Mubarak Shah[4]
[1]Mohamed bin Zayed University of AI, UAE [2]Australian National University, Australia
[3]Linköping University, Sweden [4]University of Central Florida, USA

Abstract

We propose a novel transformer-based styled handwritten text image generation approach, HWT, that strives to learn both style-content entanglement as well as global and local style patterns. The proposed HWT captures the long and short range relationships within the style examples through a self-attention mechanism, thereby encoding both global and local style patterns. Further, the proposed transformer-based HWT comprises an encoder-decoder attention that enables style-content entanglement by gathering the style features of each query character. To the best of our knowledge, we are the first to introduce a transformer-based network for styled handwritten text generation.

Our proposed HWT generates realistic styled handwritten text images and outperforms the state-of-the-art demonstrated through extensive qualitative, quantitative and human-based evaluations. The proposed HWT can handle arbitrary length of text and any desired writing style in a few-shot setting. Further, our HWT generalizes well to the challenging scenario where both words and writing style are unseen during training, generating realistic styled handwritten text images. Code is available at: https://github.com/ankanbhunia/Handwriting-Transformers

1. Introduction

Generating realistic synthetic handwritten text images, from typed text, that is versatile in terms of both writing style and lexicon is a challenging problem. Automatic handwritten text generation can be beneficial for people having disabilities or injuries that prevent them from writing, translating a note or a memo from one language to another by adapting an author's writing style or gathering additional data for training deep learning-based handwritten text recognition models. Here, we investigate the problem of realistic handwritten text generation of unconstrained text sequences with arbitrary length and diverse calligraphic attributes representing writing styles of a writer.

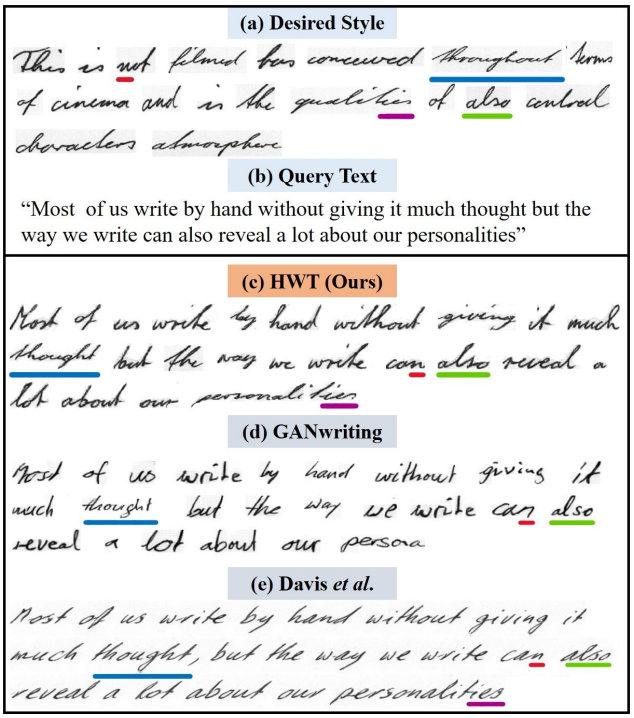


Figure 1: Comparison of HWT **(c)** with GANwriting [14] **(d)** and Davis *et al*. [5] **(e)** in imitating the desired unseen writing style **(a)** for given query text **(b)**. While [14, 5] capture global writing styles (*e.g*., slant), they struggle to imitate local style patterns (*e.g*., character style, ligatures). HWT (c) imitates both global and local styles, leading to a more realistic styled handwritten text image generation. For instance, style of '`n`' (red line) appearing in (a) is mimicked by HWT, for a different word including same character '`n`'. Similarly, a group of characters in '`thought`' and '`personalities`' (blue and magenta lines) are styled in a way that matches with words ('`throughout`' and '`qualities`') sharing some common characters in (a). Furthermore, HWT preserves cursive patterns and connectivity of all characters in word '`also`' (green line).

Generative Adversarial Networks (GANs) [8] have been investigated for offline handwritten text image generation

[4, 3, 14, 7, 5]. These methods strive to directly synthesize text images by using offline handwriting images during training, thereby extracting useful features, such as writing appearance (*e.g*., ink width, writing slant) and line thickness changes. Alonso *et al*. [3] propose a generative architecture that is conditioned on input content strings, thereby not restricted to a particular pre-defined vocabulary. However, their approach is trained on isolated fixed-sized word images and struggles to produce high quality arbitrarily long text along with suffering from style collapse. Fogel *et al*. [7] introduce a ScrabbleGAN approach, where the generated image width is made proportional to the input text length. ScrabbleGAN is shown to achieve impressive results with respect to the content. However, both [3, 7] do not adapt to a specific author's writing style.

Recently, GAN-based approaches [5, 14] have been introduced for the problem of styled handwritten text image generation. These methods take into account both content *and* style, when generating offline handwritten text images. Davis *et al*. [5] propose an approach based on StyleGAN [15] and learn generated handwriting image width based on style and input text. The GANwriting framework [14] conditions handwritten text generation process to both textual content and style features in a few-shot setup.

In this work, we distinguish two key issues that impede the quality of styled handwritten text image generation in the existing GAN-based methods [5, 14]. First, both style and content are loosely connected as their representative features are processed separately and later concatenated. While such a scheme enables entanglement between style and content at the word/line-level, it does not explicitly enforce style-content entanglement at the character-level. Second, although these approaches capture global writing style (*e.g*., ink width, slant), they do not explicitly encode local style patterns (*e.g*., character style, ligatures). As a result of these issues, they struggle to accurately imitate local calligraphic style patterns from reference style examples (see Fig. 1). Here, we look into an alternative approach that addresses both these issues in a single generative architecture.

1.1. Contributions

We introduce a new styled handwritten text generation approach built upon transformers, termed Handwriting Transformers (HWT), that comprises an encoder-decoder network. The encoder network utilizes a multi-headed self-attention mechanism to generate a self-attentive style feature sequence of a writer. This feature sequence is then input to the decoder network that consists of multi-headed self- and encoder-decoder attention to generate character-specific style attributes, given a set of query word strings. Consequently, the resulting output is fed to a convolutional decoder to generate final styled handwritten text image. Moreover, we improve the style consistency of the generated text by constraining the decoder output through a loss term whose objective is to re-generate style feature sequence of a writer at the encoder.

Our HWT imitates the style of a writer for a given query content through self- and encoder-decoder attention that emphasizes relevant self-attentive style features with respect to each character in that query. This enables us to capture style-content entanglement at the character-level. Furthermore, the self-attentive style feature sequence generated by our encoder captures both the global (*e.g*., ink width, slant) *and* local styles (*e.g*., character style, ligatures) of a writer within the feature sequence.

We validate our proposed HWT by conducting extensive qualitative, quantitative and human-based evaluations. In the human-based evaluation, our proposed HWT was preferred 81% of the time over recent styled handwritten text generation methods [5, 14], achieving human plausibility in terms of the writing style mimicry. Following GANwriting [14], we evaluate our HWT on all the four settings on the IAM handwriting dataset. On the extreme setting of out-of-vocabulary and unseen styles (OOV-U), where both query words and writing styles are never seen during training, the proposed HWT outperforms GANwriting [14] with an absolute gain of 16.5 in terms of Frèchet Inception Distance (FID) thereby demonstrating our generalization capabilities. Further, our qualitative analysis suggest that HWT performs favorably against existing works, generating realistic styled handwritten text images (see Fig. 1).

2. Related Work

Deep learning-based handwritten text generation approaches can be roughly divided into stroke-based online and image-based offline methods. Online handwritten text generation methods [9, 2] typically require temporal data acquired from stroke-by-stroke recording of real handwritten examples (vector form) using a digital stylus pen. On the other hand, recent generative offline handwritten text generation methods [4, 3, 14, 7] aim to directly generate text by performing training on offline handwriting images.

Graves [9] proposes an approach based on Recurrent Neural Network (RNN) with Long-Term Memory (LSTM) cells, which enables predicting future stroke points from previous pen positions and an input text. Aksan *et al*. [4] propose a method based on conditional Variational RNN (VRNN), where the input is split into two separate latent variables to represent content and style. However, their approach tends to average out particular styles across writers, thereby reducing details [17]. In a subsequent work [1], the VRNN module is substituted by Stochastic Temporal CNNs which is shown to provide more consistent generation of handwriting. Kotani *et al*. [17] propose an online handwriting stroke representation approach to represent latent style information by encoding writer-, character- and writer-

character-specific style changes within an RNN model.

Other than sequential methods, several recent works have investigated offline handwritten text image generation using GANs. Haines *et al*. [11] introduce an approach to generate new text in a distinct style inferred from source images. Their model requires a certain degree of human intervention during character segmentation and is limited to generating characters that are in the source images. The work of [4] utilize CycleGAN [24] to synthesize images of isolated handwritten characters of Chinese language. Alonso *et al*. [3] propose an approach, where handwritten text generation is conditioned by character sequences. However, their approach suffers from style collapse hindering the diversity of synthesized images. Fogel *et al*. [7] propose an approach, called ScrabbleGAN, that synthesizes handwritten word using a fully convolutional architecture. Here, the characters generated have similar receptive field width. A conversion model is introduced by [20] that approximates online handwriting from offline samples followed by using style transfer technique to the online data. This approach relies on conversion model's performance.

Few recent GAN-based works [5, 14] investigate the problem of offline styled handwritten text image generation. Davis *et al*. [5] propose an approach, where handwritten text generation is conditioned on both text and style, capturing global handwriting style variations. Kang *et al*. [14] propose a method, called GANwriting, that conditions text generation on extracting style features in a few-shot setup and textual content of a predefined fixed length.

Our Approach: Similar to GANwriting [14], we also investigate the problem of styled handwritten text generation in a few-shot setting, where a limited number of style examples are available for each writer. Different from GANwriting, our approach possesses the flexibility to generate styled text of arbitrary length. In addition, existing works [5, 14] only capture style-content entanglement at the word/line-level. In contrast, our transformer-based approach enables style-content entanglement both at the word and character-level. While [5, 14] focuses on capturing the writing style at the global level, the proposed method strives to imitate both global and local writing style.

3. Proposed Approach

Motivation: To motivate our proposed HWT method, we first distinguish two desirable characteristics to be considered when designing an approach for styled handwritten text generation with varying length and any desired style in a few-shot setting, without using character-level annotation.

Style-Content Entanglement: As discussed earlier, both style and content are loosely connected in recently introduced GAN-based works [14, 5] with separate processing of style and content features, which are later concatenated. Such a scheme does not explicitly encode style-content entanglement at the character-level. Moreover, there are separate components for style, content modeling followed by a generator for decoding stylized outputs. In addition to style-content entanglement at word/line level, an entanglement between style and content at the character-level is expected to aid in imitating the character-specific writing style along with generalizing to out-of-vocabulary content. Further, such a tight integration between style and content leads to a cohesive architecture design.

Global and Local Style Imitation: While the previous requisite focuses on connecting style and content, the second desirable characteristic aims at modeling both the global as well as local style features for a given calligraphic style. Recent generative methods for styled handwritten text generation [14, 5] typically capture the writing style at the global level (*e.g*., ink width, slant). However, the local style patterns (*e.g*., character style, ligatures) are not explicitly taken into account while imitating the style of a given writer. We argue that both global *and* local style patterns are desired to be imitated for accurate styled text image generation.

3.1. Approach Overview

Problem Formulation: We aim to learn the complex handwriting style characteristics of a particular writer $i \in \mathcal{W}$, where $\mathcal{W}$ includes a total of M writers. We are given a set of P handwritten word images, $\boldsymbol{X}_i^s = \{\boldsymbol{x}_{ij}\}_{j=1}^{P}$, as few-shot calligraphic style examples of each writer. The superscript 's' in $\boldsymbol{X}_i^s$ denotes use of the set as a source of handwriting style which is transferred to the target images $\tilde{\boldsymbol{X}}_i^t$ with new textual content but consistent style properties. The textual content is represented as a set of input query word strings $\mathcal{A} = \{\boldsymbol{a}_j\}_{j=1}^{Q}$, where each word string $\boldsymbol{a}_j$ comprises an arbitrary number of characters from permitted characters set $\mathcal{C}$. The set $\mathcal{C}$ includes alphabets, numerical digits and punctuation marks *etc*. Given a query text string $\boldsymbol{a}_j \in \mathcal{A}$ from an unconstrained set of vocabulary and $\boldsymbol{X}_i^s$, our model strives to generate new images $\tilde{\boldsymbol{X}}_i^t$ with the same text $\boldsymbol{a}_j$ in the writing style of a desired writer i.

Overall Architecture: Fig. 2 presents an overview of our proposed HWT approach, where a conditional generator G_θ synthesizes handwritten text images, a discriminator D_ψ ensures realistic generation of handwriting styles, a recognizer R_ϕ aids in textual content preservation, and a style classifier S_η ensures satisfactory transfer of the calligraphic styles. The focus of our design is the introduction of a transformer-based generative network for unconstrained styled handwritten text image generation. Our generator G_θ is designed in consideration to the desirable characteristics listed earlier leveraging the impressive learning capabilities of transformer models. To meticulously imitate a handwriting style, a model is desired to learn style-content entanglement as well as global and local style patterns.

To this end, we introduce a transformer-based handwrit-

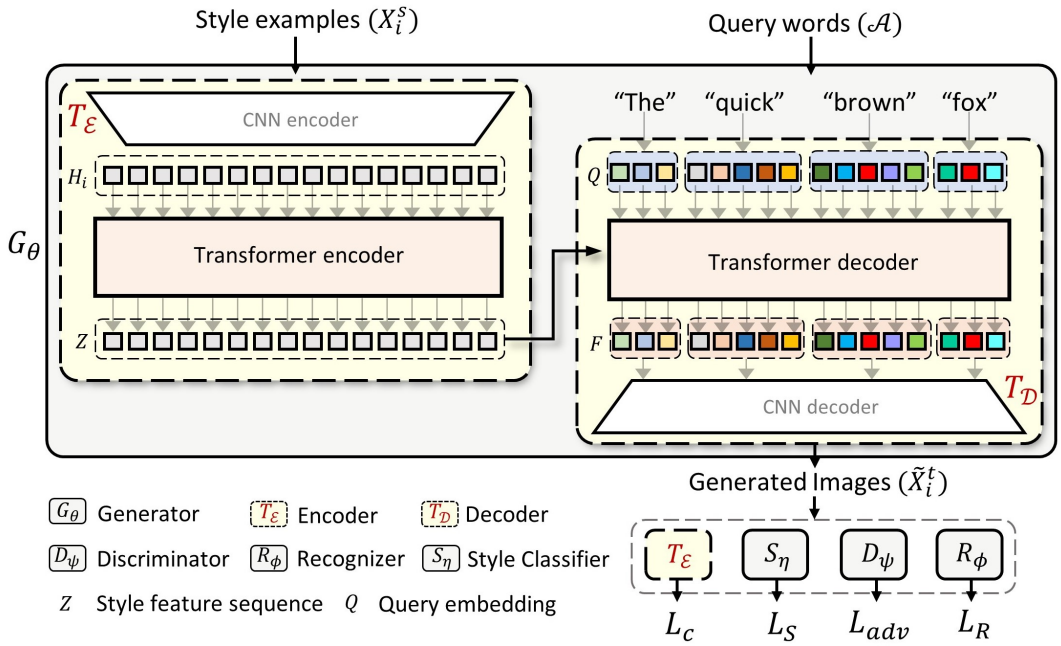


Figure 2: Overall architecture of our Handwriting Transformers (HWT) to generate styled handwritten text images $\tilde{\boldsymbol{X}}_i^t$. HWT comprises a conditional generator having an encoder $T_\mathcal{E}$ and a decoder network $T_\mathcal{D}$. Both the encoder and decoder networks constitute a hybrid convolution and multi-head self-attention design, which combines the strengths of CNN and transformer-based models *i.e.*, highly expressive relationship modeling while working with limited handwriting style example images. Resultantly, our design seamlessly achieves style-content entanglement that encodes relationships between textual content and writer's style along with learning both global and local style patterns for given inputs ($\boldsymbol{X}_i^s$ and $\mathcal{A}$).

ing generation model, which enables us to capture the long and short range contextual relationships within the style examples $\boldsymbol{X}_i^s$ by utilizing a self-attention mechanism. In this way, both the global and local style patterns are encoded. Additionally, our transformer-based model comprises an encoder-decoder attention that allows style-content entanglement by inferring the style representation for each query character. A direct applicability of transformer-based design is infeasible in our few-shot setting due to its large data requirements and quadratic complexity. To circumvent this issue, our proposed architecture design utilizes the expressivity of a transformer within the CNN feature space.

The main idea of the proposed HWT method is simple but effective. A transformer-based encoder $T_\mathcal{E}$ is first used to model self-attentive style context that is later used by a decoder $T_\mathcal{D}$ to generate query text in a specific writer's style. We define learnable embedding vector $\boldsymbol{q}_\mathtt{c} \in \mathbb{R}^{512}$ for each character $\mathtt{c}$ of the permissible character set $\mathcal{C}$. For example, we represent the query word '$\mathtt{deep}$' as a sequence of its respective character embeddings $\boldsymbol{Q}_\mathtt{deep} = \{\boldsymbol{q}_\mathtt{d} \dots \boldsymbol{q}_\mathtt{p}\}$. We refer them as query embeddings. Such a character-wise representation of the query words and the transformer-based sequence processing helps our model to generate handwritten words of variable length, and also qualifies it to produce out-of-vocabulary words more efficiently. Moreover, it avoids averaging out individual character-specific styles in order to maintain the overall (global and local) writing style. The character-wise style interpolation and transfer is ensured by the self- and encoder-decoder attention in the transformer module that infers the style representation of each query character based on a set of handwritten samples provided as input. We describe the proposed generative architecture in Sec. 3.2 and the loss objectives in Sec. 3.3.

3.2. Generative Network

The generator G_θ includes two main components: an encoder network $T_\mathcal{E} : \boldsymbol{X}_i^s \rightarrow \boldsymbol{Z}$ and a decoder network $T_\mathcal{D} : (\boldsymbol{Z}, \mathcal{A}) \rightarrow \tilde{\boldsymbol{X}}_i^t$. The encoder produces a sequence of feature embeddings $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$ (termed as style feature sequence) from a given set of style examples $\boldsymbol{X}_i^s$. The decoder takes $\boldsymbol{Z}$ as an input and converts the input word strings $\boldsymbol{a}_j \in \mathcal{A}$ to realistic handwritten images $\tilde{\boldsymbol{X}}_i^t$ with same style as the given examples $\boldsymbol{X}_i^s$ of a writer i. Both the encoder and decoder networks constitute a *hybrid* design based on convolution and multi-head self-attention networks. This design choice combines the strengths of CNNs and transformer models *i.e.*, highly expressive relationship modeling while working with limited handwriting images. Its worth mentioning that a CNN-only design would struggle to model long-term relations within sequences while an architecture based solely on transformer networks would demand large amount of data and longer training times [16].

Encoder $T_{\mathcal{E}}$. The encoder aims at modelling both global and local calligraphic style attributes (*i.e.*, slant, skew, character shapes, ligatures, ink widths *etc.*) from the style examples $\boldsymbol{X}_i^s$. Before feeding style images to the highly expressive transformer architecture, we need to represent the style examples as a sequence. A straightforward way would be to flatten the image pixels into a 1D vector [6]. However, given the quadratic complexity of transformer models and their large data requirements, we find this to be infeasible. Instead, we use a CNN backbone network to obtain sequences of convolutional features from the style images. First, we use a ResNet18 [12] model to generate lower-resolution activation maps $\boldsymbol{h}_{ij} \in \mathbb{R}^{h \times w \times d}$ for each style image $\boldsymbol{x}_{ij}$. Then, we flatten the spatial dimension of $\boldsymbol{h}_{ij}$ to obtain a sequence of feature maps of size $n \times d$, where $n = h \times w$. Each vector in the feature sequence represents a region in the original image and can be considered as the image descriptor for that particular region. After that, we concatenate the feature sequence vectors extracted from all style images together to obtain a single tensor $\boldsymbol{H}_i \in \mathbb{R}^{N \times d}$, where $N = n \times P$.

The next step includes modeling the global and local compositions between all entities of the obtained feature sequence $\boldsymbol{Z}$. A transformer-based encoder is employed for that purpose. The encoder has L layers, where each layer has a standard architecture that consists of a multi-headed self-attention module and a Multi-layer Perceptron (MLP) block. At each layer l, the multi-headed self-attention maps the input sequence from the previous layer $\boldsymbol{H}^{l-1}$ into a triplet (key $\boldsymbol{K}$, query $\boldsymbol{Q}$, value $\boldsymbol{V}$) of intermediate representations given by,

$$\boldsymbol{Q} = \boldsymbol{H}^{l-1}\boldsymbol{W}^{Q}, \boldsymbol{K} = \boldsymbol{H}^{l-1}\mathbf{W}^{K}, \boldsymbol{V} = \boldsymbol{H}^{l-1}\mathbf{W}^{V},$$

where $\mathbf{W}^{Q} \in \mathbb{R}^{N \times d_q}$, $\mathbf{W}^{K} \in \mathbb{R}^{N \times d_k}$ and $\mathbf{W}^{V} \in \mathbb{R}^{N \times d_v}$ are the learnable weight matrix for query, key and value respectively. For each head, the process is represented as,

$$\boldsymbol{O}^{j} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{T}}{\sqrt{d_k}}\right)\boldsymbol{V} \in \mathbb{R}^{N \times d_v}, \quad j \in \{1, .., J\}. \tag{1}$$

The concatenation of all J head outputs $\boldsymbol{O} = [\boldsymbol{O}^1, \ldots, \boldsymbol{O}^J]$ is then fed through an MLP layer to obtain the output feature sequence $\boldsymbol{H}^l$ for the layer l. This update procedure is repeated for a total of L layers, resulting in the final feature sequence $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$. To retain information regarding the order of input sequences being supplied, we add fixed positional encodings [23] to the input of each attention layer.

Decoder $T_{\mathcal{D}}$. The initial stage in the decoder uses the standard architecture of the transformer that consists of multi-headed self- and encoder-decoder attention mechanisms. Unlike the self-attention, the encoder-decoder attention derives the key and value vectors from the output of the encoder, whereas the query vectors come from the decoder layer itself. For an m_j character word $\boldsymbol{a}_j \in \mathcal{A}$ (length

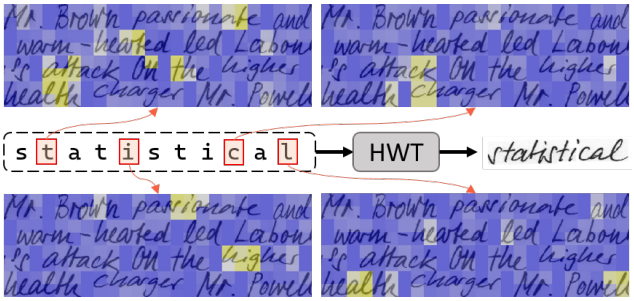


Figure 3: Visualization of encoder-decoder attention maps at the last layer of the transformer decoder. The attention maps are computed for each character in the query word ('statistical') which are then mapped to spatial regions (heat maps) in the example style images. Here, heat maps corresponding to the four different query characters 't', 'i', 'c' and 'l' are shown. For instance, the top-left attention map corresponding to the character 't', highlights multiple image regions containing the character 't'.

m_j being variable depending on the word), the query embedding $\boldsymbol{Q}_{\boldsymbol{a}_j} = \{\boldsymbol{q}_{c_k}\}_{k=1}^{m_j}$ is used as a learnt positional encoding to each attention layer of the decoder. Intuitively, each query embedding learns to look up regions of interest in the style images to infer the style attributes of all query characters (see Fig. 3). Over multiple consecutive decoding layers, these output embeddings accumulate style information, producing a final output $\boldsymbol{F}_{a_j} = \{\boldsymbol{f}_{c_k}\}_{k=1}^{m_j} \in \mathbb{R}^{m_j \times d}$. We process the entire query embedding in parallel at each decoder layer. We add a randomly sampled noise vector $\mathcal{N}(0, 1)$ to the output $\boldsymbol{F}_{\boldsymbol{a}_j}$ in order to model the natural variation of individual handwriting. For an m-character word, we concatenate these m_j embedding vectors and pass them through a linear layer, resulting in an $m_j \times 8192$ matrix. After reshaping it to a dimension of $512 \times 4 \times 4m_j$, we pass it through a CNN decoder having four residual blocks followed by a $\tanh$ activation layer to obtain final output images (styled hand written text images).

3.3. Training and Loss Objectives

Our training algorithm follows the traditional GAN paradigm, where a discriminator network D_ψ is employed to tell apart the samples generated from generator G_θ from the real ones. As the generated word images are of varying width, the proposed discriminator D_ψ is also designed to be convolutional in nature. We use the hinge version of the adversarial loss [18] defined as,

$$\begin{aligned} L_{adv} =& \mathbb{E}\left[\max\left(1 - D_\psi(\boldsymbol{X}_i^s, 0)\right)\right] + \\ & \mathbb{E}\left[\max\left(1 + D_\psi(G_\theta(\boldsymbol{X}_i^s, \mathcal{A})), 0\right)\right]. \end{aligned} \tag{2}$$

While D_ψ promotes real-looking images, it does not preserve the content or the calligraphic styles. To preserve the textual content in the generated samples we use a handwritten recognizer network R_ϕ that examines whether the gen-

erated samples are actually real text. The recognizer R_ϕ is inspired by CRNN [21]. The CTC loss [10] is used to compare the recognizer output to the query words that were given as input to G_θ. Recognizer R_ϕ is only optimized with real, labelled, handwritten samples, but it is used to encourage G_θ to produce readable text with accurate content. The loss is defined as,

$$L_R = \mathbb{E}_{\boldsymbol{x}\sim\left\{\boldsymbol{X}_i^s, \tilde{\boldsymbol{X}}_i^t\right\}}\left[-\sum \log\left(p\left(y_r|R_\phi\left(\boldsymbol{x}\right)\right)\right)\right]. \quad (3)$$

Here, y_r is the transcription string of $\boldsymbol{x} \sim \left\{\boldsymbol{X}_i^s, \tilde{\boldsymbol{X}}_i^t\right\}$.

A style classifier network S_η is employed to guide the network G_θ in producing samples conditioned to a particular writing style. The network S_η attempts to predict the writer of a given handwritten image. The cross-entropy objective is applied as a loss function. S_η is trained only on the real samples using the loss given below,

$$L_S = \mathbb{E}_{x\sim\left\{\boldsymbol{X}_i^s, \tilde{\boldsymbol{X}}_i^t\right\}}\left[-\sum y_i log\left(S_\eta\left(\boldsymbol{x}\right)\right)\right]. \quad (4)$$

An important feature of our design is to utilize a cycle loss that ensures the encoded style features have cycle consistency. This loss function enforces the decoder to preserve the style information in the decoding process, such that the original style feature sequence can be reconstructed from the generated image. Given the generated word images $\tilde{\boldsymbol{X}}_i^t$, we use the encoder $T_\mathcal{E}$ to reconstruct the style feature sequence $\tilde{\boldsymbol{Z}}$. The cycle loss L_c minimizes the error between the style feature sequence $\boldsymbol{Z}$ and its reconstruction $\tilde{\boldsymbol{Z}}$ by means of a L_1 distance metric,

$$L_c = \mathbb{E}\left[\left\|T_\mathcal{E}(\boldsymbol{X}_i^s) - T_\mathcal{E}(\tilde{\boldsymbol{X}}_i^t)\right\|_1\right]. \quad (5)$$

The cycle loss imposes a regularization to the decoder for consistently imitating the writing style in the generated styled text images. Overall, we train our HWT model in an end-to-end manner with the following loss objective,

$$L_{total} = L_{adv} + L_S + L_R + L_c. \quad (6)$$

We observe balancing the gradients of the network S_η and R_ϕ is helpful in the training with our loss formulation. Following [3], we normalize the ∇S_η and ∇R_ϕ to have the same standard deviation (σ) as adversarial loss gradients,

$$\nabla S_\eta \leftarrow \alpha\left(\frac{\sigma_D}{\sigma_S}.\nabla S_\eta\right), \nabla R_\phi \leftarrow \alpha\left(\frac{\sigma_D}{\sigma_R}.\nabla R_\phi\right). \quad (7)$$

Here, α is a hyper-parameter that is fixed to 1 during training of our model.

4. Experiments

We perform extensive experiments[1] on IAM handwriting dataset [19]. It consists of 9862 text lines with around 62,857 English words, written by 500 different writers. For thorough evaluation, we reserve an exclusive subset of 160 writers for testing, while images from the remaining 340 writers are used for our model training. In all experiments, we resize images to a fixed height of 64 pixels, while maintaining the aspect ratio of original image. For training, we use $P = 15$ style example images, as in [14]. Both the transformer encoder and transformer decoder employ 3 attention layers ($L = 3$) and each attention layer applies multi-headed attention having 8 attention heads ($J = 8$). We set the embedding size d to 512. In all experiments, we train our model for 4k epochs with a batch size of 8 on a single V100 GPU. Adam optimizer is employed during training with a learning rate of 0.0002.

[1] **Additional experiments, including (i) quantitative comparison on CVL and RIMES datasets and (ii) handwritten text recognition (HTR) results, are presented in supplementary material.**

Table 1: **Comparison of the HWT with GANwriting [14] and Davis *et al.* [5]** in terms of FID scores computed between the generated text images and real text images of the IAM dataset. Our HWT performs favorably against [14, 5] in all four settings: In-Vocabulary words and seen style (IV-S), In-Vocabulary words and unseen style (IV-U), Out-of-vocabulary content and seen style (OOV-S) and Out-of-vocabulary content and unseen style (OOV-U). On the challenging setting of OOV-U, HWT achieves an absolute gain of 16.5 in FID score, compared to GANwriting [14].

	IV-S ↓	IV-U ↓	OOV-S ↓	OOV-U ↓
GANwriting [14]	120.07	124.30	125.87	130.68
Davis *et al.* [5]	118.56	128.75	127.11	136.67
HWT (Ours)	**106.97**	**108.84**	**109.45**	**114.10**

4.1. Styled Handwritten Text Generation

We first evaluate (Tab. 1) our approach for styled handwritten text image generation, where both style and content are desired to be imitated in the generated text image. Following [14], we use Frèchet Inception Distance (FID) [13] evaluation metric for comparison. The FID metric is measured by computing the distance between the Inception-v3 features extracted from generated and real samples for each writer and then averaging across all writers. We evaluate our HWT with GANwriting [14] and Davis *et al.* [5] in four different settings: In-Vocabulary words and seen styles (IV-S), In-Vocabulary words and unseen styles (IV-U), Out-of-Vocabulary words and seen styles (OOV-S), and Out-of-Vocabulary words and unseen styles (OOV-U). Among these settings, most challenging one is the OOV-U, where both words and writing styles are never seen during training. For OOV-S and OOV-U settings, we use a set of 400 words that are distinct from IAM dataset transcription, as in [14]. In all four settings, the transcriptions of real samples and generated samples are different. Tab. 1 shows that HWT performs favorably against both existing methods [14, 5].

Fig 4 presents the qualitative comparison of HWT with [14, 5] for styled handwritten text generation. We present

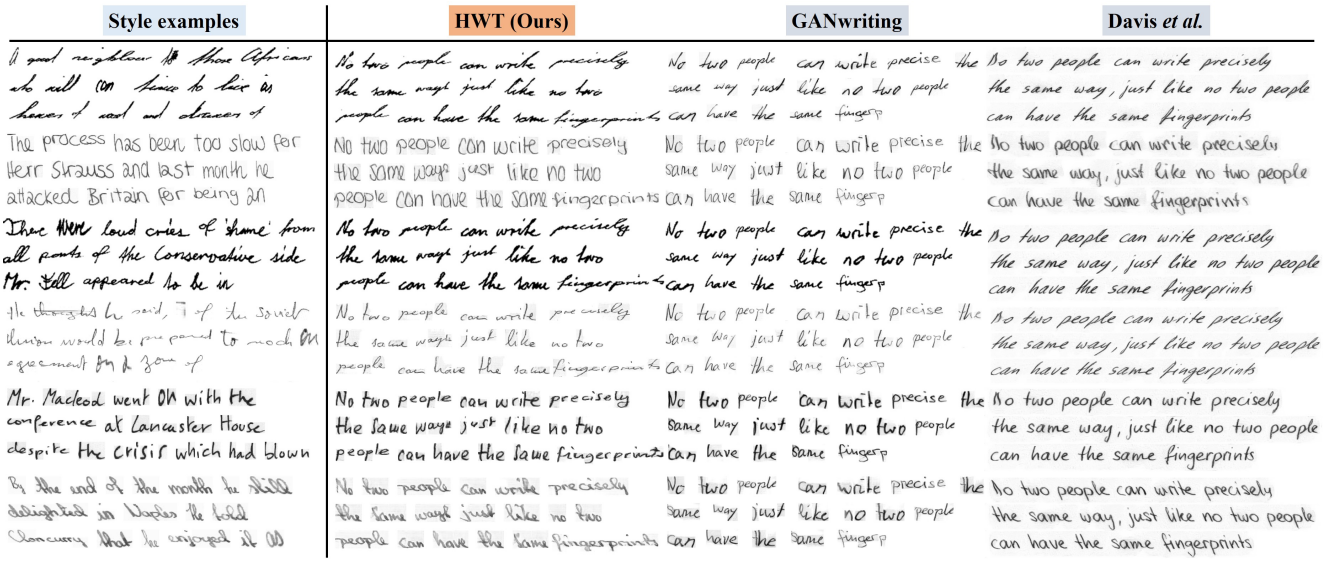


Figure 4: Qualitative comparison of our HWT (second column) with GANwriting [14] (third column) and Davis *et al*. [5] (fourth column). We use the same textual content *'No two people can write precisely the same way just like no two people can have the same fingerprints'* for all three methods. The first column shows the style examples from different writers. Davis *et al*. [5] captures the global style, *e.g*. slant, but struggles to mimic the character-specific style details. On the other hand, since GANwriting [14] is limited to a fixed length query words, it is unable to complete the provided textual content. Our HWT better mimics global and local style patterns, generating more realistic handwritten text images.

results for different writers, whose example style images are shown in the first column. For all the three methods, we use the same textual content. While Davis *et al*. [5] follows the leftward slant of the last style example from the top, their approach struggles to capture character-level styles and cursive patterns (*e.g*. see the word '`the`'). On the other hand, GANwriting [14] struggles to follow leftward slant of the last style example from the top and character-level styles. Our HWT better imitates both the global and local style patterns in these generated example text images.

4.2. Handwritten Text Generation

Here, we evaluate the quality of the handwritten text image generated by our HWT. For a fair comparison with the recently introduced ScrabbleGAN [7] and Davis *et al*. [5], we report our results in the *same* evaluation settings as used by [7, 5]. Tab. 2 presents the comparison with [7, 5] in terms of FID and geometric-score (GS). Our HWT achieves favourable performance, compared to both approaches in terms of both FID and GS scores. Different from Tab. 1, the results reported here in Tab. 2 indicates the quality of the generated images, compared with the real examples in the IAM dataset, while ignoring style imitation capabilities.

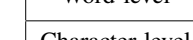
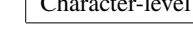
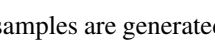
4.3. Ablation study

We perform multiple ablation studies on the IAM dataset to validate the impact of different components in our framework. Tab. 3 shows the impact of integrating transformer encoder (Enc), transformer decoder (Dec) and cycle loss (CL) to the baseline (Base). Our baseline neither uses transformer modules nor utilizes cycle loss. It only employs a CNN encoder to obtain style features, whereas the content features are extracted from the one-hot representation of query words. Both content and style features are passed through a CNN decoder to generate styled handwritten text images. While the baseline is able to generate realistic text images, it has a limited ability to mimic the given writer's style leading to inferior FID score (row 1). The introduction of the transformer encoder into the baseline (row 2) leads to an absolute gain of 5.6 in terms of FID score, highlighting the importance of our transformer-based self-attentive feature sequence in the generator encoder. We observe here that the generated sample still lacks details in

Table 2: **Handwritten text image generation quality comparison of our proposed HWT with ScrabbleGAN [7] and Davis *et al*. [5]** on the IAM dataset. Results are reported in terms of FID and GS by following the same evaluation settings, as in [7, 5]. Our HWT performs favorably against these methods in terms of both FID and GS. Best results are in bold.

	FID ↓	GS ↓
ScrabbleGAN [7]	20.72	2.56×10^{-2}
Davis *et al*. [5]	20.65	4.88×10^{-2}
HWT (Ours)	**19.40**	$\mathbf{1.01 \times 10^{-2}}$

Table 3: **Impact of integrating transformer encoder (Enc), transformer decoder (Dec) and cycle loss (CL) to the baseline (Base)** on the OOV-U settings of IAM dataset. Results are reported in terms of FID score. Best results are reported in bold. On right, we show the effect of each component when generating two example words '`freedom`' and '`precise`' mimicking two given writing styles.

	FID ↓	Style Example
		
Base	134.45	
Base + Enc	128.80	
Base + Dec	124.81	
Base + Enc + Dec	116.50	
Base + Enc + Dec + CL	**114.10**	

terms of character-specific style patterns. When integrating the transformer decoder into the baseline (row 3), we observe a significant gain of 9.6 in terms of FID score. Notably, we observe a significant improvement (17.9 in FID) when integrating both transformer encoder and decoder to the baseline (row 4). This indicates the importance of self- and encoder-decoder attention for achieving realistic styled handwritten text image generation. The performance is further improved by the introduction of cycle loss to our final HWT architecture (row 4).

As described earlier (Sec. 3.2), HWT strives for style-content entanglement at character-level by feeding query character embeddings to the transformer decoder network. Here, we evaluate the effect of character-level content encoding (conditioning) by replacing it with word-level conditioning. We obtain the word-level embeddings, by using an MLP that aims to obtain string representation of each query word. These embeddings are used as conditional input to the transformer decoder. Table 4 suggests that HWT benefits from character-level conditioning that ensures finer control of text style. The performance of word-level conditioning is limited to mimicking the global style, whereas our character-level approach ensures locally realistic as well as globally consistent style patterns.

4.4. Human Evaluation

Here, we present results of our two user studies on 100 human participants[2] to evaluate whether the proposed HWT achieves human plausibility in terms of the style mimicry. First, a *User preference study* compares styled text images generated by our method with GANwriting [14] and Davis *et al*. [5]. Second, a *User plausibility study* that evaluates the proximity of the synthesized samples generated by our method to the real samples. In both studies, synthesized

[2]Additional details are provided in supplementary material.

Table 4: **Comparison between word and character-level conditioning** on IAM dataset. Results are reported in terms of FID score. Our character-level conditioning performs favorably, compared to its word-level counterpart. Best results are reported in bold. On the right, we show the effect of word and character-level conditioning, when generating two example words '`symbols`' and '`same`' mimicking two given writing styles.

	FID ↓	Style Example
		
Word-level	126.87	
Character-level	**114.10**	

samples are generated using *unseen writing styles* of test set writers of IAM dataset, and for textual content we use sentences from Stanford Sentiment Treebank [22] dataset.

For *User preference study*, each participant is shown the real handwritten paragraph of a person and synthesized handwriting samples of that person using HWT, Davis *et al*. [5] and GANwriting [14], randomly organized. The participants were asked to mark the best method for mimicking the real handwriting style. In total, we have collected 1000 responses. The results of this study shows that our proposed HWT was preferred 81% of the time over the other two methods.

For *User plausibility study*, each participant is shown a person's actual handwriting, followed by six samples, where each of these samples is either genuine or synthesized handwriting of the same person. Participants are asked to identify whether a given handwritten sample is genuine or not (forged/synthesized) by looking at the examples of the person's real handwriting. Thus, each participant provides 60 responses, thereby we collect 6000 responses for 100 participants. For this study, only 48.1% of the images have been correctly classified, thereby showing a comparable performance to a random choice in a two-class problem.

5. Conclusion

We introduced a transformer-based styled handwritten text image generation approach, HWT, that comprises a conditional generator having an encoder-decoder network. Our HWT captures the long and short range contextual relationships within the writing style example through a self-attention mechanism, thereby encoding both global and local writing style patterns. In addition, HWT utilizes an encoder-decoder attention that enables style-content entanglement at the character-level by inferring the style representation for each query character. Qualitative, quantitative and human-based evaluations show that our HWT produces realistic styled handwritten text images with varying length and any desired writing style.

References

[1] Emre Aksan and Otmar Hilliges. Stcn: Stochastic temporal convolutional networks. *arXiv preprint arXiv:1902.06568*, 2019.

[2] Emre Aksan, Fabrizio Pece, and Otmar Hilliges. Deepwriting: Making digital ink editable via deep generative modeling. In *CHI*, pages 1–14, 2018.

[3] Eloi Alonso, Bastien Moysset, and Ronaldo Messina. Adversarial generation of handwritten text images conditioned on sequences. In *ICDAR*, pages 481–486. IEEE, 2019.

[4] Bo Chang, Qiong Zhang, Shenyi Pan, and Lili Meng. Generating handwritten chinese characters using cyclegan. In *WACV*, pages 199–207. IEEE, 2018.

[5] Brian Davis, Chris Tensmeyer, Brian Price, Curtis Wigington, Bryan Morse, and Rajiv Jain. Text and style conditioned gan for generation of offline handwriting lines. *BMVC*, 2020.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, and Roee Litman. Scrabblegan: semi-supervised varying length handwritten text generation. In *CVPR*, pages 4324–4333, 2020.

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[9] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[10] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*, pages 369–376, 2006.

[11] Tom SF Haines, Oisin Mac Aodha, and Gabriel J Brostow. My text in your handwriting. *TOG*, 35(3):1–18, 2016.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

[14] Lei Kang, Pau Riba, Yaxing Wang, Marçal Rusiñol, Alicia Fornés, and Mauricio Villegas. Ganwriting: Content-conditioned generation of styled handwritten word images. In *ECCV*, pages 273–289. Springer, 2020.

[15] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410, 2019.

[16] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*, 2021.

[17] Atsunobu Kotani, Stefanie Tellex, and James Tompkin. Generating handwriting via decoupled style descriptors. In *ECCV*, pages 764–780. Springer, 2020.

[18] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

[19] U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *IJDAR*, 5(1):39–46, 2002.

[20] Martin Mayr, Martin Stumpf, Anguelos Nikolaou, Mathias Seuret, Andreas Maier, and Vincent Christlein. Spatio-temporal handwriting imitation. *arXiv preprint arXiv:2003.10593*, 2020.

[21] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *PAMI*, 39(11):2298–2304, 2016.

[22] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[24] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2223–2232, 2017.