DiffusionPen: Towards Controlling the Style of Handwritten Text Generation

Konstantina Nikolaidou[1(✉)], George Retsinas[2], Giorgos Sfikas[3], and Marcus Liwicki[1]

[1] Luleå University of Technology, Luleå, Sweden
{konstantina.nikolaidou,marcus.liwicki}@ltu.se
[2] National Technical University of Athens, Athens, Greece
gretsinas@central.ntua.gr
[3] University of West Attica, Athens, Greece
gsfikas@uniwa.gr

Abstract. Handwritten Text Generation (HTG) conditioned on text and style is a challenging task due to the variability of inter-user characteristics and the unlimited combinations of characters that form new words unseen during training. Diffusion Models have recently shown promising results in HTG but still remain under-explored. We present DiffusionPen (DiffPen), a 5-shot style handwritten text generation approach based on Latent Diffusion Models. By utilizing a hybrid style extractor that combines metric learning and classification, our approach manages to capture both textual and stylistic characteristics of seen and unseen words and styles, generating realistic handwritten samples. Moreover, we explore several variation strategies of the data with multi-style mixtures and noisy embeddings, enhancing the robustness and diversity of the generated data. Extensive experiments using IAM offline handwriting database show that our method outperforms existing methods qualitatively and quantitatively, and its additional generated data can improve the performance of Handwriting Text Recognition (HTR) systems. The code is available at: https://github.com/koninik/DiffusionPen.

Keywords: Handwriting Generation · Latent Diffusion Models · Few-shot Style Representation

1 Introduction

Handwritten Text Generation (HTG) or Styled HTG is a challenging task recently gaining increased attention. The challenge lies in preserving the readability of specific textual content while capturing the unique characteristics of

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-73013-9_24.

A. Leonardis et al. (Eds.): ECCV 2024, LNCS 15143, pp. 417–434, 2025.
https://doi.org/10.1007/978-3-031-73013-9_24

a writer. The ability to automatically generate text that resembles a specific writing style could enhance personalization in digital design or potentially assist people facing writing challenges. Furthermore, more relevant to this work, it enables the augmentation of datasets to train efficient text recognition systems.

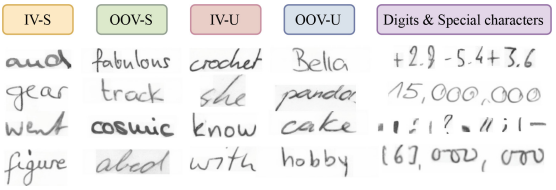


Fig. 1. Qualitative results generated using our method for four cases: In-Vocabulary words and Seen style (IV-S), In-Vocabulary words and Unseen style (IV-U), Out-of-Vocabulary words and Seen style (OOV-S), Out-of-Vocabulary words and Unseen style (OOV-U), as well as digits and punctuations.

Generative Adversarial Networks (GANs) have been the predominant method for offline HTG [1,6,15,22,34]. In terms of network architecture, Transformer-based solutions [3,25,38] are invariably employed, following the trends set in other fields. Among these standard methods, Denoising Diffusion Probabilistic Models (DDPM) [13] have recently emerged as a compelling alternative for HTG, offering a new paradigm distinct from traditional GANs and showcasing impressive results [24,43]. A common approach in GAN-based methods concerning the treatment of handwriting style is to incorporate a writer recognizer to classify the generated samples during training in order to force the generator to learn how to generate a specific handwriting style. However, adversarial training is known to suffer from limited diversity in the generated samples and presents instabilities in the training process [26, §15.1.4]. The same approach is not straightforward when using DDPM since the objective during training is to model the noise. Thus, the style space must be modeled more carefully.

DDPMs are a class of hierarchical Variational Autoencoders (VAEs) [13,17,36] that have recently garnered considerable interest within the representation learning and vision communities. Among an assortment of impressive results on a range of tasks, they have notably dominated the field of text-to-image generation by creating high-quality images given a text prompt [23,28,30,32]. Their success relies on the efficiency of the model itself and the use of pre-training techniques of large-scale image-text pairs [27]. While numerous diffusion-based systems demonstrate high-quality results in generating images given a text description [2,23,28,30,32], fewer works focus on generating readable scene-text images [4,41,43] or fonts [11,40] and, related to this work, generating handwriting [10,24,43].

In this work, we present a latent diffusion model that generates handwritten text images conditioned on a text prompt and a limited set of style samples in a few-shot scheme. As it can be seen in Fig. 1, our proposed method manages to generate realistic samples of seen and unseen styles as well as In-Vocabulary (IV) and Out-of-Vocabulary (OOV) words. Most importantly, we deal successfully with the problem of limited diversity when sampling from the posterior and

attempt to manipulate the output samples through various strategies. An effect relating conditional weighting and stereotypical sampling has been recently discussed in the context of diffusion-based modeling [26, §18.6.3]. To the best of our knowledge, this is one of the first works incorporating the few-shot style scheme in diffusion-based methods for HTG. We show that the resulting model leads to handwriting samples of simultaneously high diversity and high quality while conditioned on textual and style information.

Contributions. We propose *DiffusionPen* (*DiffPen*), a styled handwritten text generation method based on latent diffusion models. The method comprises a latent denoising autoencoder that performs the denoising diffusion process as the main network, and two auxiliary pre-trained encoders to create the style and textual conditions. The style encoder is based on the combination of classification and metric-learning training, which creates a continuous space for the style embeddings, providing more diversity to the generation process. The style condition is introduced in the main network in a few-shot setting to represent the unique characteristics of each writer from a limited set of $k = 5$ samples.

Our method is able to imitate the style of a writer given specific text content and five images from the specific writer. In particular, we show that we:

- *avoid posterior collapse*; given a text and style embedding, the proposed model is capable of producing highly diverse handwriting samples.
- *estimate a meaningful style space*; points in the style space invariably correspond to realistic, unseen styles.
- *outperform numerically the current state of the art by a significant margin.*
- *control style generation via style interpolation, style mixture & noise bias.*

We evaluate our proposed method by presenting both qualitative and quantitative results. Through qualitative results, we show that our method is able to generate IV and OOV words of both seen and unseen writer styles. We quantify generated data quality by computing commonly used metrics and comparing versus other SotA methods. Furthermore, we quantify style quality by examining whether a writer recognizer trained on real data can recognize the writer class of the generated data. To quantify diversity of the generated data, we conduct extensive experiments using an "auxiliary" HTR task; in particular, we use our model to imitate the real dataset and proceed to explore the variation present in the generated data by measuring the extent of improvement over HTR performance after using diffusion-generated data in the training process. Moreover, we present different sampling strategies that incorporate noise bias, style interpolation, and style mixture that showcase how style can be controlled and give extra variation to the generation. Finally, we present limitations with practical solutions, leading to future work perspectives and discuss ethical considerations.

2 Related Work

Handwritten Text Generation. The steady progress in the expressiveness and sophistication of generative modeling has enabled HTG, especially after the

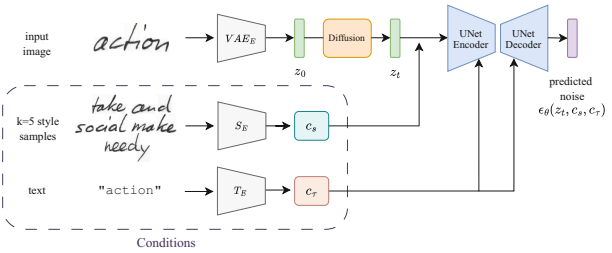
advent of adversarial modeling. Most works focusing on offline HTG rely on GAN-based approaches. Alonso *et al.* [1] present a GAN-based approach that takes as input a sequential text embedding encoded by an LSTM Recurrent Neural Network and further deploys an auxiliary network that uses CTC loss to recognize the generated text. Similarly, ScrabbleGAN [34] uses a text recognizer to help improve the quality of the generated text and character filters, showing variability in style and stroke width. Both approaches focus mostly on conditioning on the text content. On the contrary, Davis *et al.* [6] present a method that conditions on both text and style to generate realistic handwritten lines of arbitrary length by predicting the space required between text. Likewise, GANwriting [15] is a GAN-based system conditioned on text and few-shot stylistic samples and is trained in an adversarial manner with additional help from a text recognizer and a writer classification network. The method manages to generate realistic handwritten images of in-vocabulary and out-of-vocabulary words of seen and unseen writer styles. The work is further extended in [14] to also work for whole sentences. Although GANwriting generates understandable and stylistic samples there are several artifacts present in the generated data. An approach based on GANwriting named SmartPatch was introduced in [22] to tackle these artifacts.

Also based on a GAN framework and adversarial training, [3] and [25] have combined the encoder-decoder nature of Transformers with a few-shot style encoding to generate handwritten text. Further following the image synthesis trends, the works presented in [24] and [43] have introduced the application of Diffusion Models to synthesize understandable handwritten text. These systems have the ability to generate high quality text conditioning on a writer style and a text content, however they are limited in the way they represent and handle unseen styles. In this work, we address the limitations of the aforementioned approaches [24,43] and propose a Diffusion-based generative model that can produce unseen writing style samples by deploying pre-trained writer classifiers in a few-shot setting.

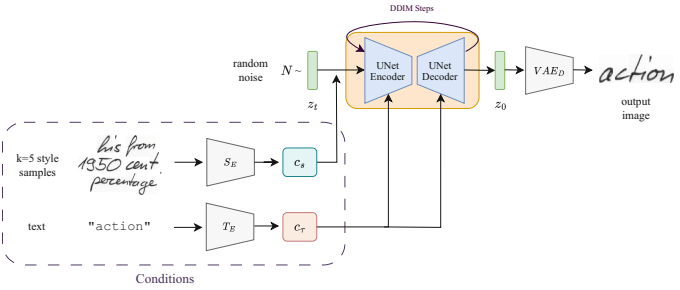
Few-Shot Conditional Diffusion Models. Few-shot Conditional Generation is the task of generating new samples of a specific class or object by conditioning on a few samples instead of a class embedding. This further enables the generation of unseen classes. Few-Shot Diffusion Models [8] condition the generation on a small set of image patches using a Vision Transformer (ViT). D2C [35] is a conditional few-shot Latent Diffusion Model that utilizes contrastive self-supervision to learn the latent space. The existing work indicates that there is plenty of room to explore conditioning diffusion models in few-shot schemes.

3 Proposed Method

The problem formulation of this work can be described as follows. Given $k = 5$ samples written by a writer $w \in W$ and a word t comprising i characters, our goal is to generate new images Y_w^t that depict the text content in t and the style of writer w. This task can be cast in terms of a conditional generative model,



(a) DiffusionPen Training Pipeline.



(b) DiffusionPen Sampling Pipeline.

Fig. 2. Overview of *DiffusionPen*. *DiffusionPen* comprises the conditional generator UNet Encoder-Decoder, having a Text Encoder T_E, a Style Encoder S_E, and a VAE_E encoder during training (2a) and VAE_D decoder during sampling (2b).

where we need to learn a distribution of handwriting samples $q(\cdot)$. Sampling over the distribution (conditioned on w, t) will produce the desired new images Y_w^t.

Prior work on HTG, using similar considerations with GAN-based approaches [3,15,25] or diffusion modeling [24,43] is hindered by two correlated issues. First, the style space is inadequately modeled in the sense that points in the sample space are not guaranteed to correspond to a meaningful style. This is particularly visible in the results of Sect. 4, where some of the compared methods achieve very high CER and WER scores when used to train an HTR system, indicating that they lack variation due to mode collapse. Second, sampling from the posterior given style and content gives samples that are practically too close to specific distribution modes.

We deploy a Conditional Latent Diffusion Model in combination with an existing text encoder and a feature extractor that operates in a few-shot scheme on the style samples. The feature extractor is trained using a hybrid metric-learning and classification approach to obtain a more intuitive feature space for the writer-style representations. In this manner, we constrain the learned style space to retain a sense of prescribed style distance.

Style Encoder. Given a batch of images, the goal of the style encoder S_E is to extract meaningful feature representations that encapsulate the writer characteristics of each image to ultimately be used in a few-shot learning setting and

condition the diffusion model training. To this end, we utilize a MobileNetV2 backbone [33] as the style encoder S_E due to its high performance and lightweight design, and we combine a classification and metric learning approach during training. A small ablation on the choice of the backbone is presented in the supplementary material.

Given a sample image s_w, used as an anchor, the model learns its stylistic characteristics from a random positive sample s_+ from the same writer and a random negative sample s_- from a different writer. The model learns the similarity between the samples using a triplet loss $\mathcal{L}_{triplet}$ formulated as: $\mathcal{L}_{triplet}(s_w, s_+, s_-) = \max(0, \delta_+ - \delta_- + \alpha)$, where $\delta_\pm = \|f_{s_w} - f_{s_\pm}\|_p$ and α is a margin. Furthermore, the feature representation of the anchor sample f_{s_w} is passed through a classification layer to predict its writer class. The classification part is optimized using Cross-Entropy as the classification loss $\mathcal{L}_{class}$. The model is trained using the combination of the two parts, formulating the loss as: $\mathcal{L}_{comb} = \mathcal{L}_{class}(f_{s_w}, w) + \mathcal{L}_{triplet}(s_w, s_+, s_-)$. A graphical representation of the style encoder hybrid training is presented in Fig. 3. This hybrid approach provides a feature space that keeps the different classes well-separated and robustness in intra-class variation. More details about the training of the style encoder are presented in Sect. 4.1.

Given the pre-trained style encoder, the style condition c_s that is fed into the diffusion model is created as follows. For every image in the training set, we consider k samples from the same writer class and pass them through S_E to extract the feature representation of each style image s_k. Then, we aggregate the extracted d-dimensional features of the style images by obtaining the mean of the k feature embeddings $s_{emb} \in \mathbb{R}^d$. Unlike most works that use 15 samples [3,15,22,25] to get the stylistic characteristics, we condition the writer style on $k = 5$ style features. Finally, the mean feature embedding is projected to the model dimension using a linear layer F_{proj}, giving the final style condition as $c_s = F_{proj}(s_{emb}) \in \mathbb{R}^{d_{model}}$.

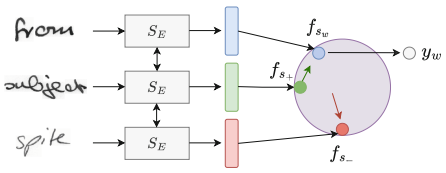


Fig. 3. A graphical representation of the hybrid style encoder S_E training. The style encoder creates the feature representations of the anchor sample f_{s_w}, positive sample f_{s_+}, and negative sample f_{s_-}. The metric learning training part pushes the positive features closer to the anchor and the negative features further away. The model uses the class prediction y_w of the anchor for the classification optimization.

Text Encoder. The text condition c_τ defines the textual content depicted in the images. The condition is created by using CANINE-C [5] as the text encoder. CANINE-C is an encoder that operates directly on character sequences without using an explicit vocabulary. This is particularly useful in the case of handwriting generation in order to generate OOV words. First, the raw character input

sequence τ is given to the CANINE tokenizer to get a structured format of the words, giving a unique token to each character and padding every word to a maximum length for batch processing. An encoded embedding τ_{emb} of the tokenized input is then created by the text encoder, and then, F_{proj} is applied to the text embedding to obtain the text condition $c_\tau = F_{proj}(\tau_{emb}) \in \mathbb{R}^{d_{model}}$. The text encoder and Conditional Latent Diffusion Model are trained using the objective described in the following paragraphs.

Conditional Latent Diffusion Model. Diffusion models can be understood as a special case of a Variational Autoencoder, where the latent space is defined as a Markov chain consisting of random variables $z_1, \cdots, z_T$. These variables have the same dimensionality as the initial sample x_0 and furthermore the encoder is (usually) fixed, with Gaussian noise being added layer after layer of the Markov chain. In diffusion modeling, we aim to learn the decoder or otherwise termed reverse or denoising phase, to be understood as letting the network learn how to gradually remove noise from z_T gradually back to the original sample space.

For the latent diffusion-based network, we utilize a UNet architecture [31], similar to WordStylist [24], as the network that learns the noise distribution to be removed. To reduce computational cost, we use a pretrained VAE encoder [30] to map the original image input of shape $W \times H$ into a 4-D latent representation $z \in \mathbb{R}^{4\times W/8\times H/8}$ as input to the network that performs the diffusion and the denoising process. In the forward diffusion process, a timestep $t \in [0, T]$ and Gaussian noise $\epsilon \in \mathbb{R}^{4\times W/8\times H/8}$ are sampled to corrupt the initial latent representation z_t. The network is trained using the denoising loss between the sampled Gaussian noise ϵ and the predicted noise ϵ_θ, as: $L = \|\epsilon - \epsilon_\theta(z_t, c_s, c_\tau)\|_2^2$. In the backward denoising process or sampling, given a style embedding and a text condition, the denoising autoencoder predicts and subtracts the present noise given the previous denoised sample. Finally, the predicted latent sample is given to the VAE decoder to create the final image. Figure 2 presents the overall architecture of our method.

4 Experiments

4.1 Datasets, Training Setup, and Considered SotA Approaches

Datasets. IAM Offline Handwriting Database [21] is one of the most commonly used datasets for handwriting recognition. It contains ∼115K isolated words and their transcriptions written by various writers. Similar to [15], we use 339 writers to train the style encoder and diffusion model, and we keep 160 for the experimental evaluation of the unseen style scenario and the HTR system. Additionally, we use the GNHK dataset [19], which includes unconstrained camera-captured images of English handwritten text, and show qualitative results in the supplementary material.

Training Setup. The training process occurs in two stages: the *Style Encoder* and the *Denoising Model* training. The *Style Encoder* is trained as described

in Sect. 3, using IAM database. The style extractor is trained for 20 epochs, with a batch size of 320, Adam [18] as the optimizer, and a learning rate of 0.001 that is reduced by a factor of 0.1 every 3 epochs and weight decay of 0.0001. We used a random selection for the negative samples as the inherently varied and nuanced differences across writers reduce the chance of selecting an easier negative; hence, the randomly chosen examples are sufficiently challenging, and no convergence issues were observed. All images are initially rescaled to a height of 64 pixels, preserving the aspect ratio. If the width of an image after rescaling is less than 256 pixels, padding is added to a fixed width of 256 pixels. Otherwise, the image is resized in height and width until obtaining a width less than 256 pixels and then padded in both height and width to a fixed size of 64×256. The same image pre-processing is also used in the main model training. The style encoder is trained independently from the diffusion model and is kept frozen during the diffusion training to create the stylistic feature condition. For the *Denoising Model* training, we use DDIM [37] noise scheduler for the noise injection and sampling. During training, the diffusion timesteps are set to 1K, while for sampling, the noise scheduler gives the flexibility to reduce the backward timesteps to 50. AdamW [20] is the optimizer with a weight decay of 0.2 and a learning rate of 0.0001. Every model is trained with a batch size of 320 for 1K epochs on a single A100 SXM GPU.

Considered SotA Approaches. For qualitative and quantitative comparison with the literature, we consider the GAN-based methods GANwriting [15] and SmartPatch [22], the Transformer-based VATr [25], and the Diffusion-based WordStylist [24]. GANwriting, SmartPatch, and VATr are similar to our method in terms of few-shot style condition. However, these methods use 15 samples to create the style embedding, while we use only 5. Furthermore, these methods use an auxiliary writer identification network as the feature extractor that is trained dynamically for the task with the generator. On the other hand, WordStylist is relevant to our method, as the main denoising diffusion process and network are similar, while the key difference is that WordStylist conditions on the style as a whole class embedding, which limits it to create only previously seen styles.

4.2 Quality Assessment

Figure 1 shows that our approach manages to generate samples of Seen (S) and Unseen (U) styles, In-Vocabulary (IV) and Out-of-Vocabulary (OOV) words, as well as digits and special characters. Comparative visual results with the SotA methods are also presented in Fig. 4. Furthermore, examples of generated words containing more than 10 characters are presented in Fig. 5b, showcasing the ability of our model to create longer words. Unseen styles are also presented in Fig. 5a. Finally, we present small paragraphs generated using our method in Fig. 6. More visual examples of both IAM and GNHK datasets, highlighting the notable variety of simulated styles and capabilities of our method, are included in the supplementary material.

To assess and quantify the quality of the generated words, we compute the Fréchet Inception Distance (FID) score [7], the Mean Structural Similarity Index

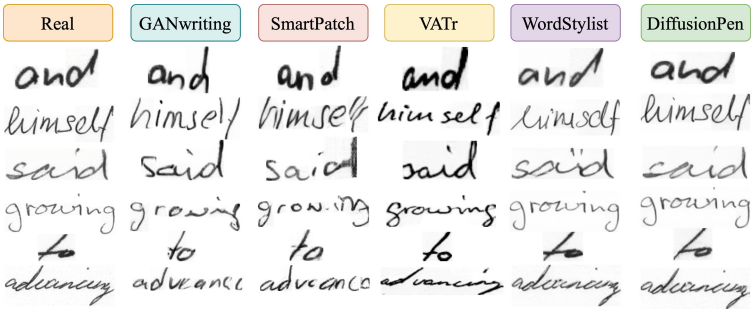
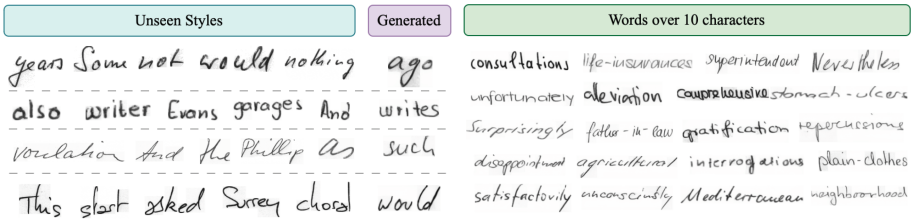


Fig. 4. Visual comparison of images generated by the considered approaches and our proposed method (DiffusionPen).



(a) Unseen styles generation. **(b)** Words containing more than 10 characters.

Fig. 5. (a) Exemplar generated samples from Unseen Styles. On the left, we can see the 5 style samples used for the style condition, and on the right, the generated word. **(b)** Generated words of different styles comprised of more than 10 characters.

(MSSIM) [39], the Root Mean Squared Error (RMSE), and the Learned Perceptual Image Patch Similarity (LPIPS) [42] scores. These metrics are commonly used to evaluate generative models. However, their use in HTG is not really intuitive, as they either rely on an ImageNet pre-trained network or compute pixel-wise similarities. Furthermore, we approach the evaluation through a writer classification strategy, following a more document-oriented strategy. To this end, we deploy a ResNet18 architecture [12], pre-trained on ImageNet, and finetune it on the IAM database for the task of writer-style classification. We use a *different backbone* from our style extractor to avoid any bias induced by our model training. The goal is to train the recognizer on a subset of the real training set and then evaluate its performance on the entire set of synthetic samples generated by different methods that simulate IAM (same words, same styles). This approach aims to determine whether the recognizer can correctly classify the generated samples, regardless of whether it has seen the corresponding real samples during training or not.

Table 1 presents the aforementioned metrics obtained using the different considered methods. Our proposed method and its variations non-trivially outperform the other HTG approaches for all metrics. Similarly, for the task of writer identification, the model successfully classifies a high percentage of samples gen-

Table 1. Comparison of FID, MSSIM, RMSE, LPIPS, and classification accuracy with previous methods. For FID, RMSE, and LPIPS, the lower, the better.

Method	FID↓	MSSIM↑	RMSE↓	LPIPS↓	Acc(%)↑
Real IAM	–	–	–	–	92.34
GANwriting	43.97	0.777	0.3118	0.2912	3.25
SmartPatch	50.21	0.757	0.3207	0.3003	2.14
WordStylist	34.20	0.955	0.1576	0.1080	68.25
DiffPen-class (Ours)	22.23	0.967	0.1587	0.1114	68.04
DiffPen-triplet (Ours)	22.06	0.953	0.1612	0.1127	67.50
DiffPen (Ours)	22.54	0.963	0.1505	0.1072	70.31

erated by our method with an accuracy of 70.31%. This result overpasses the 68.25% of the recent WordStylist approach that learns the style class in an explicit manner. In general, WordStylist and DiffusionPen have very similar performances in most metrics. This behavior is expected, as the two systems share backbone architecture. Our results suggest that, while we are able to reproduce the style slightly better than Wordstylist (which explicitly uses the style class as an embedding), we have managed to introduce more variation to the generated samples, which is the most crucial component in improving HTR performance when training on generated samples. Within this experimental setup, we also conduct an ablation study on the usefulness of our proposed style extractor by exploring the role of the loss terms. A breakdown of the loss terms of each ablation variation and additional discussion on the benefits of the style extractor are presented in the supplementary material. Due to format issues, the metrics for VATr are not included in the table. From the presented results, we can draw the following conclusions. First, the writer classification accuracy is increased using the hybrid style embedding, namely through the joint classification and triplet scheme. This is not the case for the other metrics, but the differences are non-significant, and their relevance to the HTG task is far inferior compared to the writer classification paradigm. Moreover, previous methods, such as GANwriting [15] and SmartPatch [22], despite having paved the way towards generating realistic images of handwritten words, seem unable to simulate the varieties of writing styles existing in IAM. Finally, our proposed hybrid style embedding outperforms all the reported methods for all the considered metrics.

Unseen Styles. Unlike previous works that use Diffusion Models for HTG [24, 43], DiffusionPen can imitate a writing style not seen during training from a few samples. We present qualitative results of generating unseen writing styles in Fig. 5a. Our method can replicate the style of the unseen style samples used as conditions and produce understandable text. More generation examples of Unseen styles, with both IV and OOV words, are included in the supplementary material. We should highlight the low number of required exemplars – only 5 – used for the generation, enabling potential applications where writers' data are

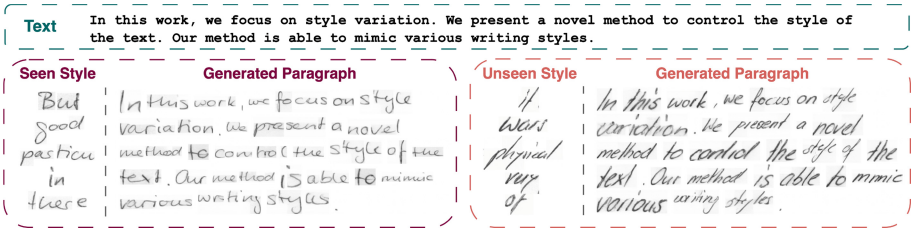


Fig. 6. A small paragraph generated in a Seen and Unseen Style.

limited. Additionally, the mean aggregation used over the exemplar embeddings suggests that one can use a variable number of exemplar images without issue. We showcase how the number of samples in the few-shot setting affects the generation in the supplementary material.

4.3 Handwriting Text Recognition

Similar to previous works [24,43], we evaluate the quality of the generated handwritten text on the task of Handwriting Text Recognition (HTR) on the word level. We use a CNN-LSTM HTR system [29] trained with Connectionist Temporal Classification (CTC) loss [9], as used in the evaluation process of [24].

Imitating IAM. We regenerate the training set and use the generated data to train the HTR system. Then, we evaluate the HTR performance on the real test set, aiming to reach results as close as possible to the real training data. The motivation behind this experiment is straightforward yet powerful. Specifically, an HTG method could reproduce the performance of the real IAM data, or even surpass it, if these three abilities are satisfied: 1) the textual information is generated correctly, 2) styles differ substantially between them, and 3) given a text and a style a non-trivial variation would be generated. Even if point (1) is very crucial, the main shortcomings of recent HTG methods concern points (2) and (3) in the sense that these methods do not generate enough variations in order to be efficiently utilized for such a learning task. An example to understand the importance of this rationale is that if the inner-class variance of the generation process is trivial, generating 10 times

Table 2. Comparison of HTR Results using only the synthetic IAM samples for training. The closer to the Real IAM result (first row), the better.

Dataset	CER(%)↓	WER(%)↓
Real IAM	5.16 ± 0.01	14.49 ± 0.07
GANwriting	39.94 ± 0.35	73.38 ± 0.61
SmartPatch	39.81 ± 0.83	72.75 ± 0.19
VATr	21.74 ± 0.32	50.55 ± 0.47
WordStylist	8.26 ± 0.05	23.36 ± 0.16
DiffPen-class (Ours)	7.12 ± 0.03	18.55 ± 0.10
Diff-triplet (Ours)	7.13 ± 0.11	18.48 ± 0.14
DiffPen (Ours)	6.94 ± 0.06	18.11 ± 0.25

the common word "and", typically met numerous times in documents, can not provide any useful extra information to the training procedure.

Concerning the imitation experiment, we follow the same steps in [3,15,22,24,25]. The HTR results are presented in Table 2. Our method reaches the closest results to the real data with a CER of 6.94% and WER of 18.11%, outperforming all the other methods. Similar to the quality assessment experiments, we include experiments to assess the effectiveness of our introduced style extractor and its loss terms. The variations of DiffusionPen, where the style encoder is trained with the classification or the triplet protocol, achieve the next best performance. WordStylist achieves the next closest performance with a CER of 8.26% and a WER of 23.36%.

Improving HTR Performance. Given the results of Table 2, we use the data generated from the best performing method, which is our proposed DiffusionPen, as an augmentation to the real training set aiming to improve the performance of the baseline HTR system that achieves a CER of 5.16% and a WER of 14.49%. We also compare our performance with other works [16,34,43] that use synthetic data, as shown in Table 3. Using the additional data from DiffusionPen can enhance the performance of the HTR system, showing promising potential for future use in larger generated datasets to assist the training process.

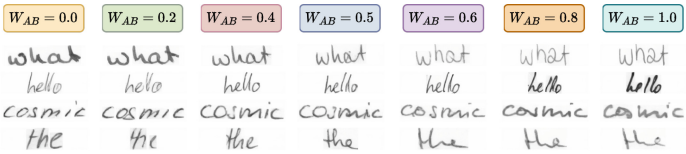
Table 3. HTR performance with additional synthetic data to the real training set. The baseline values The baseline values are the ones from the original paper [29].

Dataset	# Synthetic Data	CER(%)↓	WER(%)↓
ScrabbleGAN [34]	100K	13.42	23.61
Kang et al. [16]	-	6.75	17.76
CTIG-DM [43]	1M	5.19	13.37
Baseline [29]	-	5.14	14.33
DiffusionPen (Ours)	55K	4.71	13.61

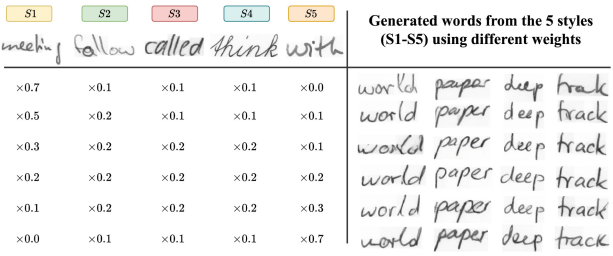
4.4 Style Variation

To reflect the natural variations of human handwriting in automatic handwriting generation, it is crucial not only to generate realistic text but also to have diversity. Under our framework, style variation can be simulated seamlessly. First, the few-shot paradigm is, by its nature, a variation-promoting mechanism since, for the same style, different embeddings are calculated from the randomly selected exemplar images. Nonetheless, the style space is less sensitive to "exploration", compared to previous works, enabling us to search for more "aggressive" style augmentations. Specifically, we explore the aspect of style variation through the following scenarios: interpolation and noisy style embedding.

Interpolation. We tweak the style embedding between two writer styles to obtain samples between the two styles. We interpolate a generated sample of a style S_A to a style S_B using a weighted average $S_{AB} = (1 - W_{AB})S_A + W_{AB}S_B$ for different values of W_{AB}. Figure 7a presents several examples of interpolating between two random styles. We can see from the results that there is a smooth transition between the styles as the value of W_{AB} changes. An interesting result can be observed in the last row of the figure, where for $W_{AB} = 0.5$ of the word `the`, the curvature of the character `h` does not resemble either of the style classes. This hints that we may "stumbled" upon an entirely different style by cursing through the style space.



(a) Interpolation between two writer styles with various values of the weight W_{AB}.



(b) Multi-style mixture generation.

Fig. 7. Generated samples by **(a)** interpolation and **(b)** multiple styles.

Multi-style Mixture. Following the interpolation strategy, instead of mixing two writing styles, we generate samples by conditioning on 5 different styles. Figure 7b shows that combining 5 different styles (S1-S5) with different weights can create new styles and variations of the same word, showcasing our model's capability of exploiting the style space. More style mixture examples are presented in the supplementary material.

Noisy Style Embedding. We explore two different noise variations that could inject diversity into the generated data. One variation is the *noisy style embedding*, where we add random noise to the style embedding to avoid explicitly getting the learned style. The other variation is the *noise bias*. We inject noise bias into the diffusion model by replacing the random noise given to the diffusion model as the initial latent variable with a noisy image from a wished style. We regenerate the IAM database using either the noise bias to initiate the denoising of the model or the noisy style embedding for different levels of

noise or the combination. Then, we train the HTR system using the different generated databases instead of the real IAM training set to determine how the noise injections can assist the data variation in the generation.

The results are presented in Table 4. One can observe that a small magnitude of noise (i.e., 0.25) could assist the training procedure and improve the HTR performance, indicating that the noise variations are beneficial. On the other hand, increasing noise above a threshold may complicate the system - potentially diverging from the manifold of useful styles. Finally, the noise bias does not seem to assist performance, even when combined with the noisy embedding of 0.25 magnitude.

Table 4. Exploration of random noise variations in the style embedding or the prior. The ✓ indicates whether there is a bias in the prior, and the values 0.25–2.00 indicate the weight of the noise added to the style embedding.

Noise Bias	Style Noise	CER(%)↓	WER(%)↓
–	2.00	7.56 ± 0.06	19.56 ± 0.11
–	0.50	6.99 ± 0.06	18.30 ± 0.21
–	0.25	6.79 ± 0.08	17.85 ± 0.09
✓	–	6.93 ± 0.20	18.18 ± 0.49
✓	0.25	7.02 ± 0.20	18.26 ± 0.26

5 Limitations and Ethical Considerations

Limitations. Fail cases may occur in rare combinations of characters ("xyzyxz") or complex ligatures in some cursive styles ("affluent"), as shown in Fig. 8a. Considering the word "affluent", our model successfully generates the top style that has less complex connections, while it struggles with the cursive "ffl" ligature in the more complex style on the bottom. This might not be observed in comparing GAN-based methods, which tend to simplify the style to force the generation of understandable text. However, a trade-off between text and style variation should be found to get a robust generation. Furthermore, although our method is able to generate words over 10 characters (see Fig. 5b), there is still a length limit due to the maximum word length present in the training and the noise initialization of the denoising process. Such a case is presented in Fig. 8b, where the model fails to generate the word "interoperabilitationism" (top). This issue could be solved by practical tricks such as patching generated samples of smaller parts of the word, as presented in the bottom of Fig. 8b, where we generate the parts "interoper", "abilitation" and "ism".

Ethical Considerations. The possibility of mimicking a specific handwriting style from a limited set of image samples poses a significant risk for handwriting forgery. Although technologically impressive, this capability of HTG models could potentially be exploited for fraud and identity theft by creating unwanted text or signatures resembling a person's writing style. There is an entire field in forensics that attempts to detect such frauds with predictive models and techniques to distinguish authentic from machine-generated imitations.

(a) Rare character combinations (left) and complex ligatures (right).

(b) Unsuccessful generation of the word `interoperabilitationism` and solution.

Fig. 8. Examples of fail cases. See text for details.

6 Conclusion

We presented *DiffusionPen*, a few-shot style latent diffusion model for handwriting generation that incorporates a hybrid metric-learning and classification-based style extractor. Our approach captures stylistic features of seen and unseen writers while preserving readable text content. We present qualitative and quantitative results and compare them with other SotA methods based on GANs, Transformers, and DDPM. Given the HTR task results, our method is the closest to the performance of the real IAM, and using data generated from *DiffusionPen* enhances HTR performance, allowing us to envisage utilizing HTG large-scale dataset generation. Finally, further exploration of style and noise variation in different stylistic aspects shows potential directions for future work.

Acknowledgment. The computations and data handling were enabled by the Berzelius resource provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre at Linköping University. The publication/registration fees were partially covered by the University of West Attica.

References

1. Alonso, E., Moysset, B., Messina, R.: Adversarial generation of handwritten text images conditioned on sequences. In: 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 481–486. IEEE (2019)
2. Balaji, Y., et al.: eDiff-I: text-to-image diffusion models with an ensemble of expert denoisers. arXiv preprint arXiv:2211.01324 (2022)
3. Bhunia, A.K., Khan, S., Cholakkal, H., Anwer, R.M., Khan, F.S., Shah, M.: Handwriting transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1086–1094 (2021)
4. Chen, J., Huang, Y., Lv, T., Cui, L., Chen, Q., Wei, F.: Textdiffuser: diffusion models as text painters. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
5. Clark, J., Garrette, D., Turc, I., Wieting, J.: Canine: pre-training an efficient tokenization-free encoder for language representation. Trans. Assoc. Comput. Linguist. **10**, 73–91 (2021). https://api.semanticscholar.org/CorpusID:232185112
6. Davis, B.L., Tensmeyer, C., Price, B.L., Wigington, C., Morse, B., Jain, R.: Text and style conditioned GAN for the generation of offline-handwriting lines. In: Proceedings of the 31^{st} British Machine Vision Conference (BMVC) (2020)
7. Dowson, D., Landau, B.: The Fréchet distance between multivariate normal distributions. J. Multivar. Anal. **12**(3), 450–455 (1982)

8. Giannone, G., Nielsen, D., Winther, O.: Few-Shot Diffusion Models. ArXiv **abs/2205.15463** (2022). https://api.semanticscholar.org/CorpusID:249210127
9. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. **31**(5), 855–868 (2008)
10. Gui, D., Chen, K., Ding, H., Huo, Q.: Zero-shot generation of training data with denoising diffusion probabilistic model for handwritten Chinese character recognition. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) ICDAR 2023. LNCS, vol. 14188, pp. 348–365. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41679-8_20
11. He, H., et al.: Diff-Font: diffusion model for robust one-shot font generation. Int. J. Comput. Vis. 1–15 (2024)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
13. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems, vol. 33, pp. 6840–6851 (2020)
14. Kang, L., Riba, P., Rusinol, M., Fornés, A., Villegas, M.: Content and style aware generation of text-line images for handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. (2021)
15. Kang, L., Riba, P., Wang, Y., Rusiñol, M., Fornés, A., Villegas, M.: GANwriting: content-conditioned generation of styled handwritten word images. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12368, pp. 273–289. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58592-1_17
16. Kang, L., Rusinol, M., Fornés, A., Riba, P., Villegas, M.: Unsupervised writer adaptation for synthetic-to-real handwritten word recognition. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3502–3511 (2020)
17. Kingma, D., Salimans, T., Poole, B., Ho, J.: Variational diffusion models. In: Advances in Neural Information Processing Systems, vol. 34, pp. 21696–21707 (2021)
18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: International Conference on Learning Representations (2015)
19. Lee, A.W.C., Chung, J., Lee, M.: GNHK: a dataset for English handwriting in the wild. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021 Part IV. LNCS, vol. 12824, pp. 399–412. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86337-1_27
20. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2017). https://api.semanticscholar.org/CorpusID:53592270
21. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recogn. **5**, 39–46 (2002)
22. Mattick, A., Mayr, M., Seuret, M., Maier, A., Christlein, V.: SmartPatch: improving handwritten word imitation with patch discriminators. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12821, pp. 268–283. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86549-8_18
23. Nichol, A.Q., et al.: GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In: International Conference on Machine Learning, pp. 16784–16804. PMLR (2022)

24. Nikolaidou, K., et al.: WordStylist: styled verbatim handwritten text generation with latent diffusion models. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) ICDAR 2023. LNCS, vol. 14188, pp. 384–401. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41679-8_22
25. Pippi, V., Cascianelli, S., Cucchiara, R.: Handwritten text generation from visual archetypes. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 22458–22467 (2023). https://api.semanticscholar.org/CorpusID:257766680
26. Prince, S.J.: Understanding Deep Learning. MIT Press, Cambridge (2023)
27. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning (2021). https://api.semanticscholar.org/CorpusID:231591445
28. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 **1**(2), 3 (2022)
29. Retsinas, G., Sfikas, G., Gatos, B., Nikou, C.: Best practices for a handwritten text recognition system. In: Uchida, S., Barney, E., Eglin, V. (eds.) DAS 2022. LNCS, vol. 13237, pp. 247–259. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-06555-2_17
30. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10684–10695 (2022)
31. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015 PArt III. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
32. Saharia, C., et al.: Photorealistic text-to-image diffusion models with deep language understanding. In: Advances in Neural Information Processing Systems, vol. 35, pp. 36479–36494 (2022)
33. Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018). https://api.semanticscholar.org/CorpusID:4555207
34. Fogel, S., Averbuch-Elor, H, Cohen, S., Mazor, S., Litman, R.: ScrabbleGAN: semi-supervised varying length handwritten text generation. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4323–4332 (2020)
35. Sinha, A., Song, J., Meng, C., Ermon, S.: D2C: diffusion-decoding models for few-shot conditional generation. In: Advances in Neural Information Processing Systems, vol. 34, pp. 12533–12548 (2021)
36. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In: International Conference on Machine Learning, pp. 2256–2265. PMLR (2015)
37. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: International Conference on Learning Representations (2021)
38. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
39. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)

40. Yang, Z., Peng, D., Kong, Y., Zhang, Y., Yao, C., Jin, L.: FontDiffuser: one-shot font generation via denoising diffusion with multi-scale content aggregation and style contrastive learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 6603–6611 (2024)
41. Zhang, L., Chen, X., Wang, Y., Lu, Y., Qiao, Y.: Brush your text: synthesize any scene text on images via diffusion model. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 7215–7223 (2024)
42. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 586–595 (2018)
43. Zhu, Y., Li, Z., Wang, T., He, M., Yao, C.: Conditional text image generation with diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14235–14245 (2023)