DiffusionPen: Towards Controlling the Style of Handwritten Text Generation - Supplementary Material

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

[1] Luleå University of Technology, Sweden
firstname.lastname@ltu.se
[2] National Technical University of Athens, Greece
gretsinas@central.ntua.gr
[3] University of West Attica, Greece
gsfikas@uniwa.gr

We present the supplementary material for our proposed few-shot styled Handwritten Text Generation system, named DiffusionPen (DiffPen). In particular, we include additional information considering the Style Encoder backbone and the versions of our method in Sec. 1. Furthermore, we present additional qualitative results of DiffusionPen in Sec. 2 and compute the FID score for the generated IAM test sets that combine both In-Vocabulary (IV) and Out-of-Vocabulary (OOV) words but only include Unseen styles. In the same section, we showcase the ability of the model to generate smaller paragraphs or longer words. In Sec. 3, we present further visual examples to explore the effect of the style embedding, the noise induction in the style embedding, and the initialization noise bias. Furthermore, we show examples of style mixtures where we generate new styles by combining up to 5 different styles. Moreover, we show that our model can generate high-quality samples even with 1-shot style sampling. Finally, Sec. 4 shows extended Handwriting Text Recognition (HTR) results using generated data as an augmentation to the real IAM data [4] to improve the HTR performance.

1 Style Encoder

Backbone. In our work, we utilize a Style Encoder S_E that is trained with a hybrid triplet and classification loss to model the style of the word images. We conducted preliminary experiments on style classification using ResNet18, ResNet50, and MobileNetV2, all pre-trained on ImageNet, to choose the backbone of the Style Encoder. The resulting accuracy, along with the number of parameters of every model, is presented in Tab. 1. One can see that the performance of all three models is similar, with ResNet50 achieving the best accuracy, being close to the following best, which is MobileNetV2. However, both ResNet architectures are much heavier in terms of parameters than the MobileNet architecture. Thus, MobileNet's combination of high performance and lightweight design made us proceed with that choice.

Table 1: Ablation on the Style Encoder backbone network.

Method	**Accuracy(%)↑**	#parameters
MobileNetV2	89.21	2.7M
ResNet18	88.23	11.4M
ResNet50	90.37	24.2M

Table 2: Loss terms included in the variations of our proposed method for the ablation of the style encoder S_E. $\mathcal{L}_{class}$ represents the classification loss term and $\mathcal{L}_{triplet}$ the metric-learning term.

Method	$\mathcal{L}_{class}$	$\mathcal{L}_{triplet}$
DiffPen-class	✓	
DiffPen-triplet		✓
DiffPen	✓	✓

Hybrid Loss. Within the experimental setup of our work, we conduct an ablation on the usefulness of our proposed style extractor by exploring the role of the loss terms. A breakdown of the loss terms of every ablation variation is presented in Tab. 2. DiffPen corresponds to the model that uses the hybrid Style Encoder with both triplet and classification terms in the loss. Besides the results presented in the full model, we also present qualitative results using the model that uses each loss term separately. Hence, DiffPen-class corresponds to the model where the Style Encoder is trained only with Cross-Entropy loss $\mathcal{L}_{class}$ and DiffPen-triplet $\mathcal{L}_{triplet}$ to the one trained only with the triplet loss.

In general, a simple classification loss (as used in previous approaches) will indeed generate samples that look like specific styles. However, such an approach forms a space of style descriptors that can easily degenerate to a set of centers around which points are classified to a style in a "nearest neighbour" sense - this explains the limited style variability of previous methods. A classification loss is adequate for a *discriminative* model, but here it is inadequate because it is oblivious of the topology of the inferred space. The proposed loss faces this problem exactly by explicitly bringing into play *the metric characteristics* of the inferred style space. In practical terms, while the result in Tab1 doesn't seem significant (from WordStylist to DiffPen), there is still an improvement of 2% in reproducing the styles. The improvement is also very clear in Table 2 of the main paper, where if we run a significance test between DiffPen-class and DiffPen, we obtain a t-value of 4.77 and a p-value of 0.0088, making the improvement significant. All our model variations are also significantly better than the baseline WordStylist. This proves that we significantly add more variation to our produced samples, which is the main goal of our paper.

Table 3: Comparison of FID with previous GAN-based methods on the generated test set of IAM database. The test set consists of only Unseen styles and IV and OOV words. For FID, the lower, the better.

Method	**FID↓**
SmartPatch	48.01
GANwriting	44.67
DiffusionPen (Ours)	29.77

2 Qualitative and Quantitative Results

Comparison wit SotA. We present additional qualitative generated examples using our method and the comparing methods GANwriting [2], SmartPatch [5], Visual Archetype Transformers (VATr) [7], and WordStylist [6]. Fig. 1 shows several In-Vocabulary words of seen writer styles present in the original train set, created using the different generative methods. To quantify the observed results, we compute the FID score between the real and the generated test set, which contains solely unseen styles and both IV and OOV words, and compare with the corresponding test sets generated using GANwriting and SmartPatch. The resulting FID scores are presented in Tab. 3, where we can see that our system achieves the best result on the generated test set. It should be noted that we cannot compute the test set FID for WordStylist, as the system can only generate seen styles. The FID scores obtained from VATr are notably higher than the rest of the methods, and the cause of that requires further investigation. Thus, similar to our main paper, we decide not to include the FID for the VATr method.
Unseen Styles. Furthermore, we present additional qualitative results of unseen styles that were not present during the training of our proposed system for both In-Vocabulary (IV) and Out-of-Vocabulary (OOV) words. Fig. 2 shows the generated IV words on the right column (Generated) and the unseen style samples on the right column that constitute the 5-shot style embedding (Unseen Styles - IV). Fig. 3 shows similar results for the case of Unseen styles and OOV words (Unseen Styles - OOV). In both cases, one can observe that the generated handwriting on the right column maintains a consistent style in terms of letter formation, spacing, slant, and thickness with the style samples, suggesting that our method has effectively learned the handwriting characteristics from the limited set of unseen examples provided as a condition.
Paragraphs. We present two small paragraphs, comprised of two sentences, generated using our method in Fig. 4. This way, we show the practical applicability of not constraining the generation in words and the ability to generate larger parts of text by using 5 style word samples and specific content as conditions.
Limitations. As showcased in the main paper, our method has the constraint of generating the words in a specific image size due to noise initialization during sampling. Furthermore, the dataset is limited to a maximum number of characters in the training set, which limits the model when asking the model to generate longer words. However, this can be solved by patching smaller parts of the long

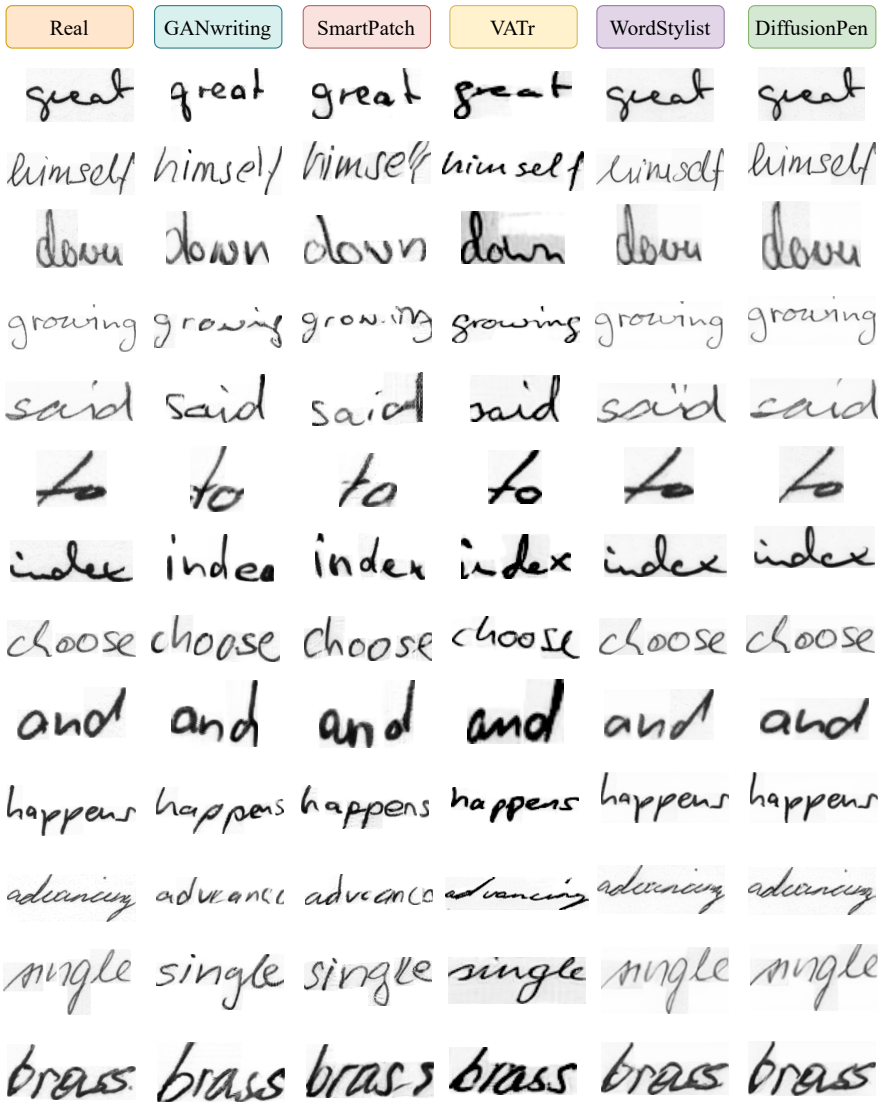


Fig. 1: Qualitative results of In-Vocabulary (IV) words and Seen (S) styles. We compare our method with GANwriting, SmartPatch, VATr, and WordStylist.

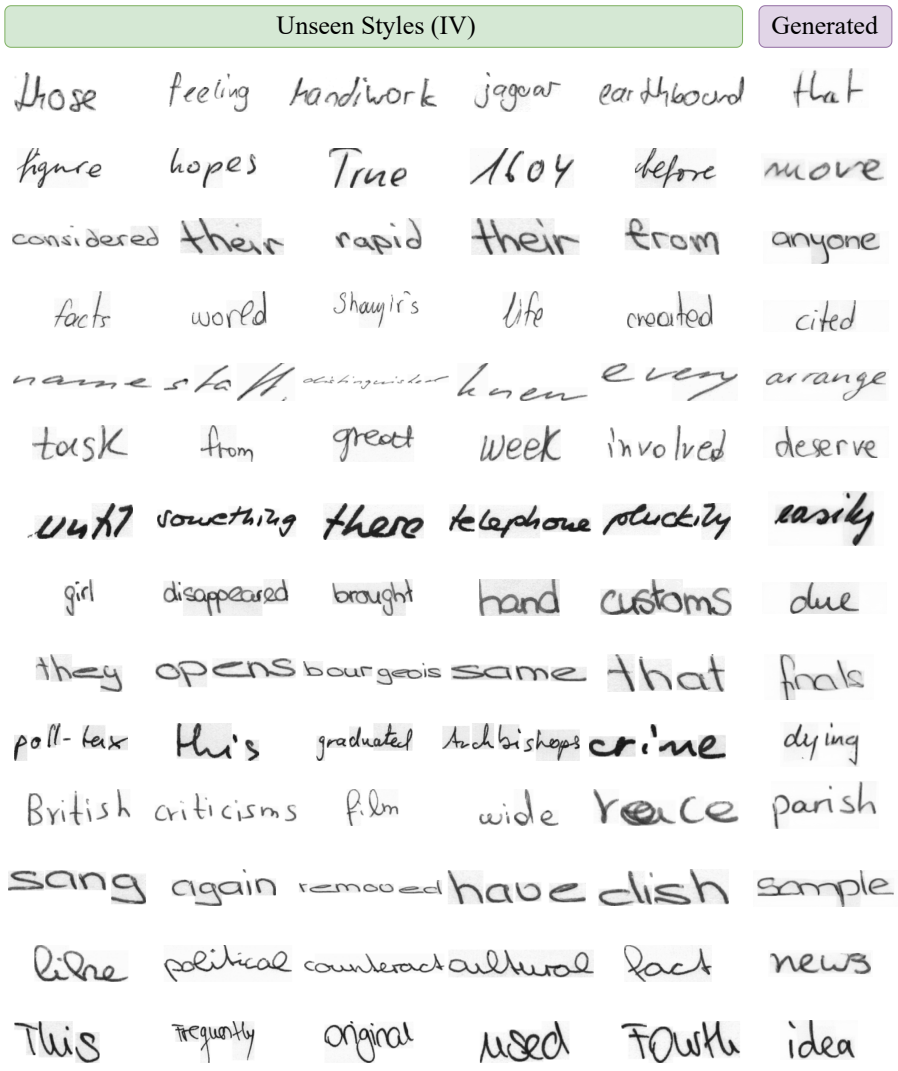


Fig. 2: Qualitative results of In-Vocabulary (IV) words and Uneen (U) styles. The left column (Unseen Styles IV) shows the style samples used for the 5-shot condition, and the right column (Generated) shows the generated IV word.

Fig. 3: Qualitative results of Out-of-Vocabulary (OOV) words and Uneen (U) styles. The left column (Unseen Styles OOV) shows the style samples used for the 5-shot condition, and the right column (Generated) the generated OOV word.

Fig. 4: Small paragraphs generated in a Seen (left) and Unseen Style (right).

word. We present a few more examples in Fig. 5. In these examples, we manage to generate the word “antidisestablishmentarianism" through the generation of the words “antidis", “establish", “mentarianism", and “anism". Similarly, we generate the word “collaborationalitatively" through the combination of “collabora", “tionalita", and “tively" and the word “fergalicioussodelicious" through the generation of the subwords “ferga", “licious", “so", and “delicious". In the presented examples, we have used a random split of the long words, as the main point is to have subparts shorter than the max word length present in the dataset. A systematic strategy could be devised by breaking a long word into (randomly-lengthed) segments, with each segment length smaller than the maximum word length.

GNHK dataset. We include several qualitative results of the GNHK dataset [3] on the word level in Fig. 6. The figure shows a reference style and the corresponding generated samples of the randomly selected conditioned text. We can see that our method successfully generates samples imitating the GNHK style. However, the dataset is much more complex than the IAM database, with more complex backgrounds and less benchmarking on the word level as it is provided as page images. Hence, more experimentation and adaptation are needed to meet the dataset’s needs.

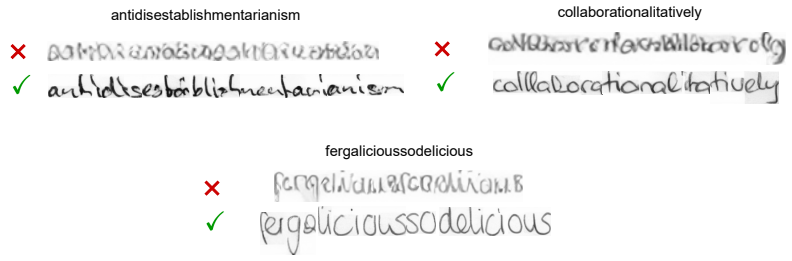


Fig. 5: Generation of very long words.

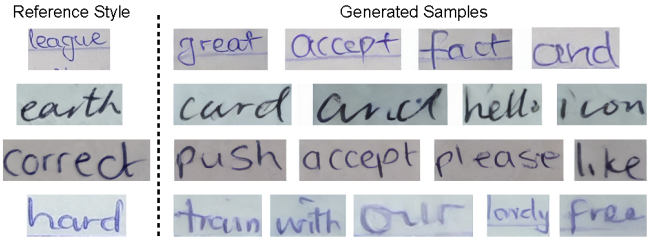


Fig. 6: GNHK generated examples using DiffusionPen.

3 Style Variation

We explore the effect of the style embedding on the generation process. We visualize a few examples of the same word-writer pair generated multiple times using different style embedding conditions. Furthermore, we explore the noise induction to the style embedding as well as the effect of adding bias to the prior noise that initializes the sampling process of the diffusion model.

Style Embedding. Fig. 7 shows the exploration of the style embedding. For a fixed text content, we generate the same word written by the same writer style multiple times while changing the style samples that constitute the style embedding condition. The results show that, for different style examples as conditions, the generated word has different variations.

Noisy Style Embedding & Noise Bias. Following our experiments presented in the main paper, we continue the exploration of the style variation through noise induction to the style embedding or through biasing the initialization noise. In Fig. 8, we present qualitative results and compare the generation of the same samples with our method (DiffusionPen) without any change, with the noise induction to the style condition embedding with a magnitude of 0.25 (Style Noise 0.25), and with the prior noise bias, where instead of initializing the generation with random noise, we randomly select an image from the same style and adding noise to it (Noise Bias). While the results are very close between the different cases, having a closer look, one can observe differences. The noise induction (Style Noise 0.25) seems to generate some marks in character “t" of the word “towards" or the digit “9" in “1951" that resemble ink stains. Furthermore, while

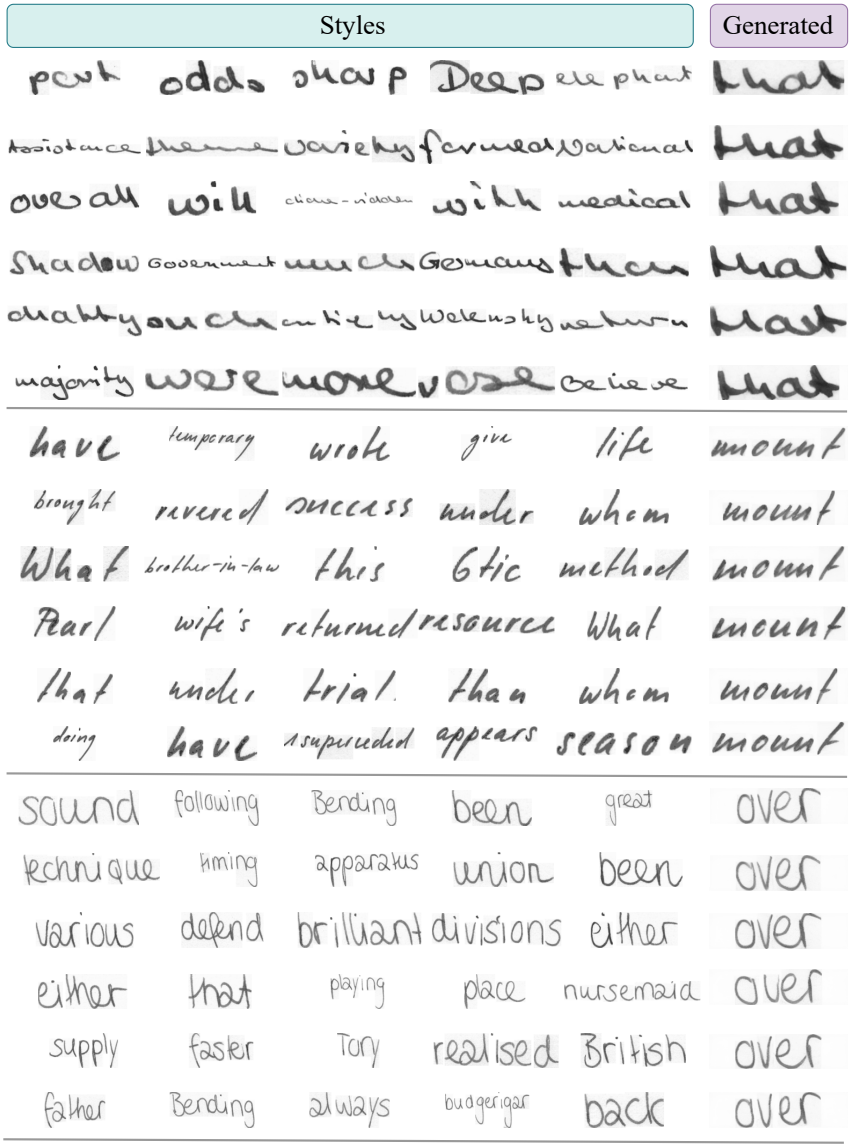


Fig. 7: Multiple generations of the same word (right column), conditioning on different seen style samples (left column). For every different style combination, we get a different variation of the word.

all cases have the accent of the first “i" of the word “pianist" slightly on the left of the character, the Noise 0.25 case is placed right on top of the letter. Considering the Noise Bias case (last column), the generated results seem to be the closest to the real data, such as the words “were", “1951", and “chines". There are small details that differentiate them, such as the extended “r" in “Minister", or the “b" in “blow". These details can induce small variations in the data while striving to control the generation's style. We further explore the HTR performance of these cases and comment on the results in Sec. 4.

Table 4: HTR performance with additional synthetic data to the real training set. The first row shows the performance of the real IAM database without any augmentation. The second row shows the performance of the real IAM dataset with the additional IAM synthetic samples generated from DiffusionPen. The results show the mean and standard deviation over three runs of each experiment.

Dataset	**CER (%) ↓**	**WER (%) ↓**
Real IAM	5.16 ± 0.01	14.49 ± 0.07
Real IAM + GANwriting IAM	5.22 ± 0.03	14.40 ± 0.13
Real IAM + SmartPatch IAM	5.48 ± 0.13	14.97 ± 0.35
Real IAM + VATr IAM	5.20 ± 0.16	14.37 ± 0.40
Real IAM + WordStylist IAM	4.75 ± 0.04	13.29 ± 0.11
Real IAM + DiffPen IAM	4.78 ± 0.07	13.72 ± 0.13

Table 5: HTR performance of the real IAM data and the data generated from DiffusionPen using the two exploration variations. Style Noise (0.25) represents the synthetic data created by adding noise to the style embedding of 0.25 magnitude, while Noise Bias represents the data where instead of random noise, a noisy image belonging to the same writer as the style condition is used to initialize the sampling process.

Dataset	**CER (%) ↓**	**WER (%) ↓**
Real IAM	5.16 ± 0.01	14.49 ± 0.07
Real IAM + Style Noise (0.25)	4.88 ± 0.05	13.97 ± 0.20
Real IAM + Noise Bias	4.86 ± 0.02	13.72 ± 0.16

Style Mixture. We extend the qualitative examples of Style Mixture presented in Figure 7 of the main paper and show the results in Fig. 9. One can see that our method is able to generate new styles by mixing more than two and adjusting the weights. This ability comes from modeling the style space and using the mean embedding of the 5 feature samples. This is not the case for the comparing methods that perform the few-shot style condition, as they concatenate the style features, thus obtaining a different embedding every time.

Fig. 8: Style variations of the noisy style embedding and the noise bias exploration.

Few-Shot Style Effect. We explore how the number of style samples affects the generation during sampling. We experiment with 1-5 samples to condition the word generation and present the results in Fig. 10. One can see that although the model is trained with a condition of $k = 5$ samples, it can still generate quality samples with fewer style images, even one.

4 Handwriting Text Recognition

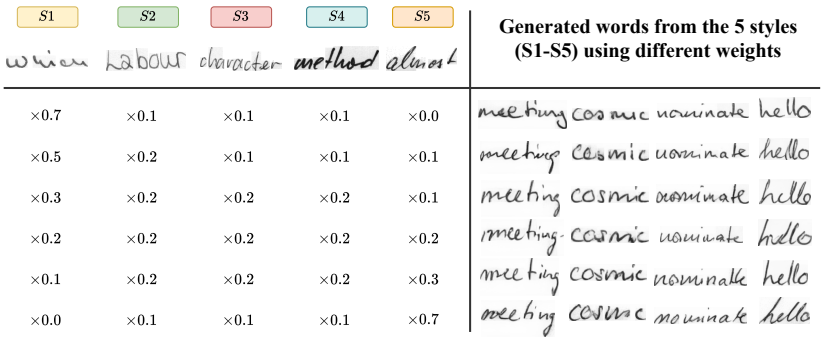
We present a more detailed analysis of the Handwriting Text Recognition (HTR) experiments present in our main paper using the CNN-LSTM HTR system presented in [8], which is trained with Connectionist Temporal Classification (CTC) loss [1]. We train the HTR system using the real data of the IAM training set and explore the effect of additional synthetic sets. Tab. 4 shows the results of the generated data used as additional data to the real training set. We observe that WordStylist [6] and our proposed method, DiffusionPen, show improved HTR performance in terms of Character Error Rate (CER) and Word Error Rate (WER), with WordStylist obtaining a slightly better performance. It should be noted that the HTR results presented in the original paper of WordStylist [6] are only on a subset of IAM that excludes punctuation and words smaller than 2 characters and larger than 10 characters for both train and test set. In our work, we have re-trained WordStylist [6], GANwriting [2] and SmartPatch [5], using the full character set. Thus, our obtained HTR performance for WordStylist differs from the one presented in the original paper [6].

We further explore the effect of the style variation data as an augmentation of the existing training set. We present results using the data generated using the noisy style embedding and the noise bias concepts mentioned in Sec. 3 in Tab. 5. We can see that both cases (Style Noise 0.25 and Noise Bias) can improve the performance of the HTR system; however, they cannot achieve a better performance than our standard method (see last row of Tab. 4).

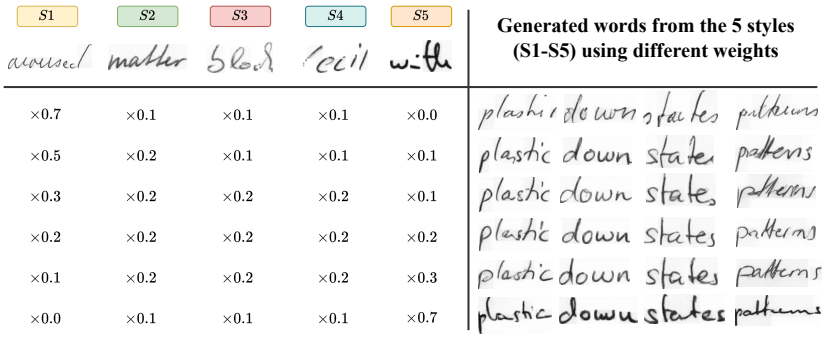
References

1. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A Novel Connectionist System for Unconstrained Handwriting Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(5), 855–868 (2008)
2. Kang, L., Riba, P., Wang, Y., Rusinol, M., Fornés, A., Villegas, M.: GANwriting: Content-Conditioned Generation of Styled Handwritten Word Images. In: European Conference on Computer Vision. pp. 273–289. Springer (2020)
3. Lee, A.W., Chung, J., Lee, M.: GNHK: A Dataset for English Handwriting in the Wild. In: Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part IV 16. pp. 399–412. Springer (2021)
4. Marti, U.V., Bunke, H.: A full english sentence database for off-line handwriting recognition. In: Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR '99 (Cat. No.PR00318). pp. 705–708 (1999). https://doi.org/10.1109/ICDAR.1999.791885

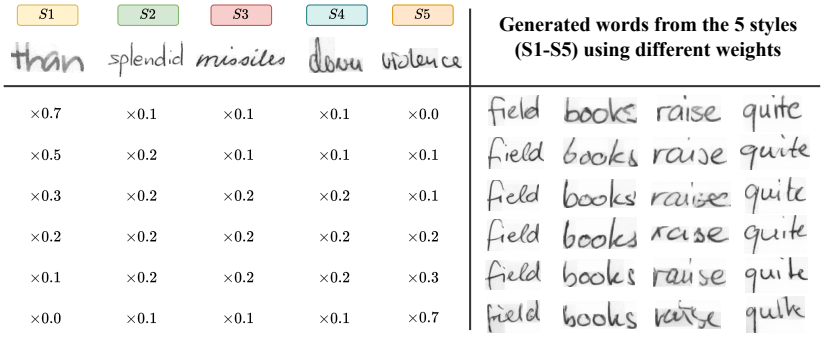
5. Mattick, A., Mayr, M., Seuret, M., Maier, A., Christlein, V.: Smartpatch: Improving handwritten word imitation with patch discriminators. In: International Conference on Document Analysis and Recognition. pp. 268–283. Springer (2021)
6. Nikolaidou, K., Retsinas, G., Christlein, V., Seuret, M., Sfikas, G., Smith, E.B., Mokayed, H., Liwicki, M.: Wordstylist: Styled verbatim handwritten text generation with latent diffusion models. In: International Conference on Document Analysis and Recognition. pp. 384–401. Springer (2023)
7. Pippi, V., Cascianelli, S., Cucchiara, R.: Handwritten Text Generation from Visual Archetypes. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 22458–22467 (2023), https://api.semanticscholar.org/CorpusID:257766680
8. Retsinas, G., Sfikas, G., Gatos, B., Nikou, C.: Best practices for a handwritten text recognition system. In: International Workshop on Document Analysis Systems. pp. 247–259. Springer (2022)



(a) Style Mixture between 5 styles $S1 - S5$.



(b) Style Mixture between 5 styles $S1 - S5$.



(c) Style Mixture between 5 styles $S1 - S5$.

Fig. 9: Generated samples by combining five different writing styles.

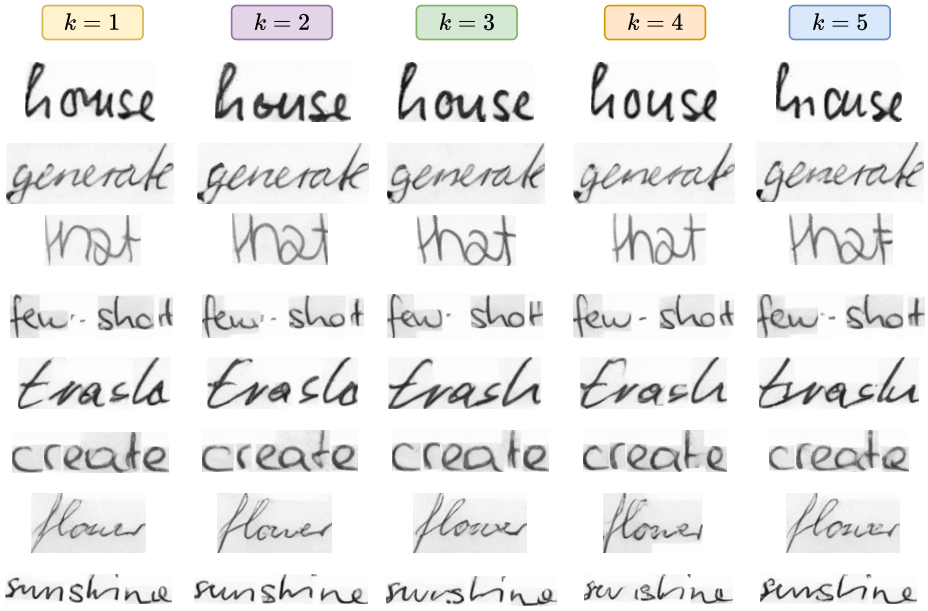


Fig. 10: Effect of the number of style samples during sampling.