Enhancing Realism in Batch Distillation Simulations: Data-Efficient Time Series Style Transfer with Transformers

Justus Will[1], Justus Arweiler[2] Indra Jungjohann[2] Jennifer Werner[3], Mayank Nagda[2], Michael Bortz[3], Hans Hasse[2], Fabian Jirasek[2], Jochen Schmid[3], Marius Kloft[2], Stephan Mandt[1], and Sophie Fellenz[2]

[1] University of California, Irvine, USA
[2] RPTU Kaiserslautern-Landau, Germany
[3] Fraunhofer Institute for Industrial Mathematics ITWM, Kaiserslautern, Germany

Abstract. In chemical process engineering, the accuracy and realism of simulated data are crucial for the effective design and optimization of a wide range of processes. In this paper, we demonstrate the efficacy of neural style transfer methods to enhance the realism of time series data generated from simulations. Specifically, this machine learning technique allows us to learn the style characteristics of non-parallel experimental data obtained from real-world chemical plants and then use them to transform simulated data to more closely reflect the realistic behaviors and variabilities not captured by the simulation model. We propose a transformer-based architecture with a latent representation that disentangles content and style information. After training, the underlying generative model allows for fast and data-efficient stylized generation without requiring many iterations of gradient-based optimization per sample, as in other time series style transfer baselines. We show the efficacy of our approach on both synthetic data and in an application to batch distillation.

Keywords: Generative Models · Style Transfer · Disentanglement · Enhanced Realism · Chemical Process Engineering

1 Introduction

In many scientific applications, we can produce simulated data of the measurable process of interest. However, simulations make simplifying assumptions that result in over-simplified predictions that do not account for all real-world effects, environmental conditions and imperfections. They are often systematically biased and produce inaccurate results under specific conditions. Further processing of the simulation data, such as in deep learning models that are trained on this data to perform downstream tasks such as anomaly detection, requires the data to be as realistic as possible to make robust predictions that are still useful when applied to real experimental data. Therefore, to make the simulation data more realistic and augment existing experimental data sets, the simulation data needs

to be adapted. In this paper, we demonstrate this in an application to batch distillation.

Batch processes, such as batch distillations, play a vital role in flexible small-scale production processes which are relevant in, e.g., biotechnology or pharmaceutical industries. In a batch process, a feed mixture is introduced into the plant prior to start-up, and after completion of one plant-operation, the plant is emptied, then new feed is introduced [27, 3]. The behavior of the whole process is inherently transient, one operating point of a process translates to a plethora of different conditions in the plants. Researchers greatly benefit from the ability to rapidly and inexpensively evaluate their hypotheses before they are tested in practical experiments. Simulation studies are, therefore, one of the most important tools in process engineering.

There are two challenges in adapting simulation data to be more realistic. The first is that the amount of experimental data to learn from is typically limited since experiments are expensive. The second is that the data should be transformed without changing the main content of the data. Due to these challenges, we propose to view the problem at hand as a style transfer problem. In image style transfer [14, 11, 18, 8], an image is transformed to a different style without changing its content, typically based on a small set of reference images. This can be achieved, for example, by learning a latent representation that separates style and content. Then, during inference, we can adjust the style part of the latent representation independently from the content part. Specifically, our proposed method is based on stylized variational autoencoders and leverages separate content and style losses to disentangle the style and content information.

In this paper, we focus on the application of style transfer methods to time series. As in image style transfer, we define style as a set of characteristic features inherent in a given dataset and learn them from the data with a neural network. This permits the automatic extraction of style information from a reference style sample or dataset, even for complex style transfer tasks. Recent work [7, 12] has shown promising results but has been limited to slow iterative methods that require many iterations per sample. To summarize, our main contributions are:

- We propose a style transfer method based on a variational autoencoder with a disentangled latent space that is fast and well-suited for style transfer tasks involving time series.
- We present a data-efficient training scheme that allows us to effectively obtain stylized samples even in the absence of a large style dataset.
- We demonstrate the efficacy of the style transfer methodology to enhance realism of simulation data in a chemical process engineering application.

2 Background

2.1 Variational Autoencoder

A generative latent variable model first samples – from a prior $p(\mathbf{z})$ – a point $\mathbf{z}$ in a low-dimensional latent space, which then informs a conditional distribution

$p_\theta(\mathbf{x}|\mathbf{z})$ over the target domain. Most commonly, this is a Gaussian model with a mean that is parameterized through a neural network $\mu_\theta(\mathbf{z})$, providing a complex and flexible model that can approximate a large class of data distributions:

$$\begin{aligned} \mathbf{z} &\sim \mathcal{N}(\mathbf{0}, \mathcal{I}) \\ \mathbf{x}|\mathbf{z} &\sim \mathcal{N}\big(\mu_\theta(\mathbf{z}), \sigma^2\mathcal{I}\big) \end{aligned} \tag{1}$$

To fit the model to data, we minimize the negative evidence lower bound (NELBO) $-\mathcal{L}_\theta(q)$ that bounds the intractable negative marginal likelihood from above via

$$-\mathcal{L}_\theta(q) \coloneqq \mathbb{E}_{\mathbf{z}\sim q}\left[-\log \frac{p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q(\mathbf{z})}\right] = -\log p_\theta(\mathbf{x}) + \underbrace{\mathrm{KL}(q \,\|\, p_\theta(\mathbf{z}|\mathbf{x}))}_{\geq 0}. \tag{2}$$

using an approximation q to the posterior $p_\theta(\mathbf{z}|\mathbf{x})$. A variational autoencoder (VAE) [21] uses amortized variational inference [23, 2] to optimize θ and q jointly by setting, e.g., $q = q_\psi(\mathbf{z}) = \mathcal{N}\big(g_\psi(\mathbf{x}), \mathrm{Diag}(h_\psi(\mathbf{x})^2)\big)$, where $g_\psi(\mathbf{x})$ and $h_\psi(\mathbf{x})$ are neural networks. This allows us to optimize (2) for θ and ψ by minimizing

$$\mathcal{L}(\theta, \psi) \overset{\beta=1}{=} \mathbb{E}_{\mathbf{z}\sim q_\psi}\left[\frac{1}{2\sigma^2}\|\mathbf{x} - \mu_\theta(\mathbf{z})\|_2^2\right] + \beta\mathrm{KL}(q_\psi(\mathbf{z}) \,\|\, p(\mathbf{z}))$$

by first sampling $\mathbf{z}$ from the variational encoder $q_\psi(\mathbf{z})$ and then decoding the mean $\mu_\theta(\mathbf{z})$ of the data likelihood $p_\theta(\mathbf{x}|\mathbf{z})$. We then compute the NELBO as a sum of a reconstruction term and a KL term that regularizes the latent space and back-propagate the gradients. By letting β be a hyperparameter, we can determine the trade-off between reconstruction quality and latent space structure.

2.2 Style Transfer

Style transfer is a complex task due to the subjective and task-specific nature of style and realism. Style is defined in terms of distinctive characteristics in data, such as tone in text or color palette in images. There are two main approaches to formalizing style: The feature-based approach identifies specific features that define style, while the distinction-based approach focuses on the overall distinctiveness, often using methods like Generative Adversarial Networks (GANs) [16] to ensure stylized outputs are indistinguishable from real style samples. Here, we focus on the former. First, we define a feature map ϕ that maps from data space to feature space such that $\phi_i(\mathbf{x})_j$ is the value (i.e. activation) of feature ϕ_i at position j of data sample $\mathbf{x}$. Although we can specify ϕ by hand [24, 12], in complex style transfer tasks, the features are usually learned implicitly via an unrelated task, such as image classifications [14] or unsupervised representation learning, for example, with a denoising autoencoder [7]. After training a (convolutional) neural network on the specified tasks we can directly use its learned features as style features. Next, we define similarity in style and content via

$$\begin{aligned} \mathcal{L}_c(\mathbf{x}, \mathbf{y}) &\propto \sum_i \|\phi_i(\mathbf{x}) - \phi_i(\mathbf{y})\|_2^2 \\ \mathcal{L}_s(\mathbf{x}, \mathbf{y}) &\propto \sum_i \|\mu(\phi_i(\mathbf{x})) - \mu(\phi_i(\mathbf{y}))\|_2^2 + \|\sigma(\phi_i(\mathbf{x})) - \sigma(\phi_i(\mathbf{y}))\|_2^2, \end{aligned} \tag{3}$$

where $\mathcal{L}_c$ and $\mathcal{L}_s$ are the content loss and style loss between data points $\mathbf{x}$ and $\mathbf{y}$, respectively. Intuitively, in samples that are close in content, the locations of detectable features are similar, yielding similar feature activations and small $\mathcal{L}_c$. Conversely, the feature mean and standard deviation capture aggregated information, revealing abstract properties unrelated to image content. If samples are close in style, they share similar feature statistics, and $\mathcal{L}_s$ is small. Finally, after choosing appropriate features and losses, we formulate style transfer as an optimization problem: Given a data sample $\mathbf{x}$ and style samples $\mathbf{s}_i \sim p(\mathbf{s})$ find

$$\mathbf{y}' = \min_{\mathbf{y}} \lambda_c \mathcal{L}_c(\mathbf{x}, \mathbf{y}) + \lambda_s \sum_{i=1}^{N_s} \mathcal{L}_s(\mathbf{s}_i, \mathbf{y}). \tag{4}$$

Here λ_c and λ_s are hyperparameters that allow for control over the mentioned trade-off. Usually, only one sample suffices. There are many different approaches to finding solutions to (4) or similar optimization problems [19], generally falling into two categories: Data-based methods optimize (4) directly. Gatys et al. [14], for example, initialize $\mathbf{y}$ randomly and use gradient descent with back-propagated gradients. This is slow, as it needs to be repeated for every generated sample. Model-based approaches, on the other hand, generate stylized samples directly, training an end-to-end model $f_\theta(\mathbf{x})$ with a fixed style via

$$\min_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})} \left[\lambda_c \mathcal{L}_c(\mathbf{x}, f_\theta(\mathbf{x})) + \lambda_s \sum_{i=1}^{N_s} \mathcal{L}_s(\mathbf{s}_i, f_\theta(\mathbf{x})) \right]. \tag{5}$$

or with arbitrary style as learned from $S = \{\mathbf{s}_1, \ldots, \mathbf{s}_{N_s}\}$ via

$$\min_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})} \left[\lambda_c \mathcal{L}_c(\mathbf{x}, f_\theta(\mathbf{x}, S)) + \lambda_s \sum_{i=1}^{N_s} \mathcal{L}_s(\mathbf{s}_i, f_\theta(\mathbf{x}, S)) \right]. \tag{6}$$

On images, arbitrary style transfer has been implemented, for example, with CNNs [11, 18] or transformers [8].

3 Method

3.1 Stylized VAE

We want to adapt a VAE, to solve arbitrary style transfer as in (6). The commonly used Gaussian model (1), however, entails minimizing a MSE loss and is not suitable for the stylized reconstruction needed in style transfer tasks: As $p(\mathbf{x}|\mathbf{z})$ adds noise directly to our observation, noisy samples, which have a different style, are modeled with a high likelihood. Moreover, this causes a smooth and blurry mean that averages out several plausible reconstructions, hindering correct stylization. We circumvent this problem by adding noise directly in feature space. This gives us a generative model that allows for sufficiently complex

generation, fine control over stylization, and a mean that has average features, in a way, the most representative reconstruction possible. Specifically, we define

$$\begin{aligned} \mathbf{z} &\sim \mathcal{N}(0, \mathcal{I}) \\ \eta^c(\mathbf{x})|\mathbf{z} &\sim \mathcal{N}(\eta^c(f_\theta(\mathbf{z})), \sigma_c^2 \mathcal{I}) \\ \eta^s(\mathbf{x})|\mathbf{z} &\sim \mathcal{N}(\eta^s(f_\theta(\mathbf{z})), \sigma_s^2 \mathcal{I}), \end{aligned} \tag{7}$$

where $\eta^c(\mathbf{x}) = [\{\phi_i(\mathbf{x})_j\}_{ij}]$, $\eta^s(\mathbf{x}) = [\{\mu(\phi_i(\mathbf{x}))\}_i, \{\sigma(\phi_i(\mathbf{x}))\}_i]$, and $[\cdot]$ denotes the concatenation of all feature activations (or feature statistics) into one vector. As our features $\eta(\mathbf{x}) = [\eta^c(\mathbf{x}), \eta^s(\mathbf{x})]$ should contain sufficient information about the original sample $\mathbf{x}$, η is approximately invertible, i.e., there exists $\tilde{\eta}^{-1}$ with $\tilde{\mathbf{x}} \coloneqq \tilde{\eta}^{-1}(\eta(\mathbf{x})) \approx \mathbf{x}$. Then, although $p(\mathbf{x}|\mathbf{z})$ may not be uniquely defined (it is only if η is invertible), any potential candidate for $p(\mathbf{x}|\mathbf{z})$ is close in distribution to the approximate conditional $p(\tilde{\mathbf{x}}|\mathbf{z})$, which we could sample from via

$$x' = \tilde{\eta}^{-1}(\eta(f_\theta(\mathbf{z})) + w),\ w \sim \mathcal{N}(0, \boldsymbol{\sigma}^2 \mathcal{I}),$$

highlighting that we add noise in feature space, allowing $f_\theta(\mathbf{z})$ to have average features, as intended. To see the connection to style transfer, note that we can find optimal parameters θ and ψ by minimizing the corresponding NELBO

$$\mathcal{L}(\theta, \psi) = \mathbb{E}_{\mathbf{z} \sim q_\psi} \left[\lambda_c \mathcal{L}_c(\mathbf{x}, f_\theta(\mathbf{z})) + \lambda_s \mathcal{L}_s(\mathbf{x}, f_\theta(\mathbf{z}))\right] + \lambda_{\mathrm{KL}} \mathrm{KL}(q_\psi(\mathbf{z}) \,\|\, p(\mathbf{z})), \tag{8}$$

with $\mathcal{L}_c$ and $\mathcal{L}_s$ as in (3) which is a regularized version of (5), where $\mathbf{x}$ is both the content and style sample. Note that λ_c, λ_s, and λ_{KL} control the trade-off between content preservation, stylization, and latent space structure.

3.2 Disentanglement and Inference

The VAE defined above allows for stylized reconstruction, where both content and style information are encoded in $\mathbf{z}$. To solve arbitrary style transfer as in (6), we propose the use of *latent swapping*. That is, we define $\mathbf{z} = [\mathbf{z}^c, \mathbf{z}^s]$ as the concatenation of two latent spaces containing the relevant latent representation of content and style, respectively. After training, we then perform style transfer by swapping in a different style latent variable with

$$\begin{aligned} [\mathbf{z}^c(\mathbf{x}), \mathbf{z}^s(\mathbf{x})] &\sim q_\psi(\mathbf{z}|\mathbf{x}) \\ [\mathbf{z}^c(\mathbf{s}), \mathbf{z}^s(\mathbf{s})] &\sim q_\psi(\mathbf{z}|\mathbf{s}) \\ \mathbf{y} &= f_\theta(\mathbf{z}^c(\mathbf{x}), \mathbf{z}^s(\mathbf{s})), \end{aligned} \tag{9}$$

for a time series $\mathbf{x}$ and a style sample $\mathbf{s}$. The sample $\mathbf{y}$ is a good solution if both $\mathcal{L}_c(\mathbf{x}, \mathbf{y})$ and $\mathcal{L}_s(\mathbf{s}, \mathbf{y})$ are small. This is the case, for example, when $\mathbf{x} \approx \tilde{\mathbf{x}} = f_\theta(\mathbf{z}^c(\mathbf{x}), \mathbf{z}^s(\mathbf{x}))$, i.e., reconstruction quality is high for all $\mathbf{x}$, and content and style are properly disentangled/separated in the latent space, e.g., as quantized by low mutual information $\mathrm{I}(q_\psi(\mathbf{z}^c|\mathbf{x}), q_\psi(\mathbf{z}^s|\mathbf{x}))$. Then, since $\tilde{\mathbf{x}}$ contains the same content and style information as $\mathbf{x}$, $\mathbf{z}^c(\mathbf{x})$ must contain the content information,

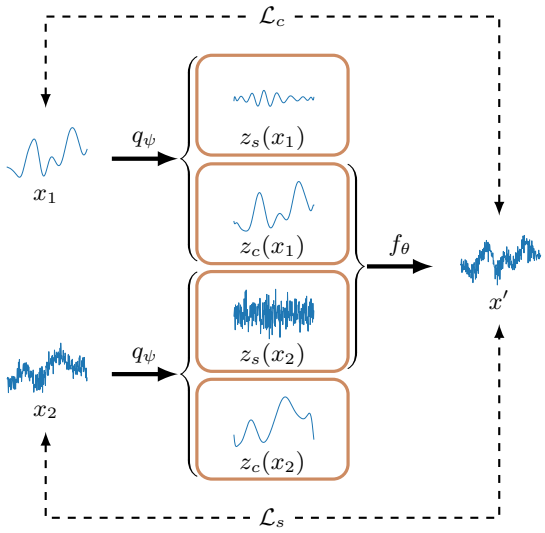


Fig. 1: A sketch of latent swapping during training and inference.

and $\mathbf{z}^s(\mathbf{x})$ must contain the style information. Even in cases where we are only able to obtain imperfect disentanglement (e.g. when the concepts of content and style are not fully independent), our method may still produce accurately stylized samples as long as f_θ ignores all residual style information in $\mathbf{z}^c(\mathbf{x})$ and content information in $\mathbf{z}^s(\mathbf{x})$.

To this end, we propose a new method of active disentanglement, guided by the two separate loss functions unique to style transfer. We perform latent swapping as in (9) during training and adapt the loss in (8) accordingly to:

$$\mathcal{L}(\theta, \psi) = \mathop{\mathbb{E}}_{\substack{\mathbf{z}^c \sim q_\psi(\cdot|\mathbf{x}_1) \\ \mathbf{z}^s \sim q_\psi(\cdot|\mathbf{x}_2)}} [\lambda_c \mathcal{L}_c(\mathbf{x}_1, f_\theta(\mathbf{z})) + \lambda_s \mathcal{L}_s(\mathbf{x}_2, f_\theta(\mathbf{z}))] + \lambda_{\mathrm{KL}} \mathrm{KL}(q_\psi(\mathbf{z}) \,\|\, p(\mathbf{z})),$$

with $\mathbf{z} = [\mathbf{z}^c, \mathbf{z}^s]$. This is illustrated in Figure 1 and has two advantages. First, we directly optimize the style transfer problem (6), which justifies our approach mathematically, and second, we tailor our disentanglement to the specific notion of content and style implicitly defined through the content and style loss. Lastly, if $\mathbf{z}^c(\mathbf{x}_1)$ and $\mathbf{z}^s(\mathbf{x}_2)$ contain undesired residual information, f_θ learns to ignore this additional information.

3.3 Model architecture

To complete the specification of our model, we define the encoder $q_\psi(\mathbf{z}|\mathbf{x})$ and the decoder f_θ. We use transformer models [30] to allow our model to attend to long-range dependencies in the data and, therefore, to quickly generate correctly stylized time series even for complex styles. An overview of our model architecture can be found in Figure 2. Similar to GPT [25] and BERT [9] models, we only

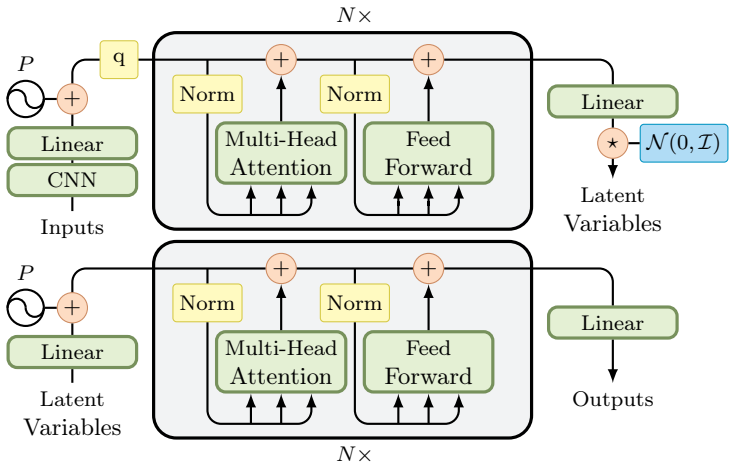


Fig. 2: The proposed transformer architecture. The model consists of two variational encoders (above) and the variational decoder (below). A query token q is appended only for the style encoder.

use a transformer encoder. We do not mask any connections in the self-attention layers and normalize before applying the attention and feed-forward layers to make training more stable [33]. Recent works [37, 36] that apply transformers to time series, embed data into a higher-dimensional space using a simple linear projection, but we found that additionally encoding context information using a convolution with a small kernel size of 5, improves training speed significantly. To this representation, we add positional information with a sinusoidal positional encoding P as in [30]. Similar to DSAE [34], our model allows for style transfer by disentangling content and style in the latent space. The latent space thus consists of a time-dependent component $\mathbf{z}^c \in \mathbb{R}^{d_{\text{len}} \times d^c_{\text{latent}}}$ for content and a time-invariant component $\mathbf{z}^s \in \mathbb{R}^{d^s_{\text{latent}}}$ for global style information. We use one transformer per component to jointly learn both the mean and the standard deviation of the latent variables. For the time-dependent $\mathbf{z}^c$ the parameters are obtained through a linear projection of the learned transformer representation at the respective position; for the time-invariant $\mathbf{z}^s$ a query-based approach similar to Carion et al. [6] is used. To this end, we append an all-zero query token to the beginning of the embedded time series, extend each position to include a query flag, and extract the latent parameters from the value accumulated in the query position with a linear projection. After sampling, the latent variables are again embedded with a linear projection and positional encoding.

3.4 Pretraining

Transformer models are very flexible and powerful but they also need a lot of training data, particularly big models like GPT and BERT [25, 9]. There-

fore, these models are first trained on unsupervised tasks with a large unlabeled dataset. Although our transformer model is much more compact, proper pretraining still has the potential to improve performance and to decrease the required amount of data significantly [29]. Similar to Denoising Autoencoders (DAE) [31], we pretrain our VAE to reconstruct the original time series from corrupted inputs. In our case, the inputs are corrupted by masking and adding noise to chunks of a fixed length at random positions. Specifically, following the BERT paper, we mask each chunk with a probability of 16% and add zero-mean standard Gaussian noise to it with a probability of 2%. To mask a position of our input, we set the embedded representation to an all-zero mask token and extend every position in the embedding by a mask flag.

4 Application of the Method

To show the effectiveness of our style transfer method, we first test on synthetic data where we have fine control and unlimited data. Then, we demonstrate the ability to enhance realism in an application to batch distillation data. We obtain style features by training a denoising autoencoder on the full combined dataset of content and style samples. For both experiments, we compare our model-based approach using Disentangled Variational Autoencoder (DVAE) with iterative direct optimization (IDO) of (4) as in [7]. As the notions of style are not very complex, we assume that IDO will give results that are close to optimal. However, we show that our approach produces samples much faster since no iterative optimization is required while only sacrificing a minimal amount of quality.

4.1 Evaluation

Objectively evaluating the quality of stylized generation is difficult, as we care about multiple contradicting goals, including content preservation, stylization, and utility as data augmentation for downstream tasks. We can control some of this trade-off with the loss weights λ_c and λ_s. The most obvious metrics for content preservation and style alignment are the feature-based content loss $\mathcal{L}_c$ and the style loss $\mathcal{L}_s$. These metrics, however, might not be robust to optimization and, therefore, not give an objective assessment, especially as they are highly dependent on the feature's ability to correctly specify the content and style. For a more intuitive metric of content preservation, we compare time series trends via the mean square error between moving averages (MASE). We obtain an unbiased assessment of style alignment with α-precision and β-recall scores [26], which measure realism and diversity, respectively. Formally, the distribution over our generated samples Q has precision α at recall β w.r.t. the true data distribution P if there exists distributions ν_P, ν_Q, and μ such that

$$P = \beta\mu + (1-\beta)\nu_P \quad \text{and} \quad Q = \alpha\mu + (1-\alpha)\nu_Q.$$

Intuitively, the α-precision is the proportion of Q that can be explained by an approximation of P whose quality increases as β increases. If α stays high for

increasing values of β, this implies that Q can be represented well by P, i.e., the generated samples are realistic. Conversely, a high β-recall implies that P can be represented well by Q, i.e., the generated samples are diverse. We find estimates from samples by clustering the joint set of generated samples and style samples and interpreting our distributions as distributions over class labels. As a crude quantifier, we report the average precision and recall over all lines through the origin. However, a visual inspection of the precision-recall curve remains essential. We refer to [26] for details.

In addition, we assess the style quality through its indistinguishability from real style samples. To this end, we train a classifier to distinguish between generated and real style samples. We fit a standard logistic regression model to half of the validation set and report the accuracy and recall on the other half. For stylized samples that closely resemble true style samples, the accuracy and recall should be close to 50%. Low recall indicates high realism as the recall is related to how many generated samples are mistaken for real ones. Lastly, we consider the predictive utility on the style dataset. Following the TSTR ('train on synthetic, test on real') framework [13], we train a simple 2-layer LSTM to perform a prediction task on the generated samples. If the trained model generalizes well to the style data, this indicates high fidelity and utility as data augmentation. We report the mean absolute error (MAE).

4.2 Toy data

Enhancing realism in simulation data is essentially equivalent to learning and adding the realistic noise found in real data, attributable, for example, to measurement error. We emulate a simple univariate case by modeling the simulation data of interest as samples from a zero-mean Gaussian process,

$$\mathbf{x} \sim \mathcal{N}(0, K),\ K_{ij} = \exp\left(-\frac{1}{2\rho^2}|i-j|^2\right),$$

with RBF kernel of bandwidth $\rho^2 = 20$. This gives us smooth and aperiodic time series that still exhibit some complexity in their contents. We assume that this perfectly models the underlying phenomena and experimental style samples are thus obtained from the same distribution, only adding measurement error modeled as Gaussian white noise with noise level $\sigma_s^2 = 0.5$, i.e.,

$$\mathbf{s} = \mathbf{c} + \mathbf{n},\ c \sim \mathcal{N}(0, K),\ \mathbf{n} \sim \mathcal{N}(0, \sigma_s^2 \mathcal{I}).$$

Here, style and content are completely separable as they are independent. We sample 50000 steps from the smooth and noisy Gaussian processes and split them into overlapping chunks of length 256, reserving 10% as hold-out validation and test sets. For completeness, we consider style transfer in both directions and train our model twice, separately optimizing the hyperparameters. Tables 1 and 2 show the quantitative evaluation of the style transfer task across all metrics. Figures 6a and 6b show the diversity-realism curves, and Figures 3 and 4 show randomly selected samples. IDO performs well, as to be expected. It generates realistic,

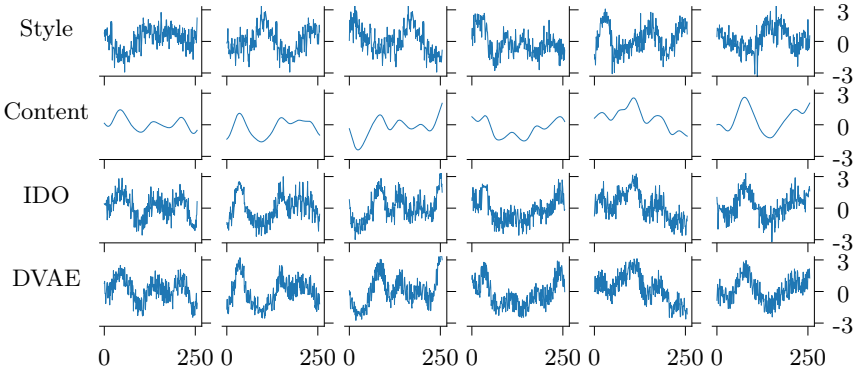


Fig. 3: Style transfer from smooth to noisy data, randomly chosen samples. Our method successfully learns to add Gaussian noise.

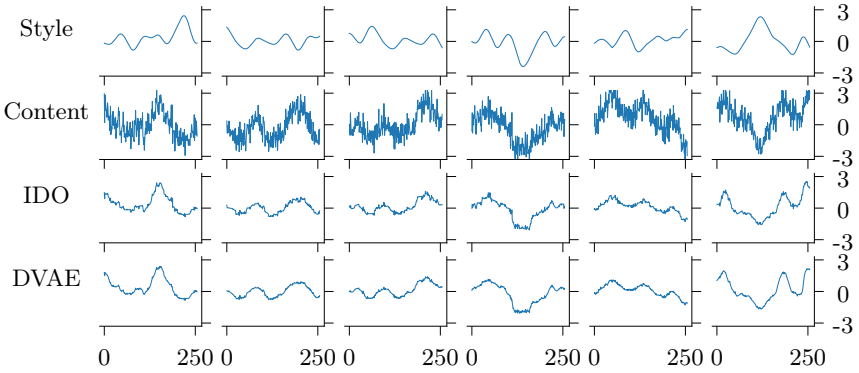


Fig. 4: Style transfer from noisy to smooth data, randomly chosen samples. Our method successfully learns to remove Gaussian noise.

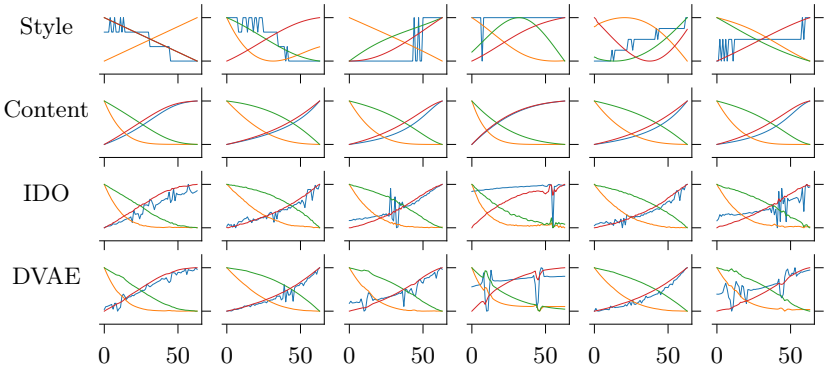


Fig. 5: Style transfer from simulated to realistic data, randomly chosen samples. Our method successfully replicates the spikes and characteristic noise of the style samples, while preserving the original content.

	content		style					
	$\mathcal{L}_c \downarrow$	MASE$\downarrow$	$\mathcal{L}_s \downarrow$	$\alpha^* \uparrow$	$\beta^* \uparrow$	Acc$\downarrow$	Rec$\downarrow$	MAE$\downarrow$
IDO	$3.88 \cdot 10^{-2}$	0.64	$6.69 \cdot 10^{-4}$	71.02%	71.06%	**50.5%**	**51.5%**	$6.95 \cdot 10^{-1}$
DVAE	$2.69 \cdot 10^{-2}$	0.37	$3.23 \cdot 10^{-3}$	70.92%	70.92%	92.7%	88.9%	$\mathbf{6.73 \cdot 10^{-1}}$

Table 1: Results on style transfer from smooth to noisy data, $\lambda_s/\lambda_c = 10$.

	content		style					
	$\mathcal{L}_c \downarrow$	MASE$\downarrow$	$\mathcal{L}_s \downarrow$	$\alpha^* \uparrow$	$\beta^* \uparrow$	Acc$\downarrow$	Rec$\downarrow$	MAE$\downarrow$
IDO	$2.63 \cdot 10^{-2}$	0.40	$4.38 \cdot 10^{-4}$	71.03%	71.08%	52.6%	**50.7**%	$1.10 \cdot 10^{-1}$
DVAE	$2.73 \cdot 10^{-2}$	0.39	$7.10 \cdot 10^{-4}$	70.84%	70.91%	53.0%	55.8%	$\mathbf{5.42 \cdot 10^{-2}}$

Table 2: Results on style transfer from noisy to smooth data, $\lambda_s/\lambda_c = 10$.

diverse, and indistinguishable time series, as indicated by high α-precision, high β-recall, and accuracy close to 50%. Our method successfully produces samples of similar high quality in a fraction of the time, with scores that are less than 0.5% lower. We note, however, that our method produces noisy samples that are more distinguishable from style samples than the baseline. A visual inspection of the generated time series in Figure 4 shows no obvious defects, so this may be due to artifacts in the generated samples that are not visually distinct but can be found by the classifier. This problem could potentially be tackled with adversarial training techniques. Arguably, this is not a significant problem as our generated samples still demonstrate high fidelity and high predictive utility, in the latter even beating the iterative baseline.

4.3 Batch Distillation Data

We explore an application in chemical process engineering: batch distillation [27, 15, 3, 20]. In batch distillation, a mixture of two or more components is separated into two or more products: distillate fractions and a bottom product or residue. To this end, a batch of a liquid mixture is charged to a pot and heated until it is boiling. The resulting vapor rises through a column and is condensed. A part of this condensate refluxes into the column. The rest is drawn out to produce a distillate. This is a non-stationary process, even for control inputs that are chosen to be constant over time, as both the compositions of components and temperature change over time. Here, we have access to two non-parallel datasets: Simulation data and sensor data recorded in a real batch distillation plant. We aim to enhance realism in the former by learning the style of the latter.

Simulation Data We consider a batch distillation column with S equilibrium stages and a condenser above these stages. A schematic of the model is shown in Figure 7a. The simulation model is described by the so-called MESH equations

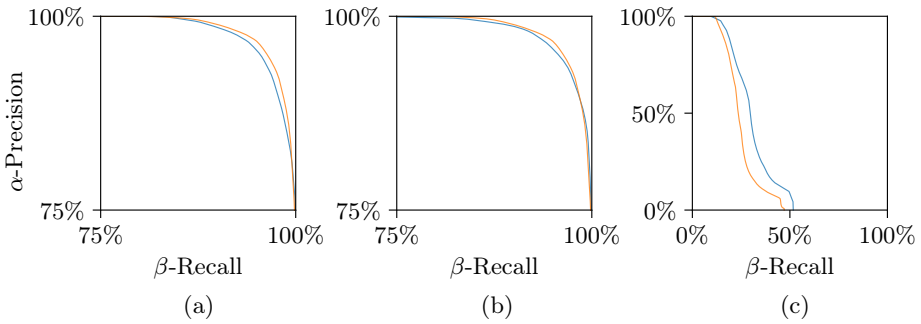
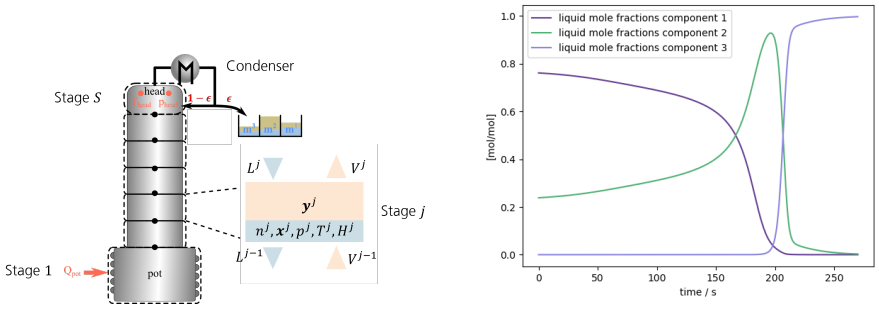


Fig. 6: α-Precision and β-Recall for different style transfer tasks, from a) smooth to noisy toy data, b) noisy to smooth toy data, and c) simulated to realistic batch distillation data. Our method (DVAE, in orange) closely matches the sample quality of the iterative baseline (IDO, in blue).

[27, 3] on each stage of the batch distillation column, describing mass balances, thermodynamic equilibria, summations, and enthalpy balances. We assume zero vapor holdups on all stages, total condensation in the condenser, and that one liquid and one vapor phase are in thermodynamic equilibrium with each other at all stages at all times; in particular, we do not consider liquid-liquid equilibria or all-liquid or all-vapor regimes. The resulting equations are a combination of nonlinear differential and algebraic equations and, hence, constitute a system of differential-algebraic equations [1, 22, 5]. We applied a suitable index reduction and implemented the index-reduced version of the system of equations in Python using the optimization modeling library *Pyomo* [4, 17], solved with *Ipopt* [32].

We generate 26 simulations with a mixture of acetone, methanol, and butanol, setting $S = 8$. The simulations differ in the initial composition of the components in the pot, and each simulation includes four signals (time series): The liquid mole fractions for the three components and the temperature in the head of the column. For an example simulation, refer to Figure 7b.

Experimental Data We conducted several distillation experiments with a laboratory-sized batch-distillation plant. The plant is equipped with a glass pot and a put-on glass column with a height of 150 cm. 27 sensors, including temperature, pressure, level and flow sensors, as well as analytics of mixture compositions collect time-series data while conducting experiments. We performed 25 batch distillation experiments with mixtures of 1-butanol, 2-propanol, and water with varying compositions, gathering data for 1000 to 10000 time steps. The mixtures were heated, the uprising vapor was cooled down in a condenser on top of the column, and a part of the vapor was collected as distillate. The remainder refluxes into the column, enabling efficient separation of components.



(a) (b)

Fig. 7: Schematic of a batch distillation column with S stages (a) and example of simulated liquid mole fractions in the head of batch distillation column for a mixture of acetone, methanol, and butanol (b).

	content		style					
	$\mathcal{L}_c \downarrow$	MASE$\downarrow$	$\mathcal{L}_s \downarrow$	$\alpha^* \uparrow$	$\beta^* \uparrow$	Acc$\downarrow$	Rec$\downarrow$	MAE$\downarrow$
IDO	$2.33 \cdot 10^{-5}$	$1.01 \cdot 10^{-2}$	$8.57 \cdot 10^{-4}$	38.49%	28.46%	**93.5%**	**90.5%**	0.25
DVAE	$1.74 \cdot 10^{-5}$	$7.91 \cdot 10^{-3}$	$9.25 \cdot 10^{-4}$	34.79%	26.06%	93.3%	89.5%	0.25

Table 3: Results on style transfer from simulated to realistic data, $\lambda_s/\lambda_c = 1$.

Style Transfer Results To align the different datasets, we first remove the trend by computing and removing a moving average and normalizing the time series with that trend's mean and standard deviation. After performing style transfer on the normalized data, we undo normalization and re-add the trend of the content sample. We split each time series into overlapping chunks of length $d_{\text{len}} = 64$, reserving 10% of data as hold-out validation and test sets. Table 3 shows the quantitative evaluation of the style transfer task across all metrics. Figure 6c shows the diversity-realism curve, and Figure 5 shows randomly selected samples, with values normalized to a range between 0 and 1. In this more complex setting, IDO still performs reasonably well. It captures the main difference between the two datasets, adding noise to the simulated temperature values. Real sensors capture temperature data across finite values due to their finite accuracy, resulting in time series with sudden jumps and spikes, which are visible in the stylized samples too. Lower values for α-precision, β-recall, and a higher accuracy indicate the increased complexity of this style transfer application. Nonetheless, our method is again able to successfully produce samples of a similar high quality in a fraction of the time.

5 Related work

Thus far, the application of style transfer methods to time series has been limited. Most approaches adopt feature-based methods from image style transfer, for example, by directly transforming time series to images [28]. This approach is limited to high-dimensional time series and restricts the class of learnable styles, as features trained on image classification are not well-suited for most time series applications. The choice of features in (6) is crucial. El-Lahman et al. [12] propose using hand-crafted features, specifically designed for their finance application. Da Silva et al. [7] use learned features that are obtained via a denoising autoencoder. Both methods are model-free, iteratively optimizing a random initialization with back-propagated gradients.

When samples of the target style are abundant, stylized time series may be generated directly, for example, with VAEs [7, 10] or GANs [13, 35, 38]. This, however, usually sacrifices control over the exact content. Control is maintained when using an autoencoder with a disentangled representation, allowing for *latent swapping* as described earlier. For high-dimensional time series such as video and speech, disentanglement can be obtained with a Disentangled Sequential Autoencoder (DSAE) [34] that splits the time series into global information (style) and time-dependent dynamics (content). If the dimension of the latent spaces are chosen appropriately, a good reconstruction is only possible by storing the time-invariant information in the latent space that encodes the global information, permitting inference-time style transfer. In our case, this is not feasible because even the smallest style latent space of size 1 might still not be restrictive enough for the univariate or low-dimensional time series we work with.

6 Conclusion

In this paper, we have investigated the utility of neural style transfer methodology to enhance realism in simulation data by learning the style of non-parallel experimental data. Our proposed model-based style transfer framework is based on loss-guided disentanglement and allows for fast real-time style transfer without iterative optimization. We have shown the efficacy of our method, both qualitatively and quantitatively, in a simplified synthetic setting and application in batch distillation. Our method is able to correctly identify the correct notion of style or realism and provide stylized reconstructions of data samples while still preserving their contents. In both settings, we closely match the sample quality of the iterative baseline that provides near-optimal solutions to the optimization problem defining the style transfer task. We can do so with only one model call to a small transformer model instead of many iterations of gradient descent per generated sample, greatly improving utility as real-time data augmentation.

Acknowledgements

The authors acknowledge support by the Carl-Zeiss Foundation, the BMBF award 01IS20048, and the DFG awards BU 4042/2-1 and BU 4042/1-1.

References

1. Ascher, U.M., Petzold, L.R.: Computer methods for ordinary differential equations and differential-algebraic equations. SIAM (1998)
2. Bamler, R., Zhang, C., Opper, M., Mandt, S.: Perturbative black box variational inference. Advances in Neural Information Processing Systems **30** (2017)
3. Biegler, L.T., Grossmann, I.E., Westerberg, A.W.: Systematic methods for chemical process design (1997)
4. Bynum, M.L., Hackebeil, G.A., Hart, W.E., Laird, C.D., Nicholson, B.L., Siirola, J.D., Watson, J.P., Woodruff, D.L.: Pyomo–optimization modeling in python, vol. 67. Springer Science & Business Media, third edn. (2021)
5. Campbell, S., Ilchmann, A., Mehrmann, V., Reis, T.: Applications of differential-algebraic equations: examples and benchmarks. Springer (2019)
6. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision. pp. 213–229 (2020)
7. Da Silva, B., Shi, S.S.: Style transfer with time series: Generating synthetic financial data. arXiv:1906.03232 (2019)
8. Deng, Y., Tang, F., Pan, X., Dong, W., Ma, C., Xu, C.: Stytr^2: Unbiased image style transfer with transformers. arXiv:2105.14576 (2021)
9. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805 (2018)
10. Dogariu, M., Ştefan, L.D., Boteanu, B.A., Lamba, C., Kim, B., Ionescu, B.: Generation of realistic synthetic financial time-series. Multimedia Computing, Communications, and Applications **18**, 1–27 (2022)
11. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artistic style. In: International Conference on Learning Representations (2017)
12. El-Laham, Y., Vyetrenko, S.: Styletime: Style transfer for synthetic time series generation. arXiv:2209.11306 (2022)
13. Esteban, C., Hyland, S.L., Rätsch, G.: Real-valued (medical) time series generation with recurrent conditional gans. arXiv:1706.02633 (2017)
14. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Computer Vision and Pattern Recognition. pp. 2414–2423 (2016)
15. Gmehling, J., Kleiber, M., Kolbe, B., Rarey, J.: Chemical thermodynamics for process simulation. John Wiley & Sons (2019)
16. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
17. Hart, W.E., Watson, J.P., Woodruff, D.L.: Pyomo: modeling and solving mathematical programs in python. Mathematical Programming Computation **3**(3), 219–260 (2011)
18. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: International Conference on Computer Vision. pp. 1501–1510 (2017)
19. Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., Song, M.: Neural style transfer: A review. Transactions on Visualization and Computer Graphics **26**, 3365–3385 (2019)
20. King, C.J.: Separation processes. Courier Corporation (2013)
21. Kingma, D., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (2014)

22. Kunkel, P.: Differential-algebraic equations: analysis and numerical solution, vol. 2. European Mathematical Society (2006)
23. Marino, J., Yue, Y., Mandt, S.: Iterative amortized inference. In: International Conference on Machine Learning. pp. 3403–3412. PMLR (2018)
24. Neumann, L., Neumann, A.: Color style transfer techniques using hue, lightness and saturation histogram matching. In: Computer Aesthetics in Graphics, Visualization and Imaging. pp. 111–122 (2005)
25. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
26. Sajjadi, M.S., Bachem, O., Lucic, M., Bousquet, O., Gelly, S.: Assessing generative models via precision and recall. Neural Information Processing Systems **31** (2018)
27. Seader, J.D., Henley, E.J., Roper, D.K.: Separation process principles: Chemical and biochemical operations. Wiley, Hoboken NJ, 3rd ed. edn. (2011)
28. Takemoto, N., de M. Araújo, L., Coimbra, T.A., Tygel, M., Avila, S., Borin, E.: Enriching synthetic data with real noise using neural style transfer. In: International Congress of the Brazilian Geophysical Society. vol. 78 (2019)
29. Turc, I., Chang, M.W., Lee, K., Toutanova, K.: Well-read students learn better: On the importance of pre-training compact models. arXiv:1908.08962 (2019)
30. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Neural Information Processing Systems **30**, 6000–6010 (2017)
31. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: International Conference on Machine learning. pp. 1096–1103 (2008)
32. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical programming **106**, 25–57 (2006)
33. Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., Liu, T.: On layer normalization in the transformer architecture. In: International Conference on Machine Learning. pp. 10524–10533 (2020)
34. Yingzhen, L., Mandt, S.: Disentangled sequential autoencoder. In: International Conference on Machine Learning. pp. 5670–5679 (2018)
35. Yoon, J., Jarrett, D., Van der Schaar, M.: Time-series generative adversarial networks. Neural Information Processing Systems **32** (2019)
36. Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C.: A transformer-based framework for multivariate time series representation learning. In: Conference on Knowledge Discovery & Data Mining. pp. 2114–2124 (2021)
37. Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: AAAI Conference on Artificial Intelligence. vol. 35, pp. 11106–11115 (2021)
38. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: International Conference on Computer Vision. pp. 2223–2232. IEEE (2017)