

StyleTime: Style Transfer for Synthetic Time Series Generation

Yousef El-Laham
yousef.el-laham@jpmchase.com
J.P. Morgan AI Research
New York, New York, USA

Svitlana Vyetenko
svitlana.vyetenko@jpmchase.com
J.P. Morgan AI Research
New York, New York, USA

ABSTRACT

Neural style transfer is a powerful computer vision technique that can incorporate the artistic “style” of one image to the “content” of another. The underlying theory behind the approach relies on the assumption that the style of an image is represented by the Gram matrix of its features, which is typically extracted from pre-trained convolutional neural networks (e.g., VGG-19). This idea does not straightforwardly extend to time series stylization since notions of style for two-dimensional images are not analogous to notions of style for one-dimensional time series. In this work, a novel formulation of time series style transfer is proposed for the purpose of synthetic data generation and enhancement. We introduce the concept of stylized features for time series, which is directly related to the time series realism properties, and propose a novel stylization algorithm, called StyleTime, that uses explicit feature extraction techniques to combine the underlying content (trend) of one time series with the style (distributional properties) of another. Further, we discuss evaluation metrics, and compare our work to existing state-of-the-art time series generation and augmentation schemes. To validate the effectiveness of our methods, we use stylized synthetic data as a means for data augmentation to improve the performance of recurrent neural network models on several forecasting tasks.

KEYWORDS

synthetic time series, neural style transfer, time series augmentation

ACM Reference Format:

Yousef El-Laham and Svitlana Vyetenko. 2022. StyleTime: Style Transfer for Synthetic Time Series Generation. In *3rd ACM International Conference on AI in Finance (ICAIF '22)*, November 2–4, 2022, New York, NY, USA. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3533271.3561772>

1 INTRODUCTION

Synthetic generation of labeled images has successfully been applied to improve generalization capability, and, therefore, performance of machine learning image detection algorithms. Little work, however, has been done on using synthetic time series to improve generalization capability of machine learning forecasting algorithms. In financial services, for example, there is interest in augmenting historical stock price time series with synthetic rare event

data in order to enable price prediction models generalize better on unseen scenarios during the unexpected market shocks due to economic events, global pandemic, flash crashes, etc. . One generally expects models trained on such augmented time series to perform well in practice if synthetic time series is statistically similar to real out-of-sample data as suggested by the “train on synthetic, test on real” (TSTR) framework [8]. Moreover, statistical realism of the augmented time series dataset is often a requirement for explainability purposes in applications such as finance.

Multi-agent simulation provides a means of generating synthetic time series using a bottom-up approach. In multi-agent market simulation, the resultant time series originate from interaction of a large number of market agents (e.g., [4]). Because of the flexibility to change market configuration on the agent level, multi-agent simulation is well-suited for modeling counterfactual scenarios, however, it is notoriously difficult to calibrate to reflect realistic properties of the market [22].

In this paper, we propose to use neural style transfer (NST) to improve statistical realism of synthetic time series, and use it to stylize time series that model counterfactual scenarios for the purpose of data augmentation. NST was initially introduced in [12] to incorporate artistic style into photographs and quickly won over the imaging community. By including the feature correlations of multiple layers, one can obtain a stationary, multi-scale representation of the input image (i.e., the Gram matrix), which captures its texture information but not the global arrangement – which is called style representation. Despite receiving attention by the computer vision community, there is little academic work on NST in the context time series. In [20], NST was used to introduce realistic noise to the seismic shock model. In [19], denoising autoencoder combined with NST was used as a generative technique for realistic daily price time series. The generated time series showed statistical properties similar to historical data; additionally, the visual inspection of generated paths indicated presence of technical patterns that are characteristic of the historical daily asset time series. Both of the aforementioned methods extract time series style representations in the same manner they are extracted for images in [12], which may not be suitable notions of styles for realistic time series since they only contain information about correlations across extracted features. Further related work also includes NST for audio and voice synthesis. This class of NST methods utilize a variety of different techniques for extracting style representations including frequency-domain representations (e.g., VGG-19 or wide-shallow-random networks applied to 2D spectrograms [21]), time-domain representations (e.g., SoundNet encoder [3] or WaveNet decoder [17] applied to raw waveforms), and domain expert knowledge of auditory perception [13]. Unfortunately, since these style representations are specific to audio data, they cannot be directly applied to other domains, such as finance.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ICAIF '22, November 2–4, 2022, New York, NY, USA

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9376-8/22/10...\$15.00

<https://doi.org/10.1145/3533271.3561772>

Main Contributions: We develop a novel framework for time series stylization with the goal of constructing realistic and useful synthetic datasets. For that, we propose a style transfer method that combines the underlying trend of one time series with the distributional properties of another. These distributional properties, which we refer to as *stylized features*, are informed by domain knowledge of time series data and can be computed using sample-based approximations, without requiring the training of dedicated feature extraction techniques like convolutional neural networks (CNNs). Trends are extracted by applying time series decomposition on either the original time series dataset or a synthetic dataset generated from another model. To validate our method, we empirically show on three different datasets that our time series style transfer method is able to achieve competitive performance with state-of-the-art methods in terms of fidelity, predictive utility, and authenticity.

Related Work: Realistic time series generation has been previously studied in the literature by using the generative adversarial networks (GANs). With the TimeGAN architecture [27], realistic generation of temporal patterns was achieved by jointly optimizing with both supervised and adversarial objectives to learn an embedding space. QuantGAN [26] consists of a generator and discriminator functions represented by temporal convolutional networks, which allows it to synthesize long-range dependencies such as the presence of volatility clusters that are characteristic of financial time series. TimeVAE [6] was recently proposed as a variational autoencoder alternative to GAN-based time-series generation. In [1], Fourier flows, a flow-based generative model for time-series in the Fourier domain, is presented. Fourier flows are able to achieve competitive performance with state-of-the-art generation techniques at much lower training times. Methods like GANs, VAEs and Fourier flows are typically used for learning the underlying distribution of the training data, and may not provide the distributionally new scenarios needed for data augmentation.

Data augmentation is well established in computer vision tasks due to the simplicity of label-preserving geometric image transformation techniques, but it is still not widely used for time series with some early work being discussed in the literature [15]. For example, simple augmentation techniques applied to financial price time series such as adding noise or time warping were shown to improve the quality of next day price prediction model [9], however, such transformations were not required to produce realistic synthetic time series. In the computer vision literature, NST was proposed to disassociate content image from style, and hence generate a dataset with new content-style combinations - by utilizing this idea, NST was used as an augmentation technique to improve quality of machine learning classification algorithms for images [28]. In this paper, we investigate the use of NST for realistic time series generation and subsequent data augmentation.

2 BACKGROUND

NST is an algorithm that was introduced in [12] as a means to combine the content of a photograph with the style of an artwork. The theory behind the approach relies on the assumption that CNNs have the capability to capture representations of both the “content” and the “style” of an image. Mathematically speaking, NST is an optimization problem, where the goal is to minimize a loss function

with respect to an input image $Z \in \mathcal{Z}$ (e.g., an RGB image). The loss function consists of two different components: the content loss and the style loss. The content loss $\mathcal{L}_c(Z, Z_c)$ penalizes values of Z that differ from a considered content image $Z_c \in \mathcal{Z}$, while the style loss $\mathcal{L}_s(Z, Z_s)$ penalizes values of Z whose styles differ from a considered style image $Z_s \in \mathcal{Z}$. The NST objective function is a weighted sum of the content and style losses:

$$\mathcal{L}(Z, Z_c, Z_s) = \alpha \mathcal{L}_c(Z, Z_c) + \beta \mathcal{L}_s(Z, Z_s), \quad (1)$$

where the hyperparameters $\alpha, \beta \in \mathbb{R}$ are the content and style weights, respectively. Minimization of this loss function leads yields the stylized image, i.e.,

$$Z^* = \arg \min_{Z \in \mathcal{Z}} \mathcal{L}(Z, Z_c, Z_s). \quad (2)$$

Evaluation of the content and style losses require a method for feature extraction. The most common approach to extracting these features is to use a CNN. The feature representation of the image Z in the ℓ th (convolutional) layer of the CNN is given by a rectangular matrix $F^\ell(Z) \in \mathbb{R}^{K_\ell \times N_\ell}$, whose elements are denoted by

$$[F^\ell(Z)]_{i,j} = f_{i,j}^\ell(Z), \quad (3)$$

where i denotes the index of the channel and j denotes the index of the feature vector. Given an input image Z and a content image Z_c , the content loss (with respect to the ℓ th layer of the CNN) is simply the distance between these two images in feature space, i.e.,

$$\mathcal{L}_c(Z, Z_c) = \frac{1}{2} \|F^\ell(Z) - F^\ell(Z_c)\|_2^2 \quad (4)$$

$$= \frac{1}{2} \sum_{i=1}^{K_\ell} \sum_{j=1}^{N_\ell} \left(f_{i,j}^\ell(Z) - f_{i,j}^\ell(Z_c) \right)^2. \quad (5)$$

To define the style loss, there needs to be a notion of “style” that can be mathematically represented. In the seminal paper on NST [12], the authors proposed to extract the style of an image by using the Gram matrix of the CNN features. The elements of the Gram matrix (of the ℓ th layer), denoted by $G_{i,j}^\ell(Z)$, are computed by taking an inner product across the feature maps, i.e.,

$$[G^\ell(Z)]_{i,j} = \sum_{k=1}^{N_\ell} f_{i,k}^\ell(Z) f_{j,k}^\ell(Z), \quad \begin{array}{l} i = 1, \dots, K, \\ j = 1, \dots, K. \end{array}$$

In matrix form, this operation is equivalent to taking the outer product of the feature map matrix, i.e., $G^\ell(Z) = F^\ell(Z)F^\ell(Z)^\top$. Using the Gram matrix representation, discrepancy in style can be computed by evaluating a distance of the Gram matrices of the input image Z and the style image Z_s :

$$\mathcal{E}_\ell(Z, Z_s) = \frac{1}{4K_\ell^2 N_\ell^2} \|G^\ell(Z) - G^\ell(Z_s)\|_2^2 \quad (6)$$

The total style loss is a weighted average of the style losses across each of the convolutional layers of the CNN, i.e.,

$$\mathcal{L}_s(Z, Z_s) = \sum_{\ell=1}^L w_\ell \mathcal{E}_\ell(Z, Z_s). \quad (7)$$

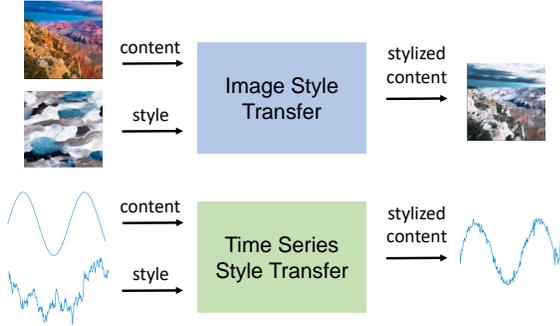


Figure 1: Style transfer for images (top) versus style transfer for time series (bottom).

3 PROBLEM FORMULATION

The goal of this work is to develop a framework for synthetic data generation and enhancement of time series by using style transfer. Under this framework, a time series $y = [y_1, \dots, y_T]^\top \in \mathbb{R}^T$ is optimized such that it captures the overall trend of a content time series $y_c = [y_{c,1}, \dots, y_{c,T}]^\top \in \mathbb{R}^T$ and the realistic distributional properties of a style time series $y_s = [y_{s,1}, \dots, y_{s,T}]^\top \in \mathbb{R}^T$ (see Fig. 1 for a comparison to image-based style transfer). One approach to time series stylization is to use the classic NST algorithm outlined in Section 2 with a CNN adapted to time series (comprised of one-dimensional convolutional layers) as a means for feature extraction. Such an approach would require that a custom model is trained to solve some task (e.g., time series generation, regression, and/or classification). This approach is utilized in [19], where a denoised autoencoder was trained as a means for NST stylization of financial time series. Another approach uses time series to image conversion techniques and then applies existing image-based NST algorithms. Here, a mapping $g: \mathbb{R}^T \rightarrow \mathcal{Z}$ can be used to convert a time series into an image. Converted content and style images can then be used to minimize the NST objective function in (1) to obtain a solution Z^* . Finally, the inverse mapping g^{-1} is used to transform the stylized content in the image domain into the time domain to obtain the stylized time series, i.e., $y^* = g^{-1}(Z^*)$. The benefit of this approach is that pre-trained models can be leveraged for feature extraction (e.g., VGG-19). A variation of this general technique has been successfully applied to stylize synthetic seismic data for realism enhancement [20]. An approach similar in spirit that proposes to transform a continuous numeric time-series classification problem to a vision problem has been shown useful for recovering financial signals in [5].

Many complications can arise from using the aforementioned approaches. In regards to the time series to image conversion to stylize time series, the mapping g must preserve the correlation structure of the original time series. Naive conversion techniques that construct images by simply arranging the time series spatially can introduce spurious autocorrelations when converting the time series back into the image domain (e.g., see Fig. 2). Another issue is that even if the mapping g is able to preserve the correlation structure of the time series, it must be a bijection (i.e., one-to-one). Conversion techniques such as Markov transition fields and Gramian angular

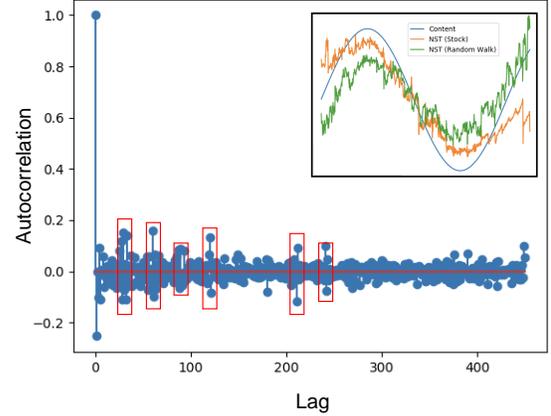


Figure 2: Image-based NST with naive image to time series conversion to stylize a sinusoid with a random walk and a stock price time series. While the resulting stylized time series appear to be reasonable, the autocorrelation function shows spurious spikes at intermittent lags (red boxes).

difference fields are non-invertible mappings and thus are automatically excluded from such an approach [11, 23]. Moreover, time series to image conversion can increase the dimensionality of the data, and hence, lead to increased computational cost of running the style transfer algorithm. For example, while Gramian angular sum fields are invertible mappings that preserve correlation structure of the time series [24], the range of the mapping is $\mathbb{R}^{T \times T}$. This would prove problematic for time series with a long time horizon T .

Finally, in regards to both of the approaches, there is no evidence to suggest that using the Gram matrix in the image domain extracts a style relevant to realism. To see this, let us denote the column vectors of the feature matrix $F^\ell(y)$ (of the ℓ th layer of a CNN adapted for time series) by $f_1^\ell(y), \dots, f_{N_\ell}^\ell(y)$. If one considers that these column vectors are realizations of some random vector $f^\ell(y)$, then the Gram matrix $G^\ell(y)$ is proportional to an empirical estimator of the second moment of that random vector, i.e.,

$$G^\ell(y) \propto \frac{1}{N_\ell} \sum_{n=1}^{N_\ell} f_n^\ell(y) f_n^{\ell\top}(y) \approx \mathbb{E}[f^\ell(y) f^{\ell\top}(y)]. \quad (8)$$

Since the second moment approximation in (8) does not capture any other distributional properties, it may not be suitable for capturing the domain-informed stylistic properties of time series.

4 STYLE TRANSFER FOR TIME SERIES

In this paper, we develop a novel style transfer technique, called StyleTime, for synthetic data generation and enhancement of time series. We introduce the concept of *stylized features* as means for style representation, which are features derived from distributional properties of time series that can effectively be used to assess realism. To that end, we formulate StyleTime as an optimization problem, where the content loss is a measure of reconstruction error with respect to underlying trend (content) and the style loss is measure of discrepancy with respect to the stylized features (style).

Unlike typical generative models, which are capable of generating time series belonging to the domain of training data from which they were trained on, StyleTime is capable of producing realistic time series data with unique trends. As an example, TimeGAN models trained on a dataset consisting of Brownian motion trajectories is only capable of generating Brownian motion trajectories. In contrast, StyleTime can introduce novel trends outside the domain of the training data. For instance, one would be capable of stylizing a step function with a Brownian motion trajectory. Approaches like this could find applications in rare-event modeling, where one would have the capability of enhancing a time series model representing a hypothetical rare-event (e.g., a market shock due to economic events [16]) with the stylized properties of historical data.

4.1 Objective Function

StyleTime follows suit to classic style transfer techniques, where features related to content and style are both extracted from two different time series and then an optimization problem is solved to obtain the stylized content. We define the objective function of StyleTime as follows:

$$\mathcal{L}(y, y_c, y_s) = \alpha \mathcal{L}_c(y, y_c) + \beta \mathcal{L}_s(y, y_s) + \gamma \mathcal{L}_{TV}(y), \quad (9)$$

where $\mathcal{L}_c(y, y_c)$ denotes the content loss, $\mathcal{L}_s(y, y_s)$ denotes the style loss, $\mathcal{L}_{TV}(y)$ denotes the total variation loss, and the coefficients $\alpha, \beta, \gamma \in \mathbb{R}^+$ are hyperparameters that determine the relative weight of the content, style, and total variation losses, respectively. We describe each of these losses in the following:

4.1.1 Content Loss. The content loss $\mathcal{L}_c(y, y_c)$ measures the discrepancy between the trend of the input time series y and the trend of the content time series y_c . This can be quantified using the squared L_2 norm:

$$\mathcal{L}_c(y, y_c) = \|y - h(y_c)\|_2^2, \quad (10)$$

where $h(\cdot)$ is a function used to extract the trend of the content time series (e.g., using a moving average filter). In our experiments, we employ an additive seasonal-trend decomposition, where a moving average filter is used in order to extract the trend of the content time series so that the loss in (10) can be evaluated.

4.1.2 Style Loss. Let $S^\ell(y) \in \mathbb{R}^{d_\ell}$ denote the ℓ th stylized feature of the input y . If we consider L stylized features, we define the style loss as

$$\mathcal{L}_s(y, y_s) = \sum_{\ell=1}^L w_\ell \|S^\ell(y) - S^\ell(y_s)\|_2^2, \quad (11)$$

where $w_\ell = 1/d_\ell$ for all $\ell = 1, \dots, L$. Here, each stylized feature $S^\ell(\cdot)$ represents a distributional property of the time series data that is related to realism. In this work, we mainly considered three different stylized features. Let $r = [r_1, \dots, r_{T-1}]^\top \in \mathbb{R}^{T-1}$ denote the log returns (i.e., log differences) of the time series y , where each element r_t is defined as follows:

$$r_t = \log(y_{t+1}) - \log(y_t), \quad t = 1, \dots, T-1. \quad (12)$$

The stylized features are derived from the following quantities:

- (1) **Autocorrelation of Returns:** For a real-valued discrete-time stochastic process $\{X_t : t = 1, 2, \dots\}$, the autocorrelation function at lag τ is given by

$$R_X(\tau) = \mathbb{E}[X_{t+\tau}X_t]. \quad (13)$$

The autocorrelation function can be used to measure the correlation between a time-series and itself as a function of the lag τ . In the context of finance, the autocorrelation function of price returns are an important indicator of realism. Therefore, we opt to use the sample autocorrelation function of the log returns as a stylized feature, which can be computed as follows:

$$\hat{\rho}_\tau(r) = \frac{\sum_{t=\tau+1}^{T-1} (r_t - \bar{r})(r_{t-\tau} - \bar{r})}{\sum_{t=1}^{T-1} (r_t - \bar{r})^2}, \quad (14)$$

where $\bar{r} = \frac{1}{T-1} \sum_{t=1}^{T-1} r_t$ is the average log return.

- (2) **Volatility:** In the context of financial time series, volatility is a measure of the standard deviation of the log returns of the price time series. Generally speaking, this quantity can be used to quantify the variation of any given time series over time. Incorporating volatility as a stylized feature facilitates that the underlying noise distribution of the time series being optimized, i.e. y , will have similar variation to that of the style time series y_s . In this work, we use the following estimator of volatility to extract the stylized feature:

$$\hat{\sigma}(r) = \sqrt{\frac{1}{T-2} \sum_{t=1}^{T-1} (r_t - \bar{r})^2} \quad (15)$$

- (3) **Power Spectral Density:** In signal processing, the power spectral density (PSD) of a signal is used to measure a signal's strength as a function of frequency. For a discrete-time process, the PSD is simply the discrete Fourier transform (DFT) of the autocorrelation function of the process. In practice, for a time-series y , we can obtain an approximation to the PSD, denoted by $\hat{S}_f(y)$, by taking the DFT of the sample autocorrelation:

$$\hat{S}_f(y) = \sum_{\tau} \hat{\rho}_\tau(y) e^{-i(2\pi f)\tau}, \quad (16)$$

where f denotes the frequency component. We use the fast Fourier transform (FFT) algorithm to obtain a fast computation of the DFT. Ultimately, we use the average value (across frequencies) of the sample PSD in (16) as the stylized feature.

4.1.3 Total Variation Loss. In the context of images and videos, the total variation loss was introduced to regularize the optimized image in the NST algorithm for realism enhancement [14]. By adding this loss to the original NST objective in (1), the resulting stylized content has smoother and more realistic appearance. For time series data, the total variation loss $\mathcal{L}_{TV}(y)$ can be used as a means to penalized highly non-stationary behavior (e.g., sudden discontinuities) and is defined as follows:

$$\mathcal{L}_{TV}(y) = \sum_{t=1}^{T-1} (y_{t+1} - y_t)^2. \quad (17)$$

Algorithm 1 StyleTime(y_c, y_s)

Input: Content time series y_c and style time series y_s .
Default Parameters: $\alpha = 1, \beta = 10, \gamma = 0.0001, I = 250$.

- 1: **Initialization:** Set $y_1 := y_c$.
- 2: **for** $i=1, \dots, (I-1)$ **do**
- 3: Evaluate content loss $\mathcal{L}_c(y_i, y_c)$ according to eq. (10).
- 4: Evaluate style loss $\mathcal{L}_s(y_i, y_s)$ according to eq. (11).
- 5: Evaluate total variation loss $\mathcal{L}_v(y_i)$ according to eq. (17).
- 6: Compute total loss as:

$$\mathcal{L}(y_i, y_c, y_s) = \alpha \mathcal{L}_c(y_i, y_c) + \beta \mathcal{L}_s(y_i, y_s) + \gamma \mathcal{L}_{TV}(y_i)$$
- 7: Gradient update of stylized time series:

$$y_{i+1} := y_i - \eta_i \nabla_{y_i} \mathcal{L}(y_i, y_c, y_s)$$
- 8: Determine the next learning rate η_{i+1} .
- 9: **end for**

Return: y_I

Empirically, we found that incorporating this regularizer is useful for obtaining more realistic time series, especially in cases where the content time series differs greatly from the style time series.

4.2 Implementation

In this subsection, we describe the practical implementation of StyleTime for generating synthetic time series datasets. First, we discuss the implementation of StyleTime for generating a single time series and follow that up with discussion on how one can generate a dataset with multiple time series.

4.2.1 StyleTime Block. A summary of the implementation of StyleTime for generating a single time series is shown in Algorithm 1. In this implementation, we begin by initializing the value of the optimized variable y (i.e., the stylized content) as the content time series. Then, at each iteration of the algorithm, we iteratively update the variable y by using a gradient descent update with respect to the loss function $\mathcal{L}(y, y_c, y_s)$. In our implementation, we used RMSprop optimizer to complete each gradient step for a fixed number of iterations I . Alternatively, one can also run the optimizer until some stopping condition is achieved.

4.2.2 Generating Synthetic Time Series Datasets. We can use Algorithm 1 to generate a new time series \tilde{y} given two other time series y_c and y_s . Given a historical “style” dataset $\mathcal{D}_s = \{y_s^{(n)}\}_{n=1}^{N_s}$ and a reference “content” dataset $\mathcal{D}_c = \{y_c^{(n)}\}_{n=1}^{N_c}$, we can generate a synthetic dataset $\tilde{\mathcal{D}} = \{\tilde{y}^{(n)}\}_{n=1}^N$ by iteratively applying StyleTime to random samples obtained from \mathcal{D}_s and \mathcal{D}_c (see Algorithm 2).

In practice, we choose our style dataset \mathcal{D}_s to be the training dataset, while the content dataset can be chosen in multiple ways. In this work, we consider the following approaches:

- (1) **Training dataset:** The content dataset can be also be chosen to be the training dataset. In this case, the synthetic dataset generated via Algorithm 2 should be statistically similar to that of the original training dataset.
- (2) **Perturbed training dataset:** The content dataset can be chosen to be a “shocked” version of the training dataset. For

Algorithm 2 Generation of Stylized Time Series Datasets

Input: Content dataset \mathcal{D}_c and style dataset \mathcal{D}_s .

- 1: **Initialization:** Set $\tilde{\mathcal{D}} := \emptyset$
- 2: **for** $n=1, \dots, N$ **do**
- 3: Randomly sample content time series $y_c \sim \mathcal{D}_c$.
- 4: Randomly sample style time series $y_s \sim \mathcal{D}_s$.
- 5: Run StyleTime on the sampled time series

$$\tilde{y}^{(n)} \leftarrow \text{StyleTime}(y_c, y_s)$$
- 6: Append $\tilde{y}^{(n)}$ to $\tilde{\mathcal{D}}$.
- 7: **end for**

Return: $\tilde{\mathcal{D}}$

example, one can add a randomly shifted and scaled unit-step function to each example in the training dataset.

- (3) **Generative model:** In principle, one can also use StyleTime to enhance the quality of synthetic data generated from generative model. This can be done by using Algorithm 2 with content dataset obtained by generating synthetic data from another generative model (e.g., Fourier flows).

5 EXPERIMENTAL RESULTS

Here, we show some experiments to demonstrate the validity of StyleTime for time series synthetic data generation. Our evaluation is conducted across three different datasets:

- (1) *Switching AR(1)*: a synthetic switching first-order autoregression dataset generated via the following model:

$$y_t = \begin{cases} \alpha_{1,1} y_{t-1} + \alpha_{1,0} + \epsilon_t, & t = 1, \dots, T_s - 1, \\ \alpha_{2,1} y_{t-1} + \alpha_{2,0} + \epsilon_t, & t = T_s, \dots, T, \end{cases}$$

where $y_0 \sim \mathcal{N}(0, 1)$, $T_s = \lfloor 0.8 \times T \rfloor$, and $\epsilon_t \stackrel{i.i.d.}{\sim} \mathcal{N}(0, 1)$ for all t . The parameters of the model are set as follows: $\alpha_{1,0} = 0.01$, $\alpha_{1,1} = 1.001$, $\alpha_{2,0} = -0.01$, and $\alpha_{2,1} = 0.999$. This synthetic dataset is intended to model a regime switch that is characteristic of financial data.

- (2) *Stock*: univariate time series extracted from the Google stock price historical data.
- (3) *Energy*: univariate time series taken from the UCI energy dataset that contains observations of energy use of appliances. This dataset and the Stock dataset were both used to validate other time series deep generative models, including TimeGAN and Fourier flows [1, 27].

To convert the above univariate time series to a dataset, we apply the sliding window technique using a window size W , which gives us a dataset $\{y^{(n)}\}_{n=1}^{T-W}$ with $y^{(n)} = [y_n, \dots, y_{n+W}]^\top \in \mathbb{R}^W$ for $n = 1, \dots, T - W$. For all datasets, we use a sliding window size of $W = 30$ and a time horizon of $T = 3030$, which leads to each dataset containing $N = 3000$ samples. Each dataset is then split into a training set and a test set, where the training set contains the first $N_{train} = 2400$ samples and the test set contains the last $N_{test} = 600$ samples.

We evaluate the performance of all synthetic data generation techniques from two different angles:

	Switching AR(1)		Stock		Energy	
	<i>F</i> -score	MAE	<i>F</i> -score	MAE	<i>F</i> -score	MAE
Fourier flows	0.9464 ± 0.0023	0.0073 ± 0.0007	0.9813 ± 0.0010	0.0079 ± 0.0022	0.9170 ± 0.0075	0.0312 ± 0.0004
StyleTime (In-Sample)	0.9970 ± 0.0003	0.0071 ± 0.0009	0.9967 ± 0.0001	0.0059 ± 0.0009	0.9969 ± 0.0003	0.0286 ± 0.0004
StyleTime (Perturbed)	0.9962 ± 0.0003	0.0069 ± 0.0006	0.9971 ± 0.0002	0.0057 ± 0.0010	0.9974 ± 0.0004	0.0287 ± 0.0001
StyleTime + Fourier flows	0.9948 ± 0.0009	0.0077 ± 0.0015	0.9827 ± 0.0099	0.0089 ± 0.0022	0.9755 ± 0.0020	0.0353 ± 0.0013
Percent Improvement	5.35%	5.55%	1.61%	27.85%	8.77%	8.33%

Table 1: Assessment of synthetic data fidelity for considered sine, stock, and energy datasets. Metrics are reported as (mean ± standard error) averaged over 5 different random seeds. Percentage improvements are computed to compare the best performing StyleTime implementation and the Fourier flows baseline for each metric separately.

- (1) *Fidelity*: We evaluate the fidelity of style transfer enhanced synthetic data in comparison to one of the state-of-the-art deep generative time-series models (i.e., Fourier flows). Synthetic data fidelity is quantified using precision/recall metrics from [18] and by using the TSTR framework [7] to assess the predictive utility of the synthetic time series.
- (2) *Effectiveness*: We evaluate the predictive utility of the enhanced synthetic data in the context of a data augmentation. Here, the synthetic data generated by each technique is used to augment the original training dataset. The goal is to assess the usefulness of the synthetic data to predict out-of-sample sequences (i.e., to improve predictive performance on the test set). In addition to comparing to synthetic data generated by Fourier flows, we also compare to time series-based data augmentation techniques [10, 25].

For StyleTime, we generate synthetic data using Algorithm 2 with the default parameter values in Algorithm 1. In each of the constructed synthetic datasets, the training dataset was used as the style dataset. We consider three different content datasets: (1) training data, (2) perturbed training data, and (3) Fourier flows synthetic data.

5.1 Assessment of Synthetic Data Fidelity

We assess the fidelity of the synthetic data using two summary metrics: the *F*-score and the mean absolute error (MAE) on a TSTR example. First, we use the framework established in [18] to compute a α -precision, β -recall score, as is done in [1]. The α -precision metric measures how much the support of the real data covers the support of the synthetic data (i.e., realism), while the β -recall metric measures how much the support of the synthetic data covers the support of the real data (i.e., diversity). The two scores are then summarized into a single metric (i.e., the *F*-score) by taking the harmonic mean. Secondly, we evaluate the quality of synthetic data by assessing its predictive utility on the training data. In particular, for each method, we use synthetic data to train a vanilla recurrent neural network (RNN) model with two LSTM layers, each with 100 hidden units. We then compute the mean absolute error (MAE) of the model on the original training dataset. If the RNN trained on synthetic data generalizes well to the training data, then it is considered to be a high fidelity dataset. We compare StyleTime with Fourier flows, a deep flow-based model shown to be capable of achieving state-of-the-art performance with respect to both the *F*-score and TSTR MAE. We also compute the metrics for the content time-series extracted for each of the considered datasets.

We summarize the results of this experiment in Table 1. Across of each of the datasets, synthetic data generated via StyleTime outperforms Fourier flows considering both metrics. Unsurprisingly, StyleTime variations that use in-sample and perturbed content give the best results in terms of both *F*-score and MAE, while the variation that uses Fourier flows content actually degrades performance with respect to the Fourier flows baseline. This could be an artifact in the quality of the Fourier flows synthetic data used as the content dataset, rather than StyleTime technique itself. This is evidenced by the TSNE plots in Fig. 4, where we can see that the Fourier flow synthetic data does not adequately learn the distribution of the energy data, possibly due to the datasets non-stationary nature. A similar result is also implied for the Stock dataset when looking at the TSNE plots in Fig. 3, where we again see that using in-sample content ultimately leads to higher fidelity synthetic data than other techniques.

5.2 Assessment of Predictive Utility

In the previous experiment, we tested the fidelity of the synthetic data using a TSTR framework. Unfortunately, TSTR does not provide a means to assessing how well the generation methods extrapolate the underlying distribution of the real dataset. Moreover, the MAE in the previous subsection was computed in absence of a baseline (i.e., train on real, test on real). Here, we test the predictive utility of the synthetic data in the context of data augmentation. For each model, we augment the training data for a variety of different levels (i.e., augmentation size) and assess the MAE of the previous vanilla RNN model for predicting the test dataset. For each method, we report the MAE at the best augmentation level. To test the novelty of the synthetic data, we also apply the authenticity metric proposed in [2] to each proposed data augmentation method.¹ The authenticity metric (denoted by $\hat{A} \in [0, 1]$) measures the degree to which the synthetic dataset is novel with respect to the training data. A high authenticity score ($\hat{A} \approx 1$) implies that the learned generative model that produced the synthetic data produces authentic (and unseen) time series that do not appear in the training data. A low authenticity score ($\hat{A} \approx 0$) implies that the learned generative model is just producing a noisy copy of data appearing in the training dataset. Intuitively, to achieve better generalization and improve predictive utility, successful time series augmentation

¹Our implementation of the authenticity metric utilizes the euclidean distance to measure the discrepancy between real and synthetic samples, which is different than what was proposed in [2].

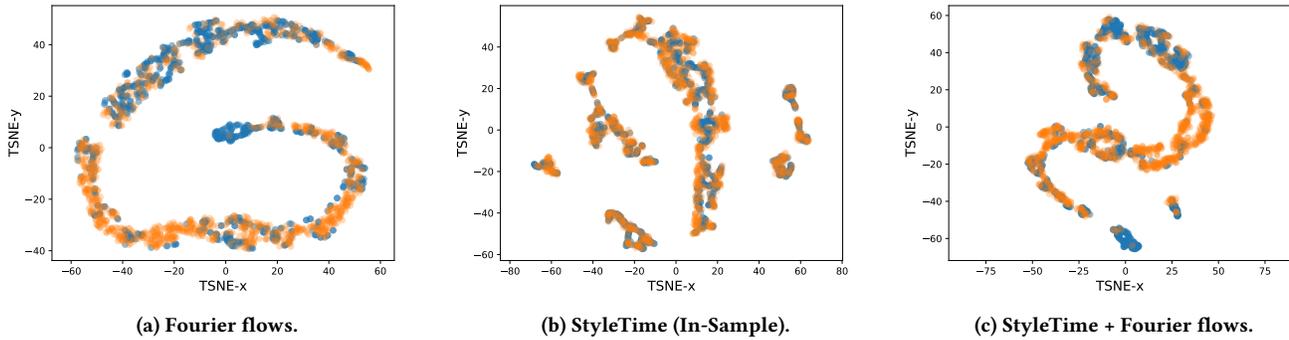


Figure 3: TSNE plots comparing a low dimensional embedding of the Stock training data (blue) with that of the synthetic data generation techniques (orange).

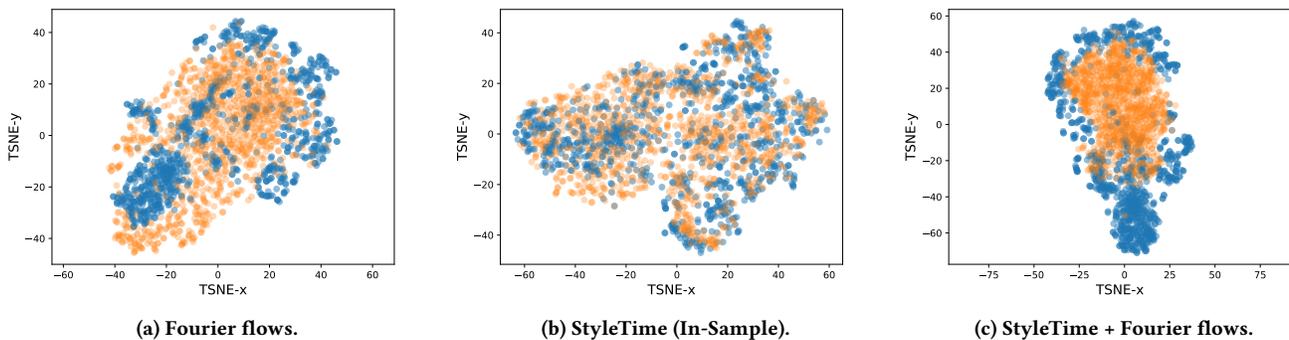


Figure 4: TSNE plots comparing a low dimensional embedding of the Energy training data (blue) with that of the synthetic data generation techniques (orange). Mismatch between training data and Fourier flows synthetic data leads to low quality stylized synthetic data (StyleTime + Fourier flows). Using training data directly as content, however, leads to high fidelity samples.

is expected to have out-of-distribution properties and, hence, display a high authenticity score. Ideally, synthetic data should have a high authenticity score, as producing copies of the data could give rise to privacy concerns. Moreover, a low authenticity score can also be a good technique for assessing whether or not the generation technique has overfit to the training data. For baselines, we compare to Fourier flows and data augmentation techniques designed for time-series (e.g., jittering, flipping, and time warping [10]). We summarize the results of this experiment in Table 2. Results indicate that StyleTime is able to achieve competitive (but not better) performance with respect to the baselines across both the augmentation MAE metric (AugMAE) and the authenticity metric. Importantly, StyleTime is able to achieve decent augmentation performance, while maintaining a high authenticity score with respect to the other baselines. For example, while the time warping augmentation method gives the best AugMAE, its authenticity score is significantly lower than that of all variations of StyleTime.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a new formulation of style transfer for realistic time series generation and enhancement. We explained the

reason why style transfer for time series deserves attention as an independent problem and justified that existing NST techniques for vision do not directly transfer to time series space. We proposed a novel time series specific stylization algorithm, called StyleTime, that uses explicit feature extraction techniques to combine the content (i.e., trend) of one time series with the style (i.e., distributional properties) of another. We show that StyleTime is able to generate synthetic data that achieves a high degree of realism with regard to a variety of metrics. Furthermore, we demonstrated that synthetic time series stylized by StyleTime can augment training datasets to improve the predictive performance of recurrent neural network forecasting models. In the future, we are looking to expand our work to designing methods for stylization of multi-dimensional time series when both time- and cross-series correlations need to be accounted for. We expect such methods to be applicable to realistic enhancement of synthetic limit orderbook datasets, which can in turn be used for data augmentation of market price and volume prediction algorithms on the microstructure level.

	Switching AR(1)		Stock		Energy	
	AugMAE	\hat{A}	AugMAE	\hat{A}	AugMAE	\hat{A}
Fourier flows	0.0089 ± 0.0008	0.9954 ± 0.0013	0.0057 ± 0.0011	0.9760 ± 0.0025	0.0265 ± 0.0004	0.9942 ± 0.0015
Flip	0.0086 ± 0.0007	0.9958 ± 0.0000	0.0062 ± 0.0005	0.9475 ± 0.0000	0.0297 ± 0.0007	1.0000 ± 0.0000
Jitter	0.0086 ± 0.0016	0.9991 ± 0.0004	0.0057 ± 0.0010	0.8665 ± 0.0029	0.0267 ± 0.0007	0.2979 ± 0.0044
Time Warp	0.0081 ± 0.0008	0.6349 ± 0.0127	0.0054 ± 0.0006	0.5981 ± 0.0056	0.0267 ± 0.0006	0.5871 ± 0.0167
StyleTime (In-Sample)	0.0084 ± 0.0010	0.9986 ± 0.0010	0.0055 ± 0.0004	0.9733 ± 0.0028	0.0264 ± 0.0003	0.8761 ± 0.0085
StyleTime (Perturbed)	0.0082 ± 0.0008	0.9997 ± 0.0002	0.0054 ± 0.0004	0.9943 ± 0.0017	0.0263 ± 0.0005	0.8787 ± 0.0077
StyleTime + Fourier flows	0.0091 ± 0.0008	0.9994 ± 0.0004	0.0058 ± 0.0013	0.9879 ± 0.0013	0.0264 ± 0.0004	0.9441 ± 0.0035
Percent Improvement	-1.23%	0.06%	0.00%	1.87%	1.5%	-5.59%

Table 2: Assessment of synthetic data predictive utility and authenticity for considered sine, stock, and energy datasets. Metrics are reported as (mean, standard error) averaged over 5 different random seeds. Percentage improvements are computed to compare the best performing StyleTime implementation and the best performing baseline for each metric separately.

ACKNOWLEDGMENTS

This paper was prepared for informational purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co and its affiliates (“J.P. Morgan”), and is not a product of the Research Department of J.P. Morgan. J.P. Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

REFERENCES

- [1] Ahmed Alaa, Alex James Chan, and Mihaela van der Schaar. 2020. Generative time-series modeling with fourier flows. In *International Conference on Learning Representations*.
- [2] Ahmed M Alaa, Boris van Breugel, Evgeny Saveliev, and Mihaela van der Schaar. 2021. How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models. *arXiv preprint arXiv:2102.08921* (2021).
- [3] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. 2016. Soundnet: Learning sound representations from unlabeled video. *Advances in neural information processing systems* 29 (2016).
- [4] David Byrd, Maria Hybinette, and Tucker Hybinette Balch. 2020. ABIDES: Towards High-Fidelity Multi-Agent Market Simulation. In *Proceedings of the 2020 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation* (Miami, FL, Spain) (SIGSIM-PADS '20). Association for Computing Machinery, New York, NY, USA, 11–22. <https://doi.org/10.1145/3384441.3395986>
- [5] Naftali Cohen, Tucker Balch, and Manuela Veloso. 2019. Trading via Image Classification. <https://doi.org/10.48550/ARXIV.1907.10046>
- [6] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. 2021. TimeVAE: A Variational Auto-Encoder for Multivariate Time Series Generation. *arXiv preprint arXiv:2111.08095* (2021).
- [7] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633* (2017).
- [8] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. 2017. Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. [arXiv:1706.02633 \[stat.ML\]](https://arxiv.org/abs/1706.02633)
- [9] Elizabeth Fons, Paula Dawson, Xiao jun Zeng, John Keane, and Alexandros Iosifidis. 2021. Adaptive Weighting Scheme for Automatic Time-Series Data Augmentation. [arXiv:2102.08310 \[cs.LG\]](https://arxiv.org/abs/2102.08310)
- [10] Elizabeth Fons, Paula Dawson, Xiao-jun Zeng, John Keane, and Alexandros Iosifidis. 2020. Evaluating data augmentation for financial time series classification. *arXiv preprint arXiv:2010.15111* (2020).
- [11] John Cristian Borges Gamboa. 2017. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887* (2017).
- [12] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2015. A Neural Algorithm of Artistic Style. [arXiv:1508.06576 \[cs.CV\]](https://arxiv.org/abs/1508.06576)
- [13] Eric Grinstead, Ngoc QK Duong, Alexey Ozerov, and Patrick Pérez. 2018. Audio style transfer. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 586–590.
- [14] Haozhi Huang, Hao Wang, Wenhan Luo, Lin Ma, Wenhao Jiang, Xiaolong Zhu, Zhifeng Li, and Wei Liu. 2017. Real-time neural style transfer for videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 783–791.
- [15] Brian Kenji Iwana and Seiichi Uchida. 2021. An empirical survey of data augmentation for time series classification with neural networks. *PLOS ONE* 16, 7 (Jul 2021), e0254841. <https://doi.org/10.1371/journal.pone.0254841>
- [16] André Varella Mollick and Tibebe Abebe Assefa. 2013. US stock returns and oil prices: The tale from daily data and the 2008–2009 financial crisis. *Energy Economics* 36 (2013), 1–18.
- [17] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016).
- [18] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. 2018. Assessing generative models via precision and recall. *Advances in neural information processing systems* 31 (2018).
- [19] Brandon Da Silva and Sylvie Shang Shi. 2019. Style Transfer with Time Series: Generating Synthetic Financial Data. [arXiv:1906.03232 \[q-fin.ST\]](https://arxiv.org/abs/1906.03232)
- [20] Naomi Takemoto, Lucas Araujo, Tiago Coimbra, Martin Tygel, Sandra Avila, and Edson Borin. 2019. Enriching synthetic data with real noise using Neural Style Transfer. In *16th International Congress of the Brazilian Geophysical Society*.
- [21] Ivan Ustyuzhaninov, Wieland Brendel, Leon A Gatys, and Matthias Bethge. 2016. Texture synthesis using shallow convolutional networks with random filters. *arXiv preprint arXiv:1606.00021* (2016).
- [22] Svitlana Vyetenko, David Byrd, Nick Petosa, Mahmoud Mahfouz, Danial Derovovic, Manuela Veloso, and Tucker Hybinette Balch. 2020. Get Real: Realism Metrics for Robust Limit Order Book Market Simulations. In *International Conference on Artificial Intelligence in Finance*.
- [23] Zhiguang Wang and Tim Oates. 2015. Encoding time series as images for visual inspection and classification using tiled convolutional neural networks. In *Workshops at the twenty-ninth AAAI conference on artificial intelligence*.
- [24] Zhiguang Wang and Tim Oates. 2015. Imaging time-series to improve classification and imputation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- [25] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. 2020. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478* (2020).
- [26] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. 2020. Quant GANs: deep generation of financial time series. *Quantitative Finance* (Apr 2020), 1–22. <https://doi.org/10.1080/14697688.2020.1730426>
- [27] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series Generative Adversarial Networks. In *NeurIPS*.
- [28] Xu Zheng, Tejo Chalasani, Koustav Ghosal, Sebastian Lutz, and Aljosa Smolic. 2019. STaDA: Style Transfer as Data Augmentation. *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications* (2019). <https://doi.org/10.5220/0007353401070114>