Generating Synthetic Spectral Data using Conditional DDPM

Fabian Kubiczek, Stefan Patzke and Jörg Thiem

University of Applied Sciences and Arts Dortmund - Information Technology
Sonnenstr. 96, 44139 Dortmund - Germany

Abstract.
This study investigates the efficiency and effectiveness of Denoising Diffusion Probabilistic Models (DDPM) for generating synthetic spectral data. A modified DDPM was implemented and evaluated in comparison to a previously established model. Both models were trained with and without Classifier-Free Guidance (CFG). In addition, training duration and sample generation are compared. The results demonstrate that the synthetic spectral data exhibits a high degree of alignment with the training data, with only minor deviations. Furthermore, the influence of CFG on the generation process is evident. The findings indicate that the modified DDPM performs better on the given data.

1 Introduction

Hyperspectral data has an impact on various fields such as hyperspectral remote sensing and medicine. The current challenge is to acquire a large amount of spectral data. This data is acquired using expensive imaging systems. To reduce these costs, neural networks can be used to synthesise the data [1, 2, 3]. For this purpose, two DDPMs are implemented and analysed. One is a modified DDPM inspired by [4]. The second is the Biodiffusion-DDPM which is used in [5]. The outcome of both DDPMs is evaluated and compared within this work. This comparison also takes the training duration into account. The difference in the results with the use of CFG is shown. In the next section, the methodology is described in detail, starting with the used dataset and the structure of the used U-Net architecture. Furthermore, the diffusion process is defined and the implementation of the DDPM is explained. Subsequently, the evaluation methods are introduced. In the penultimate step, the results are presented. The paper concludes with an outlook on future research perspectives and suggestions for further improvement.

2 Methods

In this section, we present the utilized dataset, provide a detailed description of the design and implementation of the U-Net architecture, define the diffusion process and outline the evaluation methods employed.
Spatial information is not considered in this study. This allows the generated spectra to be used for applications with a per-pixel basis, where individual data points lack spatial context.

2.1 Dataset

For this work, the Indian Pines dataset is used as it is partially labeled and well known. This data describes a landscape across Indiana, USA [6] and consists of a hyperspectral image (HSI) of dimension 145x145x200, while the first two dimensions contain spatial and the third spectral information, respectively. The HSI thus contains 21025 (145x145) pixels, for each of which a spectrum is available. Those pixels are partially labeled into 16 classes. Detailed explanations of this dataset can be found in [6]. To use the data, the HSI is reshaped into a 2D array. In this 2D array, each row is a 200-dimensional vector representing the spectrum of a single labeled pixel.

2.2 Design and implementation of U-Net architecture

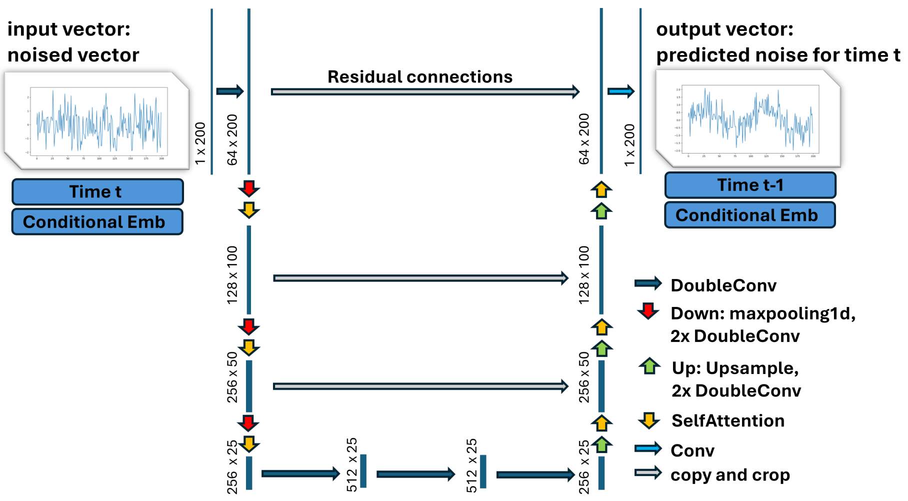


Figure 1: Structure of the U-Net architecture

The architecture of the DDPM U-Net inspired by [4] is shown in Figure 1. It is a symmetrical structure with two primary paths, which are referred to as encoder and decoder. In the encoder, the input vector is reduced to the dimensions 256 x 25 by downscaling after a convolution *DoubleConv*. This is realised with a total of three *Down* and *SelfAttention* blocks. This is followed by a sequence of three *DoubleConv* blocks in the lower part of the U-net. Three *Up* and *SelfAttention* blocks follow in the decoder path. The output is adapted to the size of the input with a final *Conv1d*. A one-dimensional noisy vector, the timestep t and the labels (used as conditional embeddings) serve as inputs to the network. In each layer, the timestep t and the conditional embedding are combined and fed into the network. The network's output is the predicted noise, which is then used to compute the data at the previous timestep *t-1*.

2.3 Definition of the diffusion process

A DDPM is a parameterized Markov chain designed to generate data samples that closely resemble the training data after a specified number of steps. The process of diffusion is reversed and starts by gradually adding noise to the data in small increments until the original signal is entirely obscured. The model then learns to reverse this process step by step, gradually restoring the original data [7].

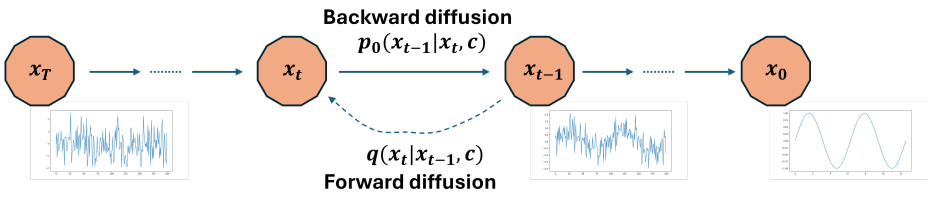


Figure 2: Visualisation of the diffusion process

The diffusion process is shown in Figure 2, where $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{c})$ represents the forward diffusion and $p_0(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{c})$ the backward diffusion, respectively. At time T, noise was added to the vector T times. The vector $\boldsymbol{x}_0$ represents the original data, which in this case refers to a spectrum.

The forward and backward diffusion process is defined according to [7]. An extension is added to the process to enable conditional data generation. By this, besides the data $\boldsymbol{x}_0$, a number of conditions $\boldsymbol{c}$ can be incorporated. The conditions can be additional information or restrictions that influence the generation process. In this case, the labels of the dataset are transferred as $\boldsymbol{c}$ [5]. This approach permits the generation of samples pertaining to a particular class.

$$p_0(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{c}) := \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu_0}(\boldsymbol{x}_t, t, \boldsymbol{c}), \boldsymbol{\Sigma_0}(\boldsymbol{x}_t, t, \boldsymbol{c})) \tag{1}$$

Equation 1 represents the backward diffusion process. Here, the label information is transferred with $\boldsymbol{c}$. The label information is combined with t. The training and sampling algorithms are implemented according to [7]. To calculate the loss, the gradient between the added and predicted noise is calculated. The model is also trained on a partial 10% of the dataset using CFG to generate generalized spectral representations. The learned unconditioned model is then utilised in the sample process, with the unconditioned predicted noise being interpolated with the conditioned model prediction. This enables a trade-off to be made between quality and diversity [8]. The definition of the diffusion process differs from that of the Biodiffusion-DDPM. The process relies on a prior variance schedule and includes a model output. The loss also uses the Kullback-Leibler divergence and optimises the evidence lower bound (ELBO). In addition, the β schedule is non-linear. These aspects are further described in [5]. The modified DDPM uses a fixed variance for the backward diffusion process, a simple loss function and a linear β schedule (see [7]).

3 Results

3.1 Training and sample duration

Table 1: Training (seconds per epoch) & sample duration (seconds per sample)

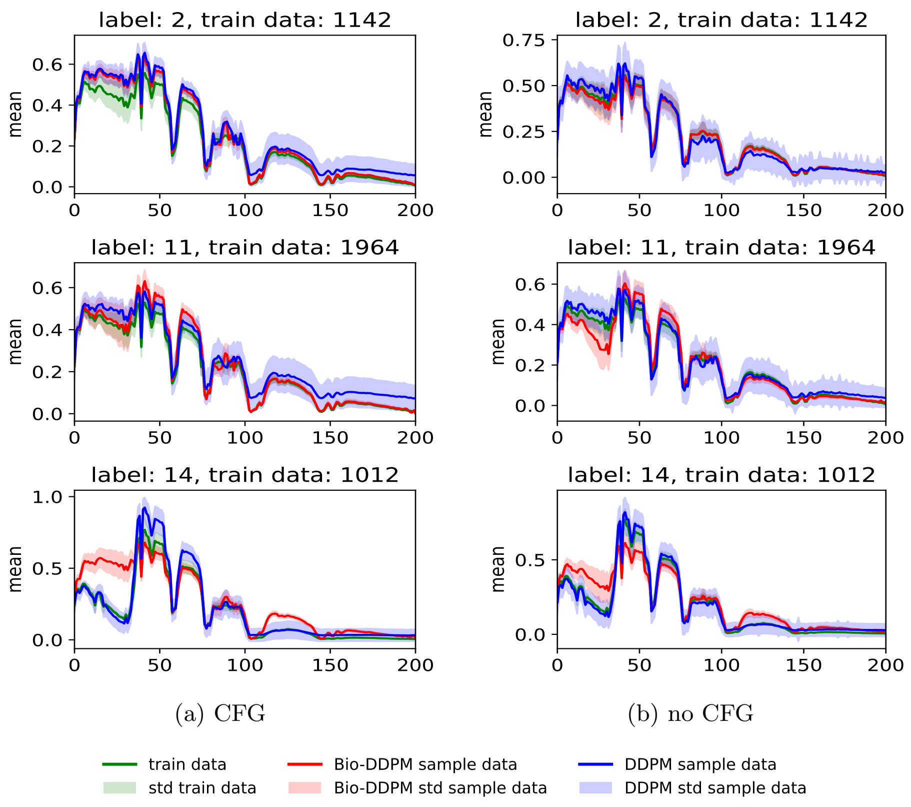
model	training duration (s/ep)	sample duration (s/sample)
DDPM CFG	27.462	16.852
DDPM	13.206	7.252
BIO-DDPM CFG	8.345	42.156
BIO-DDPM	8.095	21.23

The time required for training and sampling is displayed in Table 1. It can be seen that the modified DDPM performs best during sampling. In comparison with the Biodiffusion-DDPM, the time required is two to three times less. Without CFG, the required sampling time is lower by a factor of 2 for both models. The Biodiffusion-DDPM, on the other hand, can be trained more quickly. The training time for the DDPM without CFG is reduced by a factor of 2.

3.2 Comparison of training and sample data

The training data is compared to the sample data. This is done for both, the modified DDPM and the Biodiffusion-DDPM, each with and without CFG. The following figure shows the mean values and standard deviations of 200 samples and the training data. For the sake of clarity, not all labels are shown in the figures. The figures show the values for three most interesting labels (2, 11 and 14).

Figure 3 show the results of both DDPMs, respectively. The detailed analysis of the results for labels 2, 11, 14 is carried out because these labels best represent the variation within the dataset. Slighter deviations can be observed for labels 2 and 11 and larger deviations for label 14. These larger deviations are visible for the Biodiffusion-DDPM (CFG) with a high Mean Squared Error (MSE) of 0.015, in comparison to the modified DDPM (without CFG), which exhibits an MSE of 0.0005. The standard deviation is marginally higher for the modified DDPM. Overall, the mean values deviate more with the use of CFG, particularly for the Biodiffusion-DDPM. The deviations are particularly visible in the first 100 (modified DDPM) or 150 (Biodiffusion-DDPM) spectral bands.



(a) CFG (b) no CFG

Figure 3: Comparison of training data mean and sample data mean for both DDPMs, each with and without CFG. The training data is shown in green, the Biodiffusion-DDPM sample in red and the modified DDPM sample in blue.

3.3 Metrics

Table 2: Comparison of average metrics between training data and sample data generated by all DDPM variations: Mean Squared Error (MSE), Difference in Standard Deviation ($\Delta\sigma$), Sum of Squared Deviation (SSD), Wavelet Coherence (WC) and Correlation Coefficient (R). The best values are marked in green and the worst values in red, respectively.

model	MSE	$\Delta\sigma$	WC	SSD	R_{xy}
DDPM CFG	0.001713	0.013188	0.964	0.342	0.994
DDPM	0.000588	0.010516	0.967	0.118	0.994
BIO-DDPM CFG	0.008056	0.029566	0.816	1.611	0.961
BIO-DDPM	0.001931	0.007251	0.822	0.386	0.969

The metrical results are presented in Table 2. A comparison of the average metric values shows that the DDPM without CFG performs best and the Biodiffusion-DDPM with CFG performs worst.

4 Conclusion

Overall, the modified DDPM delivers better sample results with a shorter sample duration while the Biodiffusion-DDPM performs better in terms of the training duration. To improve the training duration on the modified DDPM, the *SelfAttention* could be replaced by *LinearAttention* or *MultiQueryAttention* [5, 9, 10]. This approach can be applied wherever the augmentation of spectral data is required, e.g. spectral unmixing or HSI classification purposes, particularly benefiting from the advantage that no spatial information is required.

References

[1] Liguo Wang and Chunhui Zhao. *Hyperspectral Image Processing.* Springer Berlin Heidelberg and Imprint and Springer, Berlin, Heidelberg, 2016.

[2] Pablo Ribalta Lorenzo, Lukasz Tulczyjew, Michal Marcinkiewicz, and Jakub Nalepa. Hyperspectral band selection using attention-based convolutional neural networks. *IEEE Access*, 8:42384–42403, 2020.

[3] Matthis Hofmann, Dominik Fromme, Tim Streckert, and Jörg Thiem. Synthetic image generation of aortic valves using conditional ddpm. *ICBRA*, 11:42384–42403, 2025.

[4] Dominic Rampas. dome272/Diffusion-Models-pytorch: Pytorch implementation of Diffusion Models (https://arxiv.org/pdf/2006.11239.pdf), 2024.

[5] Xiaomin Li, Mykhailo Sakevych, Gentry Atkinson, and Vangelis Metsis. Biodiffusion: A versatile diffusion model for biomedical signal synthesis, 2024.

[6] Hyperspectral remote sensing scenes - grupo de inteligencia computacional (gic), 2023.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models, 2020.

[8] Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance, 2022.

[9] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention.

[10] Noam Shazeer. Fast transformer decoding: One write-head is all you need.