ATIAM PROJECT 2023 - 2024

Latent audio diffusion for music generation with expressive control

Nils Demerlé

1 Introduction

Recent advances in neural audio synthesis have made possible the creation of exciting new tools for composition and sound design. The developpement of approches based on Generative Adversarial Networks (GANs) or Variationnal Auto-Encoders led to poweful audio compression models such as RAVE [1], Encodec [1] or Sounstream [2]. Such models are capable of projecting audio waveform to a low dimensional latent space capturing salient features of the data. However, it is not straighforward to use such models to generate music tracks with long-term coherency, as their receptive field remains limited to a short time window. Early work using autoregressive networks as a prior model on the latent codes has yielded impressive results, at the cost of expensive training and long synthesis time. More recently, diffusion models have proven to reach state-of-the-art results in terms generation accuracy and training stability. Such models are now widely used because of their conditionning capabilities, such as text-to-image models operating on latent codes like Stable Diffusion [3]. In this work, you will explore how diffusion models can be used as a prior model to generate latent codes of an audio compression model, and you will leverage the expressivess of such models to implement high-level controls of different nature over the generation process.

2 Diffusion

You should first get familiar with PyTorch and diffusion models. Following the initial diffusion model proposal inspired by thermodynamics [4], iterative improvements led to denoising diffusion probabilistic models (DDPMs) [5] and deterministic diffusion models (DDIMs) [6] which are currently the most commonly used formulations. In this work, we will focus on a recent proposal of reformulation and simplification of deterministic diffusion models : Iterative alpha-(de)Blending [7].

You will implement the Iterative alpha-(de)Blending method and train your model on a simple 2D point cloud generation problem. As the task is relatively simple, you can use a simple MLP network. The time conditionning scalar should be embedded using sinusoidal positional encoding [Blog Post] and concatenated with the feature map between each layer of the MLP network. Your notebook should illustrate the sampling process as in figure 1

This part of the subject must be done *individually* and submitted as a python notebook.

3 Unconditionnal Audio Generation

The goal of this section is to build you own unconditionnal diffusion model for audio. As directly generating audio waveform is a complex task due to the high dimensionality and multi-scale dependencies of audio signals, you will rely on a pretrained version of the audio compression model

Figure 1: Illustration of the sampling process

Encodec [8]. Hence, your diffusion model will learn to map gaussian noise to Encodec latent codes, which will then be decoded using Encodec's decoder to get back to the waveform domain. A pre-trained model of Encodec as well as encoding/decoding codes will be provided.

3.1 Network Architecture

The state of the art for denoising networks is the UNET architecture [9]. You will implement your own simple UNET network without attention or self-attention blocks. As we are dealing with time sequences, your UNET architecture must be fully based on 1D convolutions. For time conditionning, you will implement the Adaptative group normalisation (AdaGN) technique introduced in [8]. It is defined as follows, for a given intermediate activation h :

$$AdaGN(h, t) = y_s GroupNorm(h) + y_b \tag{1}$$

where $[y_s, y_b]$ is a linear projection of the SPE embedding of the time parameter t.

3.2 Dataset

We will provide you a large scale dataset of music with different pre-computed tags (genre, instruments..), sampled at 22.05 kHZ, alongside the relevant dataloaders. If you are interested in building your own custom dataset, please do, but be aware that this is a time consuming task and that you will not be evaluated on this.

3.3 Experiments

You will perform a quantitative evaluation of you model by computing the diffusion loss on a validation set. You can eventually compare different architectures (in term of number of layers and parameters) and assess if your model generalises well (large diffusion models are prone to overfitting). As qualitative analysis, you will experiment with :

- Audio generation with different number of sampling steps (eventually for different architectures)
- Interpolation. As alpha-deblending learns a deterministic mapping between noise (or any distribution) and the data, you can interpolate between two different noise vectors and decode the audio of the resulting interpolation. Is the interpolation smooth? Experiment with different interpolation step size and distance between the initial noise vectors
- **Bonus** : Using a sufficient number of steps, it is possible to invert the deterministic diffusion process (i.e mapping a data sample to it's noise couterpart) What happens if you map a data sample to noise, and then map it back the data space? Do you get perfect reconstruction? Experiment with a different number of steps. You can try this initially with your cloud point model for easier visualisation.

4 Conditionnal Generation

In this second section, you will empower your diffusion model with expressive controls. Early approaches to condition diffusion models relied on the use of a noise-robust classifier trained on the same dataset. More recently, Classifier-Free-Guidance (CFG) [10] achieved impressive results

without having to train a different model, while also providing users a way to control the trade-off between conditionning strengh and model expressivity. You can follow this [Blog Post] for a good introduction on how classfier free guidance works.

You will implement the CFG on your model for controls of various nature :

4.1 Fixed categorial control

High-level semantic contionning that is global for a given sample, such as genre, instrument, mood, key.. You will be provided a dataset with annotations that you can use straightaway. Such controls must be mapped to an embedding space (using for instance a simple lookup table (pytorch ref)), then fed to the UNET during the diffusion process. In this case, a good practice is to simply concatenate the control vector to the time embedding in the AdaGN layer.

4.2 Time varying controls

Controls that can vary along the sample such as audio descriptors (RMS, centroid, noise level) or rythm analysis such as BPM clock signal or onsets. Different strategies to inject the conditionning signal in the UNET will be discussed later on, based on the control signals chosen by the students.

This is the moment to be creative and find control signals that are musically relevant and exciting to play with !

4.3 Semantic Control

Finally, we propose to rely on CLAP [11], a model trained with a contrastive objective on pairs of audio and text. This model provides both a text encoder and an audio encoder that maps each modality to a shared representation space. It has been proven on multiple downstream tasks that this representation captures high-level semantics about the content of audio signals. After conditioning your diffusion model on the CLAP embeddings of your training set, you can then explore different applications :

- Encode a text query with the text encoder, and generate audio samples that matches this query.
- Encode an audio sample and produce multiple variations with the same semantic content by sampling different input noises.
- Interpolate in CLAP representation space

References

[1] Antoine Caillon and Philippe Esling. RAVE: A variational autoencoder for fast and high-quality neural audio synthesis. 11 2021.

[2] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. SoundStream: An End-to-End Neural Audio Codec. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 30:495–507, 7 2021.

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[4] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[6] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[7] Eric Heitz, Laurent Belcour, and Thomas Chambon. Iterative alpha-(de) blending: a minimalist deterministic diffusion model. *arXiv preprint arXiv:2305.03486*, 2023.

[8] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[10] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[11] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.