ATIAM
Ninon Devis
David Genova

# Final ML Project

Conditional audio synthesis
using adversarial networks

2022-10-23
ninon.devis@ircam.fr
david.genova@ircam.fr

**Abstract**

In recent years, *Generative Adversarial Networks* (GANs) have yielded impressive results on various generative tasks. In particular, *conditional GANs* have gained increasing interest in user-guided generation, yet tend to suffer from *mode collapse*, resulting in low sample diversity. In this project, you will study how GANs can be used for class-conditional audio synthesis, and how to increase the diversity of the generated sounds. Specifically, the ultimate goal is to obtain a model capable of generating audio based on the desired genre. To do so, you will need to find a proper classification for "genres" and think about how to accurately condition your model. Hence, this project tackles several challenges: the training of a model on a variety of dataset and the conditioning of this model in order to produce a diversity of audio within a genre.

## 1. Overall presentation

### Global instruction

#### Generalities

| | |
|---|---|
| Deadline | January 2023 |
| Organization | Group from 4 to 5 students |
| Deposit | `Github` |

#### Project folders

| | |
|---|---|
| `code/` with your code following the PEP8 coding style and organized in modules | |
| `report/` in PDF format | |

#### Report

| | |
|---|---|
| Redaction | LaTeX with a **given format style** that you will receive |
| Language | English |
| Maximum number of pages | 8 pages following the scientific papers |

#### Evaluation grid

| | |
|---|---|
| Report - Including content, results and style | **7 pts** |
| Code - Accuracy, evaluation and coding style | **13 pts** |

For more information on the project: ATIAM machine learning project webpage

## 2. Introduction

Among recent generative systems found in the literature, two have had a large success in the machine learning community. First, the *variational auto-encoder* (VAE) is based on a two-stage inference/generation procedure that showed great generalization properties, however can suffer from blurry reconstruction [4]. The second is the *Generative Adversarial Network* (GAN) (see [2]), that showed impressive reconstruction abilities but rather poor latent expressivity.
This project aims to study the use of GANs to create novel audio samples with external conditioning, while fostering diversity among the generation. You will evaluate the success of your approach with the following objectives :

1. *Obtaining the highest quality of reconstruction*. This implies to find an appropriate representation of the input data and design an appropriate model specifically designed for this learning task.

2. *Conditioning the model for controllable synthesis*. This means that your model should be able to generate sounds based on a given genre, which requires to think of appropriate embeddings for this task.

## 3. GAN on MNIST

The first step with generative adversarial networks will be to implement them on simple data in order to understand their inner workings. This will allow you to decide the implementation details that you will need to use for the future.

**Exercise 1: Learn Pytorch and useful libraries.** This exercise requires learning basic programmation skills that you will need for the following work.

1. Create the Github repository that will host your code.

2. Install and learn PyTorch through basic tutorials `http://pytorch.org/tutorials/`

As your models will run on GPUs, make sure all your code is CUDA compatible, you can check the CUDA semantics here : `https://pytorch.org/docs/stable/notes/cuda.html`.

**Exercise 2 : code the GAN** You will now code your own convolutional generative adersarial network, as detailed in [7]. As GAN training can be difficult, you can rely on this tutorial `https://www.tensorflow.org/tutorials/generative/dcgan`, which is coded in TensorFlow.

1. Read the founding paper on generative adversarial networks : [2].

2. Based on the tutorial or another source you will find, develop your GAN in PyTorch and train it on the MNIST dataset.

3. Implement the class-conditioning with your previous architecture as detailed in [6].

4. Keep your project as modular as possible ! You should have one module for your data, one for your models, one for your training procedures.

⋆ *Show your code and results to your supervisors* ⋆

## 4. Conditional adversarial synthesis of audio

Now that you have understood how GANs work on a simple task, it is now time to adapt your model for audio generation. To this end, you will rely on the *Short Term Fourier Transform* (STFT) of the audio data, and train your model on the log-magnitudes of these spectrograms. One of the challenge of this assignment is to formalize the boarders between genres. You will need to think about classes and subclasses of the genres you want to generate.

**Exercise 3 : Develop a GAN for audio synthesis**

1. Regarding the dataset you will first train your model on the "Vengeance" dataset, which classify samples into categories of genres.
2. Extract the log-magnitude spectrograms of the audio using the *torchaudio* package[1]. You will also need an inversion method to convert back into the audio domain, which can be done by using the Griffin-Lim algorithm [3].
3. Implement a GAN for spectrogram synthesis. As this task is significantly harder than for the MNIST dataset, you will follow the work of [1] for the architecture of the model (without the instantaneous frequencies and class conditioning at first).
4. Compute the *real time factor* of your model, which is the ratio between the time required to generate one sample against the duration of this sample. What do you think is the major impediment for real-time synthesis ?
5. You will now introduce the conditioning on your model. To do so, you need to extract from you data a conditioning vector and concatenate it to the z vector before passing to the generator and to the data forwarded to the discriminator. To adapt the conditioning seen in [7] how will you transform your classes and subclasses into vectors ?
6. Implement conditioning of the generation on the class. Generate several sounds for each class. How does the conditioning affects the resulting sounds ?

**Exercise 4 (optional) :Enhance diversity among the generated sounds with contrastive loss** We now propose to mitigate the *mode collapse* issue with conditional GANs by introducing a *contrastive* loss to enforce the audio generation. In this section, you will follow the line of [5] and apply it to your model.

1. Implement the auxiliary contrastive loss with your previous model.
2. For each class, generate a few audio samples. How does the contrastive loss affects the diversity of the generated sounds, as well as the reconstruction quality ?

**Exercise 5 (optional) : Eurorack embedding** You can now try to train your model on a crafted dataset of your choice. When your model produces great sounds, it is time to make a real instrument from it ! We propose to integrate your solution in a dedicated Eurorack module, in order to perform embedded audio synthesis.

[1] https://pytorch.org/audio/stable/index.html

## References

[1] Jesse Engel et al. “GANSynth: Adversarial Neural Audio Synthesis”. In: *International Conference on Learning Representations*. 2018.

[2] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[3] Daniel Griffin and Jae Lim. “Signal estimation from modified short-time Fourier transform”. In: *IEEE Transactions on acoustics, speech, and signal processing* 32.2 (1984), pp. 236–243.

[4] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[5] Qi Mao et al. “Mode seeking generative adversarial networks for diverse image synthesis”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 1429–1437.

[6] Mehdi Mirza and Simon Osindero. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014).

[7] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015).