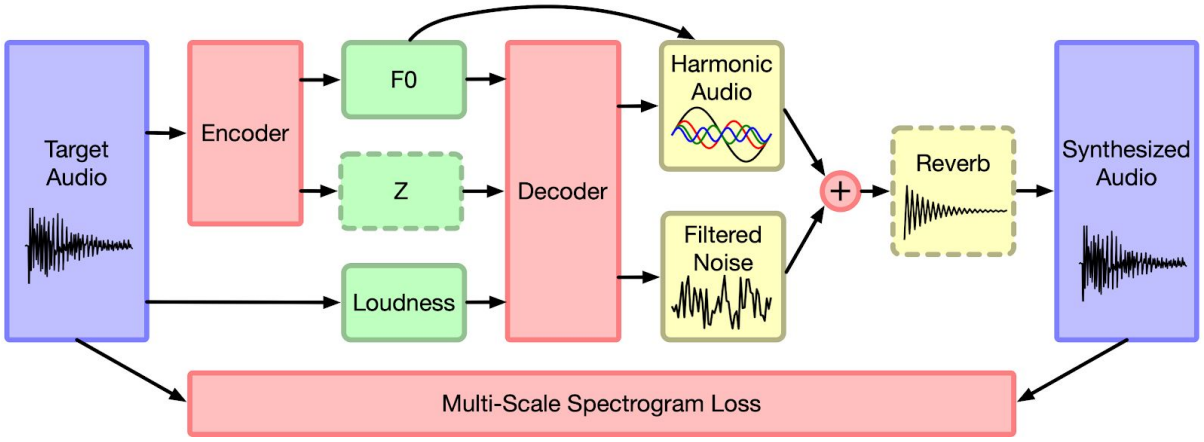
AI x Audio

Differentiable Digital Signal Processing for Artificial Sound Generation

Supervisor: Simon Leglaive (FAST team, CentraleSupélec).
Keywords: Audio signal processing, deep learning, musical sound generation.
Number of hours: 200 HEE if semester S8 or 290 HEE if semesters S7 and S8.[1]
Number of students: Maximum 5 students, who should **all** start and end the project at the same time.

Project description



DDSP architecture - source: https://magenta.tensorflow.org/ddsp

On October 12, 1964, Bob Moog unveiled the first modular voltage-controlled synthesizer, an instrument that forever changed the course of modern music [1]. Today, artificial intelligence (AI) is changing the ways we synthesize and produce audio, especially music [2]. In 2020, researchers from Google working on the Magenta project released the Differentiable Digital Signal Processing (DDSP) library. DDSP combines traditional DSP elements (such as filters, oscillators, reverberation, etc.) with more recent deep learning concepts in order to provide efficient, modular and controllable ways of synthesizing sounds [3]. One of the main advantages of DDSP is that it allows us to train high-quality audio synthesis models with very little amount of data (15 min of audio is enough). DDSP has been used essentially for manipulating a given input audio signal, for instance to convert a singing voice into a violin. However, DDSP has not yet been used as a "true" generative model, which would be able to generate new sounds without the need of any input audio signal. **The goal of this project is to turn DDSP into a "true" generative model**.

[1] This project is too big for being conducted only in S7 (90 HEE). If some students want to work on a similar topic only during S7 because they will not be in Rennes during S8, they can propose a shorter version of the project, to be discussed with the supervisor.

Expected work

The project will be organized in three tasks described below.

T1 - Preliminary work

T1.1 - The DDSP library comes with a conference paper [3], a blog, a demo web page, and several Google Colab notebooks for demos and tutorials. You will have to study this material to understand what DDSP is, what possibilities it offers for sound generation and what are the limitations.

T1.2 - DDSP official implementation is based on Tensorflow. You can find on Github other implementations based on PyTorch. If necessary, you will have to get familiar with one of these deep learning libraries, following tutorials of your choice.

Expected outcome of T1: A presentation about the general principles of DDSP and a Jupyter notebook showing the basics of Tensorflow or PyTorch (depending on the chosen library).

T2 - Benchmarking DDSP

T2.1 - As DDSP requires only a few minutes of audio for training, we let you collect and process sounds of a monophonic and harmonic instrument of your choice.

You will then have to:

T2.2 - Validate data preprocessing (pitch and loudness estimation);
T2.3 - Train different variations of the DDSP synthesizer (harmonic only, harmonic + noise, etc.);
T2.4 - Evaluate the audio reconstruction quality.

Expected outcome of T2: One or several DDSP synthesizers trained on a dataset that you built.

T3 - Making DDSP truly generative

The DDSP synthesizer is conditioned on three inputs (“f0”, “loudness” and “z”). “z” somehow encodes the residual information which is not related to the pitch (f0) and loudness of the synthesized audio. For instance, it may encode the timbre of the sound. Currently, given only pitch and loudness information, DDSP cannot generate audio, as we do not know how to define (or actually generate) the variable “z” without an input signal passed through an encoder network.

To retrieve a complete generative mechanism and sample new sounds, one has to define a probability distribution over “z”. One simple approach for this may consist in fitting a generative model over latent variables “z” provided by the pre-trained encoder network. A

natural choice for the generative model would be a Gaussian mixture model (GMM), as done in [4]. Another possibility would be to rely on the framework of variational autoencoders [5].

You will have to:

T3.1 - Build a dataset of "z" obtained from the trained encoder (from T2.3) and your dataset (from T2.1);
Why fit on the latent variable z
T3.2 - Get familiar with GMMs (if necessary) and fit a GMM on the dataset from T3.1;
T3.3 - Evaluate the generation quality of the DDSP synthesizer using the GMM from T3.2 and predefined pitch and loudness profiles.

Expected outcome of T3: A fully generative model based on DDSP which can generate musical sounds without any input audio signal.

Organization

At the beginning of the project, students will have to define an agenda with deadlines for the expected outcomes of each task. This agenda will include regular meetings with the supervisor.

In addition to the above technical outcomes, students are expected to *continuously report to the supervisor about the progress of the project*. Microsoft Teams will be used to communicate with other students and with the supervisor. After each project session, the students will have to provide a very short report about the progress they made, the difficulties they encountered, what they will do next, etc.

At the end of the project, students will have to provide one report for the whole group and each student will individually present the group's work, highlighting their personal contributions.

References

[1] https://www.moogmusic.com/news/early-years-moog-synthesizer
[2] S. Lattner, "The Sound of AI - How Machine Learning Impacts Music Production", Sony Workshop at IEEE ICASSP 2020, available at https://youtu.be/KkbVVHK52kM
[3] J. Engel, L. Hantrakul, C. Gu, A. Roberts, "DDSP: Differentiable Digital Signal Processing", International Conference on Learning Representations (ICLR), 2020
[4] P. Ghosh, M. S. Sajjadi, A. Vergari, M. Black, B. Schölkopf, "From variational to deterministic autoencoders", International Conference on Learning Representations (ICLR), 2020.
[5] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes", International Conference on Learning Representations (ICLR), 2014.