Automatic Estimation of FM Synthesis Parameters by Convolutional Neural Network

Sergio Rocha da Silva
sergio.silva16@unifesp.br
Universidade Federal de São Paulo
São José dos Campos, São Paulo, Brasil

Abstract

A clear and well-documented LaTeX document is presented as an article formatted for publication by ACM in a conference proceedings or journal publication. Based on the "acmart" document class, this article presents and explains many of the common variations, as well as many of the formatting elements an author may use in the preparation of the documentation of their work.

CCS Concepts

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Keywords

Do, Not, Us, This, Code, Put, the, Correct, Terms, for, Your, Paper

ACM Reference Format:
Sergio Rocha da Silva. 2026. Automatic Estimation of FM Synthesis Parameters by Convolutional Neural Network. In *Proceedings of The 41st ACM/SIGAPP Symposium on Applied Computing (SAC'26).* ACM, New York, NY, USA, **??** pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

2 Theoretical foundations

2.1 Musical Computation

Musical computation is a broad and interdisciplinary field that encompasses all aspects of musical processing through a computer. It is beyond the scope of this article to provide a comprehensive overview of this field; however, for the purposes of this study, it is important to cover the basic elements of sound and the fundamentals of audio digitization.

2.1.1 Sound elements. Audio is a mechanical compression wave that travels through the air. This type of wave can be converted into an electrical signal through microphones, which consist of an arrangement of coils and magnets. As a wave, its main elements are:

- Pitch: The fundamental frequency of a sound, which determines whether it is low or high. The higher the basis frequency, the higher the pitch.
- Duration: The length of time a sound is sustained and can be heard.
- Loudness (or amplitude): The intensity or energy level of a sound. In terms of a wave, it corresponds to the amplitude of the sound wave.

There are other relevant concepts that could be discussed; however, for the sake of simplicity (and because this work is focuses on Machine Learning), only these will be mentioned.

However, it is important to highlight that although sound can be described in terms of the aspects mentioned above, hearing is mainly a cognitive process. Therefore, sound comparison cannot be fully addressed by mathematical tools, leaving a significant space for perceptual comparison and classification (not to mention aspects related to personal preferences).

2.1.2 Sound timbre. Another important aspect of acoustic sounds is timbre, which can be understood as the "color of sound."

From a mathematical point of view, timbre varies according to the harmonics present in a sound sample, or, in other words, according to the set of oscillations that compose the sound.

In the case of acoustic instruments, what happens is that the air vibrations produced by the instrument are not simple pure sinusoids. In fact, when a string vibrates, or when air resonates inside a tube, many parts of the instrument vibrate simultaneously, and the frequency with which each part vibrates differs according to its composition, position, and other factors.

The result is that each instrument has a unique sound, especially in the case of handcrafted wooden instruments such as violins or cellos.

To obtain a clear visualization of this aspect, it is common to convert a sound sample from the time domain to the frequency domain using a Fourier Transform technique.

For example, consider the two images below, which compare the same violin audio sample in both the time and frequency domains:

s the figures show, there is a main frequency that defines the pitch of the sound, along with many other frequencies that allow the ear to identify the instrument and distinguish between different kinds of sounds.

2.1.3 Audio digitalization. A microphone converts a pressure wave into an electrical waveform. However, in computational music, the audio signal must be discretized to be represented as a digital audio file. The primary process for digital audio representation is sampling.

SAC'26, Thessaloniki, Greece

ACM ISBN 979-X-XXXX-XXXX-X/26/03
https://doi.org/XXXXXXX.XXXXXXX

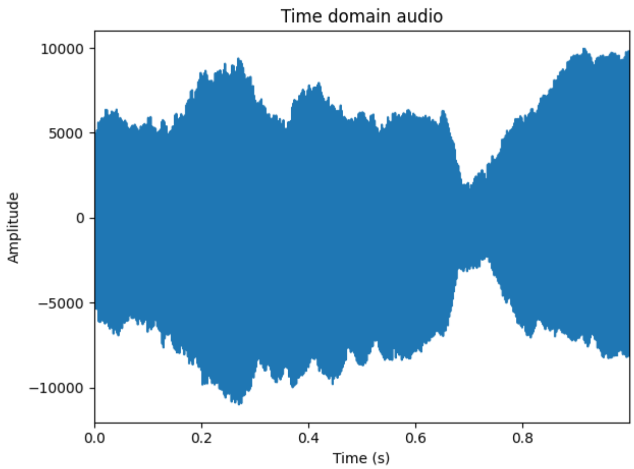

Figure 1: One second time domain violin sample.

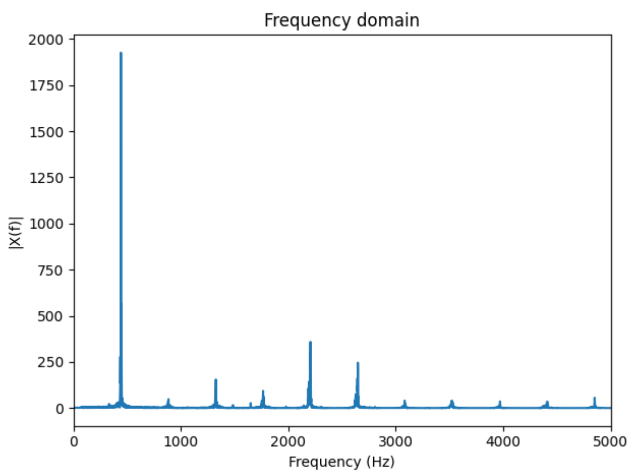

Figure 2: TThe same sample in frequency domain.

Sampling consists of recording a sequence of wave amplitudes at fixed intervals. For example, if a sound is recorded using a sampling rate of 22,100 Hz, it means that the computer will capture 22,100 audio samples per second.

This approach is very efficient because human hearing can detect frequencies between 20 Hz and 20,000 Hz. However, for most musical sounds, the maximum frequency is typically below 10 kHz (in fact, even a violin — one of the highest-pitched instruments — reaches only around 3,500 Hz). And according to the Nyquist-Shannon Theorem, to properly sample a periodic signal, the sampling rate must be at least twice the frequency of the highest component of the signal.

Thus, 22,100 Hz can be a good sampling rate for recording music while saving disk space. However, if the goal is to cover the full range of human hearing, higher rates, such as 44,200 Hz, can also be used.

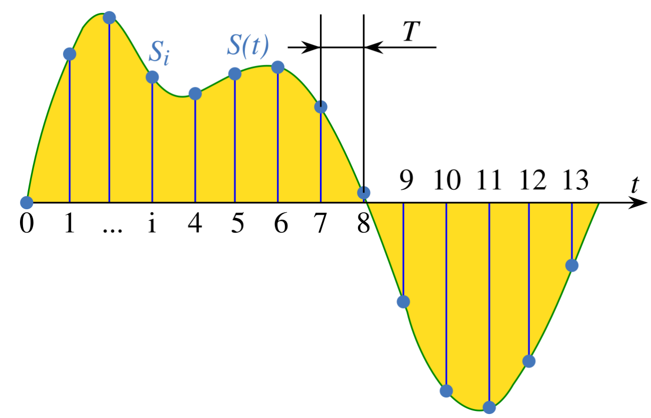

Figure 3: Signal sampling. Source: [1]

Note that with this process, all sound elements highlighted before, can be recorded and reprocedure later.

2.1.4 Sound synthesis. After the development of sound recording techniques, musical computation evolved toward the creation of artificial waveforms. Of course, before digital audio synthesis, there were several techniques that allowed analog synthesis. However, the underlying idea is essentially quite similar.

In summary, audio synthesis techniques can be divided into three basic types:

- Additive synthesis: Perhaps the most straightforward idea from a mathematical point of view. It consists of generating each frequency component of a desired timbre separately and then summing them. In fact, it is more like a weighted sum, since each component is weighted according to its contribution to the timbre. A normalization step is commonly applied after the process to avoid amplitude overload.
- Subtractive synthesis: The opposite of the previous technique, but historically more common due to its effectiveness. In short, it consists of taking a complex waveform, rich in harmonics, and applying various filters to obtain the desired sound. It is more empirical than additive synthesis (which is based on spectral analysis), but simpler to implement and less computationally intensive. Typically, the original waveforms are complex ones such as the sawtooth wave, square wave, or triangular wave.
- FM synthesis: Another empirical approach to sound synthesis, which will be addressed in detail in a later section. In short, it consists of altering a simple waveform by applying a high-frequency modulation, not to the generated waveform itself, but directly to the frequency parameter of the carrier oscillator. This kind of distortion of the base frequency produces mirrored harmonics that enrich the timbre of the sound. It is up to the musician to choose the appropriate parameters to produce the desired timbre.

There are many other techniques that could be mentioned; however, these three represent the most direct forms of synthesis, or the most purely mathematical ones, in the sense that they do not require manipulation of pre-recorded audio.

2.1.5 ADSR Audio envelop. Another important technique for achieving good results is the so-called "Attack, Decay, Sustain, and Release (ADSR) envelope", which consists of applying a multiplicative function to the waveform, altering only its amplitude.

The basic idea is to control the sound intensity by simulating the natural behavior of acoustic sounds. Typically, an ADSR envelope is defined as a function with a domain between 0 and 1, which is applied to the generated waveform:

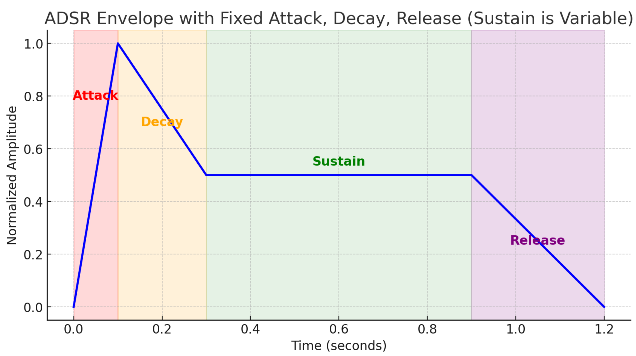

Figure 4: ADSR linear envelop.

The ADSR envelope can be understood as an amplitude function:

$$y(t) = e(t) \cdot s(t)$$

Where:

- $s(t)$: the raw generated sound wave.
- $e(t)$: the ADSR envelope function (normalized amplitude ranging from 0 to 1).
- $y(t)$: the resulting sound.

Although real audio samples can show a wide variation of envelope behaviors, it is common practice to use a linear function to achieve the goal, since the human ear is more sensitive to abrupt changes and to relative differences. However, for longer variations, the ear can still perceive the change, which is why some synthesizers allow the use of exponential or logarithmic functions as well.

2.2 FM Synthesis

As shown in the previous sections, one of the most popular sound synthesis techniques is Frequency Modulation Synthesis (FM Synthesis), which was introduced by John M. Chowning in his famous paper "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation", published at the Stanford Artificial Intelligence Laboratory.

Chowning explored the effects of frequency modulation when applied with high frequencies, close to the audible range, instead of the traditional use of inaudible frequencies.

In fact, the concept of frequency modulation was already well known before this paper; however, it had been used mainly in radio transmission of music or to apply slight distortions through low-frequency oscillators (LFOs), producing effects such as vibrato.

As Chowning himself pointed out in his paper, when the technique is applied to vary the frequency of the carrier wave through a high-frequency modulator wave, it "results in a surprising control of audio spectra" (Chowning).

In summary, considering only two oscillators (the main one, called the carrier, and the modulator) the instantaneous frequency of the output audio can be expressed by the following function:

$$f(t) = f_c + I \cdot \sin(2\pi f_m t)$$

Where:

- $f(t)$: the instantaneous frequency of the generated sound wave.
- f_c: the original frequency of the carrier wave.
- I: the modulation index, which defines the intensity of the modulation (the greater the value, the more harmonics are generated).
- f_m: the frequency of the modulator wave (which determines the spacing of the harmonics).
- t: the time variable.

The most important point, however, is that this simple manipulation results in symmetric sidebands around the carrier frequency, spaced at multiples of f_m.

For a more intuitive understanding, if a carrier frequency f_m is modulated by a frequency f_m, the resulting spectrum will contain:

$$f_c \pm n f_m$$

For example, by choosing $f_c = 440$ and $f_m = 220$, the resulting wave will contain frequency components such as 220 Hz, 440 Hz, 660 Hz, 880 Hz, and so on, with decreasing amplitudes (the amplitudes follow the Bessel coefficients, but this discussion is beyond the scope of this article).

The last important characteristic of Frequency Modulation Synthesis concerns the ratio between the carrier frequency and the modulator frequency f_c/f_m, since this ratio directly influences the perceived audio result.

This ratio determines whether the generated harmonics align with the natural harmonic series (widely used by musicians) or not:

- If the ratio is an integer number, the harmonic sidebands (in the spectral view) will appear as integer multiples of f_c, generating harmonic sounds similar to those of real instruments (consistent with traditional music theory).
- If the ratio is not an integer value, the harmonic sidebands will not align perfectly with f_c, and the generated sound may appear metallic or dissonant (this effect is useful for creating sounds such as bells or for producing special audio effects).

2.3 Audio comparison

The task of audio comparison is not straightforward, especially when the goal is to achieve equality in terms of human perception. It can be complex, computationally demanding, and inherently subjective.

However, as described in the section on timbre, one intuitive and efficient way to approach this problem is by comparing the audio in its spectral representation, since timbre is precisely defined by this combination of frequencies.

That said, this approach alone is not sufficient, because the Fourier Transform considers the entire audio signal at once, mixing different parts of a sound sample. For example, if a sample contains three different pitches in sequence, its frequency-domain representation will show them simultaneously, which may give the impression that the timbre is a mixture of all these pitches (similar to a chord rather than a sequence).

In fact, according to Claesson (2019), FFT-based spectral comparison is valid but only a part of the solution. He highlights the following additional metrics:

- FFT Distance: This consists of computing the Fast Fourier Transform of the two samples being compared and then calculating the normalized Euclidean distance between them (the maximum distance is 1, and the minimum is 0).
- Short Time Fourier Transform (STFT) Distance: Equivalent to the FFT Distance, but calculated over short, overlapping slices of the sample, which provides time-localized spectral information.
- Log-Mel Spectogram Distance: Similar to the previous metrics, but based on the Log-Mel Spectrogram, which is essentially an adaptation of the FFT spectrum that incorporates human auditory perception. In summary, the Log-Mel Spectrogram is obtained by computing the STFT, mapping the frequencies onto the Mel scale (explained later in this document), and finally projecting the amplitudes onto a logarithmic scale.

In addition to these three metrics, Claesson also recommends the use of Euclidean Distance of the Envelope to compare audio envelopes. This approach can be very useful, but since this work is focused on the similarity between original and synthesized audio in terms of timbre, the envelope comparison, while important for real-world applications, falls outside the scope of the present study.

2.3.1 Mel-Scale Representation. The Mel scale was proposed by Stevens, Volkmann, and Newman in 1937 (Claesson, 2019), and its main purpose is to reflect the human perception of frequency.

The key difference from the linear scale is that, as the values on the scale increase, the perceptual difference between two adjacent points becomes smaller (and the corresponding difference in hertz is also reduced).

In fact, the Mel scale maps linear frequencies onto a logarithmic scale. A frequency f s converted to the Mel scale through the following formula:

$$\mathrm{Mel}(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

For a better understanding of this conversion, consider the following graph, which compares the linear hertz scale with the Mel scale:

It is a very useful metric for audio comparison, as it takes into account the human auditory perception of frequency.

2.4 Convolutional Neural Networks (CNN)

2.4.1 2D Convolutional Neural Networks. In summary, a convolutional neural network is based on the idea of filters.

Before the introduction of CNNs, was already known that many computer vision problems can be solved through the evaluation of image features, which become more evident depending on the filters applied.

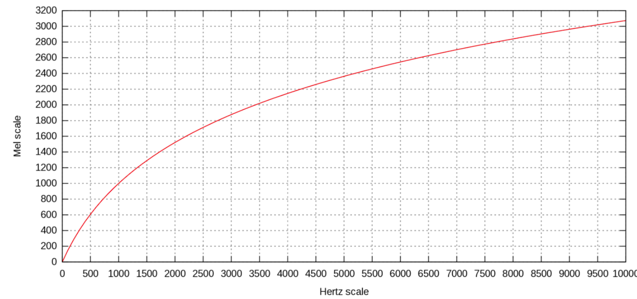

Figure 5: Comparision between linear hertz scale and Mel scale (Claesson, 2019).

For exampe, to classify differe animals, just the silhouettes can be enough. Of course, the fewer features considered, the higher the chance of mistakes, but, as popular saying goes: "if it walks like a duck, quacks like a duck, flies like a duck… it's a duck!". That's the basic idea.

So, the CNNs was concept in such a way that several layers of filters of fixed size (chosen by the network designer) are defined, in addition to some pooling layers (which will be discussed later).

These filter layers are also organized according to the designer's choices (usually based on many experiments) and can be applied in sequence, in parallel, using different combinations, and so on.

However, it is important to note that the filters serve as the link between the different representations of the input image built within the network.

That is, although the network still takes an image as input, and although this image is processed to extract the various features needed for the problem, there are no dense interconnections between these image representations.

In fact, what interposes between one image representation and another is precisely a CNN filter. And even this filter is not densely connected to the image representations.

In practice, the filter is not connected to any specific part of the image. Instead, it is "slid" across the image, performing a Hadamard multiplication (element-wise product) between the filter and each frame of the image (of the same size), producing, as output, a "new filtered image" (multiplication, element by element, of two matrices of the same size, with a final sum of all elements — and, typically, with the application of an activation function to the result).

Visually, the work can be represented as follows:

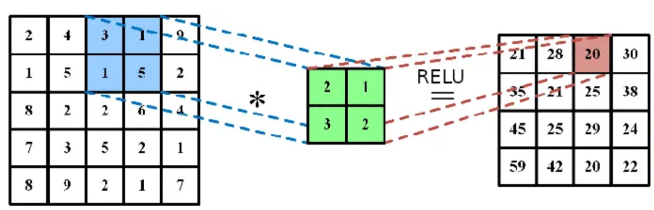

Figure 6: Hadamard product between a filter and an input layer slice (Adapted from ResearchGate, 2024)).

And it is precisely from this sliding operation that the network architecture takes its name, since this action corresponds to the mathematical operation of convolution:

To an inout image I, a filter (or Kernel) K, the convolution output O, in a point (i, j) is given by:

$$O(i, j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} K(m, n) \times I(i + m, j + n)$$

Where:

- $K(m, n)$: the filter element on position (m, n).
- $I(i+m, j+n)$: the element of the input on position of current window.
- M, N: the height and width of the filter.
- $O(i, j)$: the output element on position (i, j).

Thus, CNNs can perform image filtering while at the same time reducing the amount of memory required compared to traditional fully connected feed-forward networks (because memory is allocated only for the filters and their results, rather than for each individual interconnection weight).

After extracting the necessary features from the data, a small traditional dense (multi-layer) network is added after the convolution (and pooling) layers. This dense network is responsible for identifying the patterns required to predict the target variable (whether for classification or regression).

A CNN is typically divided into two stages. The first stage consists of the application of filters (along with pooling layers), while the second stage applies a fully connected network. In this second stage, the output of the first part undergoes a flattening process so that it can be used as input to the fully connected layers, which handle the usual tasks of classification or regression.

3 Related work

4 Proposed approach

The entire project idea can be explained as follows:

Use AI to guide an FM synthesizer to simulate any instrument.

Or, in a more complete explanation:

Use a Convolutional Neural Network (CNN) to analyze an instrument's sound sample and predict the correct parameters for an FM synthesizer to re-synthesize the "same" sound, allowing the synthesizer to simulate that instrument playing any song.

However, this idea is still very abstract, and to much comprehensive, then consider the following steps to achieve the project's goal:

(1) Implement a simple FM synthesizer, which could be later used both for generate the dataset, and to test the ability to generate the same sound.
(2) Generate an audio dataset with 20,000 random audio samples, each created by randomly selecting FM synthesizer parameters, essentially determining the synthesizer parameters and recording the tuples: (parameter values; audio sample).
(3) Train a CNN to predict the correct parameters for the same FM synthesizer, given only an audio sample.
(4) Evaluate the model results using the NSynth dataset to test the model's generalization capabilities.

4.1 Implementing the FM Synthesizer

There are several famous FM synthesizers on the market, but most are not open-source. Additionally, commercial synthesizers often have a vast number of parameters that can be used to control the audio production process. For example, Native Instruments' FM8 can use 1,000 (Claesson, 2019) parameters, and Teenage Engineering's OP-1 synthesizer can be configured with up to 10^{76} parameters (Claesson, 2019).

This incredibly large number of parameters is intended to give users fine-grained control over the generated audio, allowing for a wide variety of pitches (simulating a broad range of instruments and even non-instrument sounds, like bird songs or car horns, among others).

However, since the scope of this project is not focused on a commercial solution but rather on research purposes, a simple FM synthesizer will suffice.

Therefore, this work will implement a basic FM synthesizer with only 6 operators, inspired by the classic Yamaha DX7, which was one of the most disruptive milestones in the history of electronic music composition [TODO]. In fact, this simple setup is enough to cover all FM synthesis theories and achieve good performance during AI model evaluation.

In summary, the implemented synthesizer will consist of:

- 5 modulator operators
- 1 carrier operator
- 1 ADSR envelope
- 1 high/low pass filter
- Only 1 synthesis algorithm

4.1.1 Generating the dataset. This project will use two datasets, each for a specific purpose, the first being the generated dataset, used for training, and the second being the NSynth dataset, used for evaluate the model's generalization capabilities.

One of the most difficult steps in the KDD process is obtaining and processing data. However, for the purposes of this work, a good approach can be used. Similar to the Claesson approach (Claesson, 2019), since the goal is to achieve good control over the synthesizer parameters, it is sufficient to generate many samples of synthesizer parameters and their respective audio outputs.

Therefore, the main dataset will consist of 20,000 samples generated by the implemented FM synthesizer, with the following characteristics:

- 1 second in duration
- Monophonic samples (1-channel audio)
- 22 kHz audio sample rate

No specific control will be imposed on pitch or sound envelope formats, as the goal is for the model to learn as much control as possible.

The expectation is to achieve a learning level where the model can accurately predict the parameters for any sound that can be generated by the implemented FM synthesizer.

4.1.2 External dataset. In addition to the generated dataset, an external dataset named NSynth, from Google's Magenta project, will be used.

This dataset configuration is similar to the previous one, and the main purpose of using it is to achieve good generalization capabilities with the model. This is because the dataset contains a lot of real instrument samples.

Below are the dataset characteristics:

- 4 seconds in duration
- Monophonic samples (1-channel audio)
- 16 kHz audio sample rate
- 1,006 instruments
- Pitches varying between 27.5 Hz and 4,186.01 Hz
- The envelope includes a sustain for up to 3 seconds and a decay for 1 second
- 108,978 acoustic samples; 110,224 electronic samples; 86,777 synthetic samples; totaling 305,979 samples

5 Training the CNN

There are many experiments to conduct during the model training, but at a high level, there are two main approaches that will be explored:

(1) Training based on the generated random dataset.
(2) Training based on the NSynth dataset.

The major difference between these two approaches is that, in the first, the FM synthesizer parameters are known before training, whereas in the second, the parameters are unknown, and the implemented synthesizer must be used during the training process.

Both approaches will be detailed below, but it is important to highlight that the chosen neural network architecture is the Convolutional Neural Network (CNN).

The CNN architecture has shown good results when dealing with data domains where there are direct correlations between different portions of the same data sample [TODO]. It has demonstrated particularly strong performance in image and sound processing [TODO], and in the music field, various problems (such as genre or pitch prediction, for example) have benefited from this architecture [TODO].

Therefore, this architecture was chosen for this project mainly because of the central idea that the direct use of sound information in the time domain rather than in the frequency domain (as in Claesson's work [TODO]) can achieve better results. CNNs have been performing very well when applied to audio in the time domain [TODO]. Thus, this hypothesis will be effectively tested in this work.

5.1 Training based on the generated random dataset

analisar as diferentes propostas

In this approach, the training process uses only the generated random dataset. The main advantage is that, in this dataset, we already have the correct parameters to generate each sample sound. Therefore, the training process is straightforward and should yield good performance. Figure **??** presents the macro schema for this training process.

5.2 Training based on the NSynth dataset

o problema é que não temos o gabarito. O dataset tem amostras de instrumentos e jsons falando sobre o instrumento.

Figure 7: Training process, based on the generated random dataset

images/cnn_simple_training.jpg
usp:logo

In this approach, the training does not use the generated random dataset but instead utilizes the NSynth dataset.

The main idea is to learn directly from real instrument samples, which may achieve better generalization capabilities.

Below is the macro schema for this training process:

Figure 8: Training process, based on the NSynth dataset

images/cnn_training.jpg
usp:logo

However, as shown in the schema, there are two challenges in this approach:

- The implemented FM synthesizer must participate in the training process, which may penalize the training performance.
- The loss function should consider sound similarity measures, as the parameters cannot be evaluated directly, and the synthesized sound needs to be assessed.
  - The similarity measures will be presented in the next section.

5.3 Mixed Training Approach

In addition to the previously presented approaches, another experiment can combine both.

In summary, the third idea is to use the generated dataset for training, similar to the first approach. Then, the "transfer learning principle" will be applied by freezing the neural network weights, adding more layers, and training again, but this time using the NSynth dataset, following the same idea as the second approach.

The primary intention is to achieve better performance, as only a few layers will be trained in the most complex way.

6 Evaluating results

As introduced in the "Theoretical Background" section, the following audio similarity measures will be used in this work:

- FFT-distance: Used to compare the frequency components present in an audio signal.
- STFT-distance: Used to compare the variation of frequency components over time.
- Log-mel-spectrogram-distance: Used for a similar purpose as FFT-distance, but typically models human cognitive pitch perception more accurately.
- Envelope-distance: Used to compare the envelope characteristics of two audio signals.

These metrics can be applied in various ways during the research, especially when training based on the NSynth dataset. However, the primary goal of these metrics is to perform a final test on the model's generalization capability. That is, after the model training, the NSynth dataset (or a part of it) will be used to determine if the model is able to re-synthesize real instruments.

7 Applying to a Brazilian Instrument

Since the perception of audio similarity is primarily a cognitive process, it is inherently subjective. Therefore, this step is a natural extension of the proposed initiative, as the model will be applied to a real-world challenge.

Furthermore, the techniques will be applied to a Brazilian instrument called the *Rabeca*, and some final musical pieces will be produced, further situating this project within its national context.

To achieve this purpose, the following steps will be applied:

(1) Select an audio sample played with the *Rabeca*.
(2) Use the model to predict the FM Synthesizer parameters.
(3) Implement a simple code to synthesize internationally famous songs (such as Bach or Vivaldi), but played with a typical Brazilian instrument.
(4) Publish the songs (for future research purposes).

The final goal of this step is to make the results available for future work, potentially guiding an opinion poll with real people, providing a cognitive dimension to the evaluation process.

8 Experiments

8.1 Plan

-dataset -protococlos de avaliação

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

8.2 Results

9 Conclusion

10 Tables

The "`acmart`" document class includes the "`booktabs`" package. Table captions are placed *above* the table.

11 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

Inline equations: $\lim_{n\rightarrow \infty}x=0$, set here in in-line math style.

Numbered equation, shown as an inline equation above:

$$\lim_{n\rightarrow \infty}x=0 \tag{1}$$

Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty}x_i=\int_{0}^{\pi+2} f \tag{2}$$

12 Figures

The "`figure`" environment should be used for figures. One or more images can be placed within a figure. If your figure contains third-party material, you must clearly identify it as such, as shown in the example below.

Your figures should contain a caption which describes the figure to the reader.

Figure captions are placed *below* the figure.

Every figure should also have a figure description unless it is purely decorative. These descriptions convey what's in the image to someone who cannot see it. They are also used by search engine crawlers for indexing images, and when images cannot be loaded.

A figure description must be unformatted plain text less than 2000 characters long (including spaces). **Figure descriptions should not repeat the figure caption – their purpose is to capture important information that is not already provided in the caption or the main text of the paper.** For figures that convey important and complex new information, a short text description may not be adequate. More complex alternative descriptions can be placed in an appendix and referenced in a short figure description. For example, provide a data table capturing the information in a bar chart, or a structured list representing a graph.

Figure 9: 1907 Franklin Model D roadster. Photograph by Harris & Ewing, Inc. [Public domain], via Wikimedia Commons. (https://goo.gl/VLCRBB).

For additional information regarding how best to write figure descriptions and why doing this is so important, please see https://www.acm.org/publications/taps/describing-figures/.

Acknowledgments

This work is supported by *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES), *Conselho Nacional de Desenvolvimento Científico e Tecnológico* (CNPq), grants 406417/2022-9, 102475/2024-5, and 312209/2022-3, and *Fundação de Amparo à Pesquisa do Estado de São Paulo* (FAPESP), grant 2020/09835-1 (CPA IARA).

References

[1] Myriam KRASILCHIK, Celso de Barros Gomes, Francisco Inácio Homem Melo, Heliodoro Teixeira Bastos Filho, Julio Roberto Katinsky, and Tupã Gomes Corrêa. 1996. A USP e sua identidade visual. Tech. rep. São Paulo. http://www.scs.usp.br/identidadevisual/wp-content/uploads/myriamkrasilchik_1996.pdf.