Audio Sample Generation using Diffusion Models

Eason Wang
Hina Goto
Ram Adithya
Dwij Ravikumar

Goal

- Build audio diffusion model that is able to generate different types of drum samples (e.g. kick, snare, hi-hat)

- Slightly more ambitious goal: make the model be able to generate samples of tonal instruments (e.g. synths, keys, bass), melodic or chordal

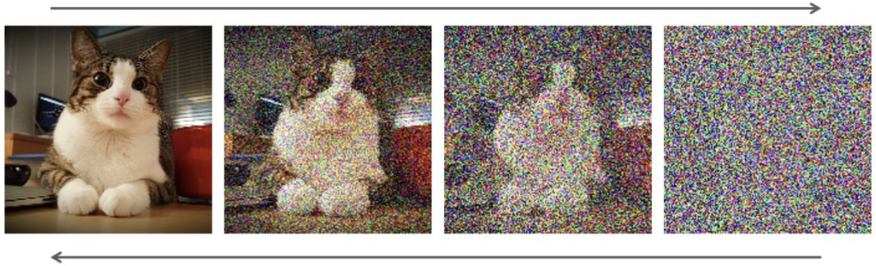
- Even more ambitious goal: make the model prompt-based and be able to take descriptive language input to create more customized samples

Understanding Diffusion Models with Images

- **Initialization with Noise:** Diffusion models starts by converting an image into pure noise through a series of steps, gradually adding Gaussian noise to the original image.

- **Training Phase:** During training, the model learns how to reverse the noise addition process. It is trained to predict the noise that was added at each step, progressively removing it from the noised image.

- **Reverse Process:** In generation, the model starts with random noise and uses the trained model to repetitively remove the predicted noise, thereby reconstructing the image step-by-step.

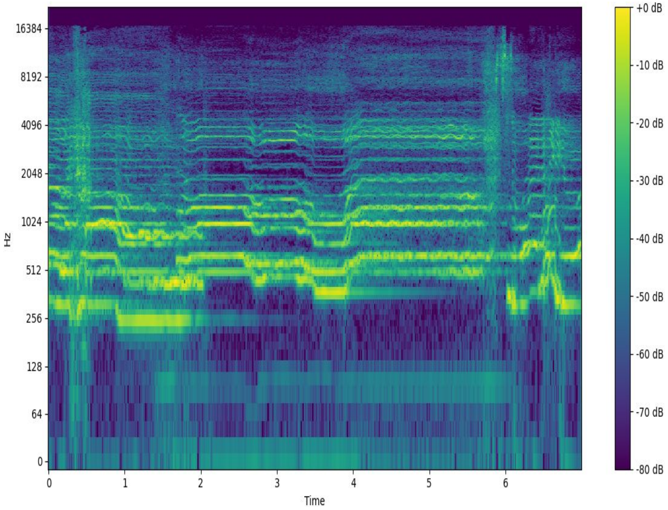


Introduction to Audio Sample Generation

- **Overview of Diffusion Models:** Diffusion models are a type of generative deep learning algorithm that create high-quality images by gradually refining noise into detailed visuals through a reverse diffusion process.

- **Application in Audio:** These models can be applied in audio to generate realistic sounds and music by transforming noise into coherent audio sequences through reverse diffusion processes.

- **Scope of Presentation:** This presentation will cover audio preprocessing and will look into future potential applications and their implications in enhancing audio quality and generating realistic soundscapes.



Differences between using Diffusion Models for Images and Audio

Aspect	Images	Audio
Nature of Data	Spatial data and Two-dimensional	Time-series data and one-dimensional
Features	Edges, colors, textures, depth.	Frequency, timbre, rhythm(tempo), Intensity and more...
Generation Process	Repeatedly denoise a grid of pixels to form images.	Repeatedly denoising a waveform to produce specific sounds

Audio Data Preparation

- When training diffusion models for audio, we focus on three main types of data: raw waveforms, spectral features, and MIDI(Musical Instrument Digital Interface) data. Each type provides different insights into the sound's characteristics, which will be important for better model training.

- Before training, preprocessing is **essential**. We normalize audio levels, remove silent segments, and apply data augmentation. These steps are vital to reduce noise and ensure that our model learns from clean, consistent data, thereby enhancing its ability to generalize across different audio inputs.

- we transform audio signals into a spectral format. This process allows the diffusion model to capture and learn from the frequency components of the audio, facilitating more effective and realistic audio generation. This is crucial for the model to understand and recreate complex audio patterns.

Generating Audio Samples

- We start with the denoised data, which has been processed by our diffusion model. From this, we generate the final waveforms.

- This step shows how the sound waves will physically manifest when played, making sure the output closely matches the real-world audio.

- By adjusting some parameters within our model, we can change the key audio characteristics such as pitch, volume, and duration. This control allows us to change the audio output to specific requirement, which makes it really versatile!

Challenges in Audio Generation

- **High-Quality Data Requirements:** The diversity in audio types, from music to speech to environmental sounds, will be a significant challenge. Each type has unique characteristics that the model must capture, due to this we must make some complex tunings to the model.

- **Computational Costs:** Audio diffusion models requires a lot of computational power due to the need to process long sequences and perform numerous iterations to refine audio output. This high resource requirement can sometimes limit their applicability in environments where computational resources are very less.

- **Generating Audio in Real Time:** Generating high-quality audio in real-time is particularly challenging. Since the model has an iterative nature, where the quality progressively gets better, it will limit its ability to produce the audio in small amount of time.

Applications of Audio Diffusion Models

- **Music Generation:** Automatically creates original music compositions, matching various styles.
- **Speech Synthesis:** Makes text-to-speech systems better by generating natural-sounding, expressive voices for multiple languages.
- **Sound Effects Production:** It can be used to generate realistic sound effects for video games, films, and virtual reality environments.

Samples

- Type of files we’re working with: **.wav files** (audio files)
- One short recordings (~1.0 sec) of drum instruments
- Kick, clap, snare, hihat. Etc
- We’re using files I already had on my computer
- Can be obtained through DAW(music production software), external packs, freesound.org
- Or record instruments

Pre-processing

- Read an audio file (obtain basic info and data as an array)

```
sample_rate, samples = wavfile.read(file)
# Load the audio file
y, sr = librosa.load(file, sr=sample_rate)
```

- Adjust the length of the data

```
# Adjust the length
y = np.pad(y, (0, 44100-y.shape[0]))
```

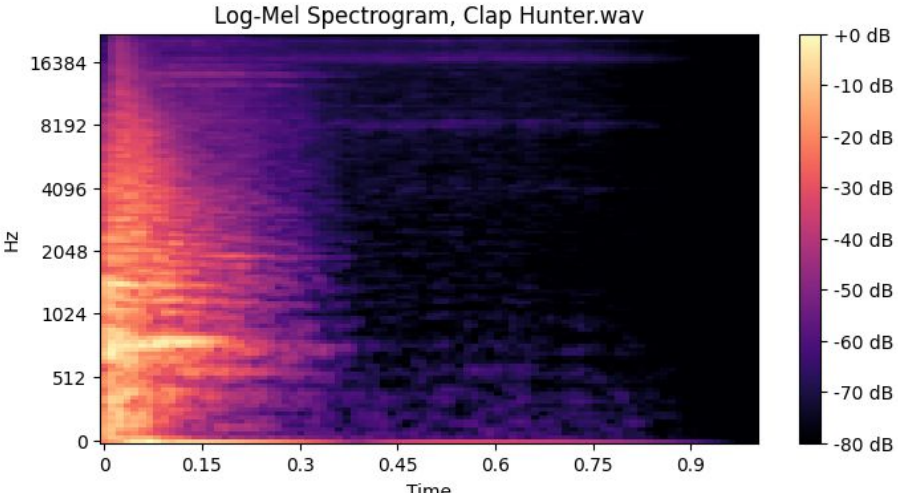
Pre-processing

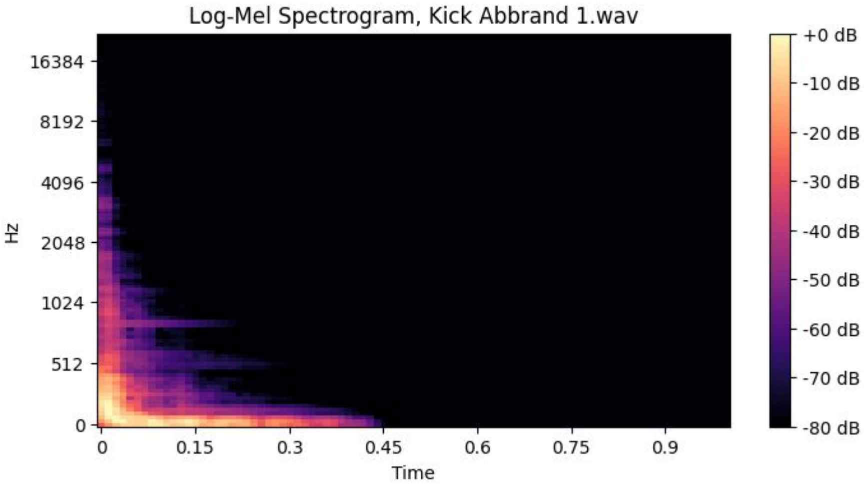
- Generate spectrogram (our training data) using an existing package

```
def compute_logmel_spectrogram(y, sr, n_mels=128, hop_length=512):
    mel_spectrogram = librosa.feature.melspectrogram(y=y, sr=sr, n_mels=n_mels,
hop_length=hop_length)

    logmel_spectrogram = librosa.power_to_db
(mel_spectrogram, ref=np.max)

    return logmel_spectrogram
```

Spectrogram we generated

- Spectrogram shows the strength of frequency over time
- Higher frequency corresponds to higher pitch
- Larger decibel (Lighter color) = louder
- This shows what frequency is loud (or quiet) at a given time





Next steps

- Explore how to best add noise to our training data
- Train model for drum samples
- Make a drum track with generated samples?
- If all successful, explore instrument samples with pitch and harmony, and make model prompt based

Thank you!