Try it!

[RIFFUSION]

(noun): riff + diffusion

Riffusion was created by Seth Forsgren and Hayk Martiros as a hobby project.

You've heard of Stable Diffusion, the open-source AI model that generates images from text?

photograph of an astronaut riding a horse

Well, we fine-tuned the model to generate images of spectrograms, like this:

funk bassline with a jazzy saxophone solo

Try it!

The magic is that this spectrogram can then be converted to an audio clip:

0:00 / 0:05

🔥🔥🔥😱

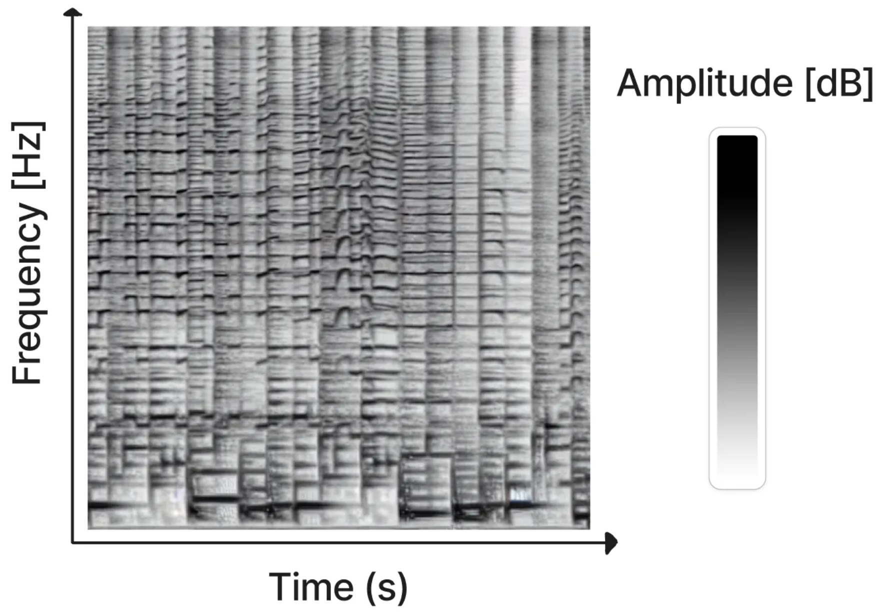
Really? Yup.

This is the v1.5 stable diffusion model with no modifications, just fine-tuned on images of spectrograms paired with text. Audio processing happens downstream of the model.

It can generate infinite variations of a prompt by varying the seed. All the same web UIs and techniques like img2img, inpainting, negative prompts, and interpolation work out of the box.

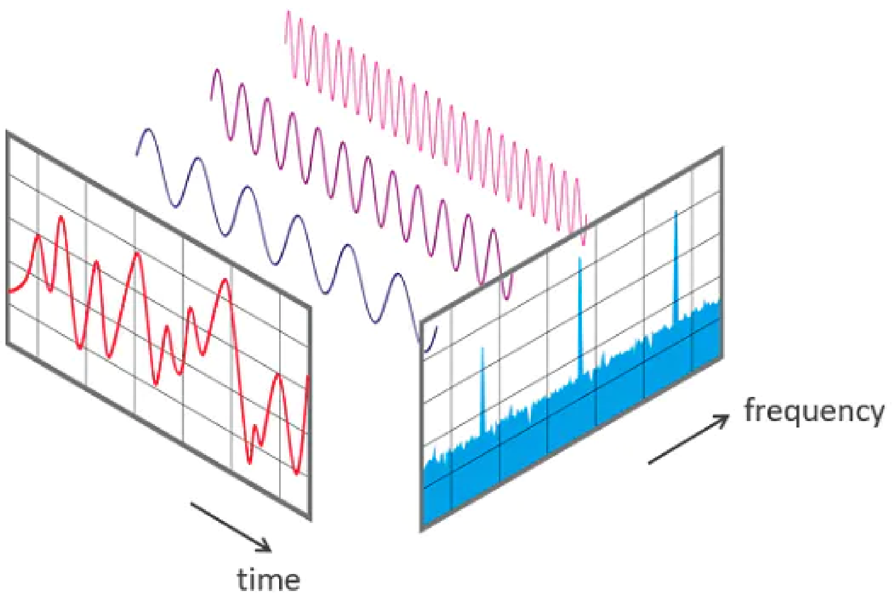
Spectrograms

An audio spectrogram is a visual way to represent the frequency content of a sound clip. The x-axis represents time, and the y-axis represents frequency. The color of each pixel gives the amplitude of the audio at the frequency and time given by its row and column.



Try it!

The spectogram can be computed from audio using the Short-time Fourier transform (STFT), which approximates the audio as a combination of sine waves of varying amplitudes and phases.

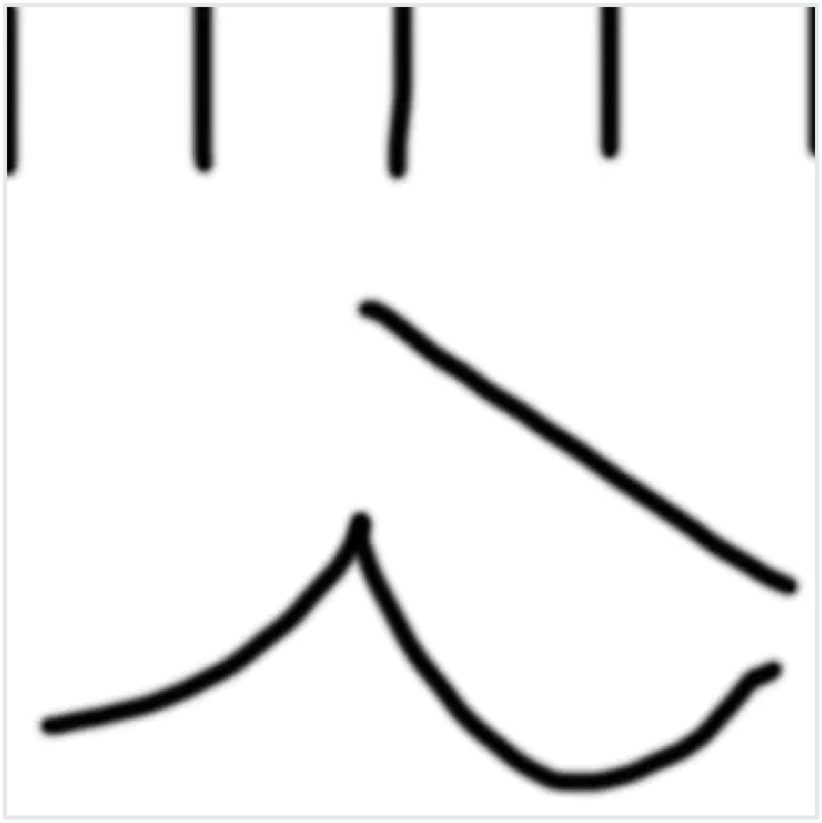


The STFT is invertible, so the original audio can be reconstructed from a spectrogram. However, the spectrogram images from our model only contain the amplitude of the sine waves and not the phases, because the phases are chaotic and hard to learn. Instead, we use the Griffin-Lim algorithm to approximate the phase when reconstructing the audio clip.

The frequency bins in our spectrogram use the Mel scale, which is a perceptual scale of pitches judged by listeners to be equal in distance from one another.

Below is a hand-drawn image interpreted as a spectrogram and converted to audio. Play it back to get an intuitive sense of how they work. Note how you can hear the pitches of the two curves on the bottom half, and how the four vertical lines at the top make beats similar to a hi-hat sound.

Try it!



0:00 / 0:05

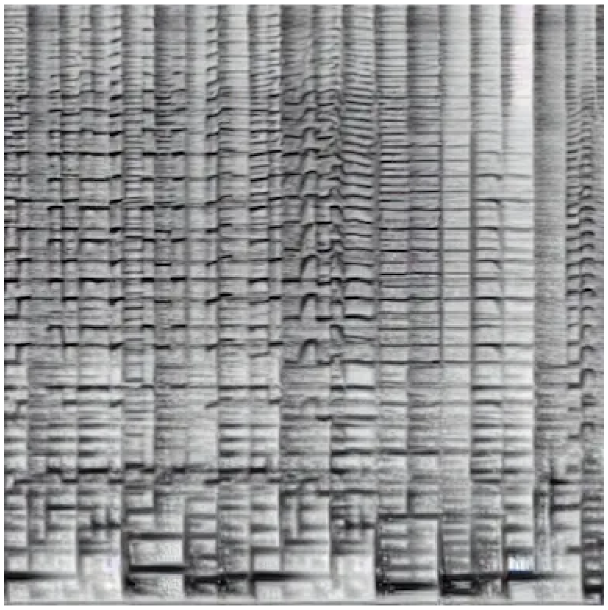
We use Torchaudio, which has excellent modules for efficient audio processing on the GPU. Check out our audio processing code here.

Image-to-Image

With diffusion models, it is possible to condition their creations not only on a text prompt but also on other images. This is incredibly useful for modifying sounds while preserving the structure of the an original clip you like. You can control how much to deviate from the original clip and towards a new prompt using the denoising strength parameter.

For example, here is that funky sax riff again, followed by a modification to crank up the piano:

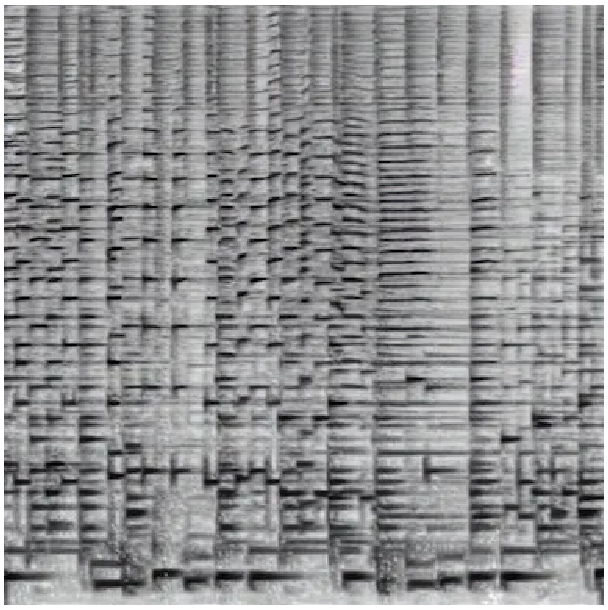
funk bassline with a jazzy saxophone solo



0:00 / 0:05

Try it!

piano funk



0:00 / 0:05

The next example adapts a rock and roll solo to an acoustic folk fiddle:

rock and roll electric guitar solo

Try it!

0:00 / 0:05

acoustic folk fiddle solo

0:00 / 0:05

Looping and Interpolation

Generating short clips is a blast, but we really wanted infinite AI-generated jams.

Let's say we put in a prompt and generate 100 clips with varying seeds. We can't concatenate the resulting clips because they differ in key, tempo, and downbeat.

Try it!

Our strategy is to pick one initial image and generate variations of it by running image-to-image generation with different seeds and prompts. This preserves the key properties of the clips. To make them loop-able, we also create initial images that are an exact number of measures.

However, even with this approach it's still too abrupt to transition between clips. Multiple interpretations of the same prompt with the same overall structure can still vary greatly in their vibe and melodic motifs.

To address this, we smoothly interpolate between prompts and seeds *in the latent space of the model*. In diffusion models, the latent space is a feature vector that embeds the entire possible space of what the model can generate. Items which resemble each other are close in the latent space, and every numerical value of the latent space decodes to a viable output.

The key is that it's possible to sample the latent space between a prompt with two different seeds, or two different prompts with the same seed. Here is an example with the visual model:

Try it!

We can do the same thing with our model, which often produces buttery smooth transitions, even between starkly different prompts. This is much more interesting than interpolating the raw audio, because in the latent space all in-between points still sound like plausible clips. The figure below is colorized to show the latent space interpolation between two seeds of the same prompt. Playing this sequence is much smoother than just playing the two endpoints. The interpolated clips are often diverse and have their own riffs and motifs come and go.

Here is one of our favorites, a beautiful 20-step interpolation from **typing** to **jazz**:

0:00 / 1:42

And another one from **church bells** to **electronic beats**:

0:00 / 1:42

Interpolation of **arabic gospel**, this time with the same prompt between two seeds:

Try it!

0:00 / 0:51

The huggingface diffusers library implements a wide range of pipelines including image-to-image and prompt interpolation, but we needed an implementation for interpolation combined with image-to-image conditioning. We implemented this pipeline, along with support for masking to limit generation to only parts of an image. Code here.

Interactive Web App

To put it all together, we made an interactive web app to type in prompts and infinitely generate interpolated content in real time, while visualizing the spectrogram timeline in 3D.

As the user types in new prompts, the audio smoothly transitions to the new prompt. If there is no new prompt, the app will interpolate between different seeds of the same prompt. Spectrograms are visualized as 3D height maps along a timeline with a translucent playhead.

The app is built using Next.js, React, Typescript, three.js, Tailwind, and Vercel.

The app communicates over an API to run the inference calls on a GPU server. We used Truss to package the model and test it locally before deploying it to Baseten which provided GPU-backed inference, auto-scaling, and observability. We used NVIDIA A10Gs in production.

Try it!

If you have a GPU powerful enough to generate stable diffusion results in under five seconds, you can run the experience locally using our test flask server.

Code

- Web app: https://github.com/hmartiro/ riffusion-app
- Inference server: https://github.com/hmartiro/ riffusion-inference
- Model checkpoint: https://huggingface.co/ riffusion/riffusion-model-v1

Prompt Guide

Like other diffusion models, the quality of the results depends on the prompt and other settings. This section provides some tips for getting good results.

Seed image - The app does image-to-image conditioning, and the seed image used for conditioning locks in the BPM and overall vibe of the prompt. There can still be a large amount of diversity with a given seed image, but the effect is present. In the app settings, you can change the seed image to explore this effect.

Denoising - The higher the denoising, the more creative the results but the less they will resemble the seed image. The default denoising is 0.75, which does a good job of keeping on beat for most prompts. The settings allow raising the denoising, which is often fun but can quickly result in chaotic transitions and tempos.

Prompt - When providing prompts, get creative! Try your favorite artists, instruments like saxophone or violin, modifiers like arabic or jamaican, genres like jazz or rock, sounds like church bells or rain, or any combination. Many words that are not present in the training data still work because the text encoder can associate words with similar semantics. The closer a prompt is in spirit to the seed image And BPM, the better the results. For example, a prompt for a genre that is much faster BPM than the seed image will result in poor, generic audio.

Try it!

Prompt Reweighting - We have support for providing weights for tokens in a prompt, to emphasize certain words more than others. An example syntax to boost a word is (vocals:1.2), which applies a 1.2x multiplier. The shorthand (vocals) is supported for a 1.1x boost or [vocals] for a 1.1x reduction.

Parameters can also be specified via URL, for example: https://www.riffusion.com/? &prompt=rainy+day& denoising=0.85& seedImageId=og_beat

Examples

The app suggests some of our favorite prompts, and the share panel allows grabbing the spectrogram, audio, or a shareable URL. We're also posting some favorites at /r/riffusion.

Here are some longer-form interpolations we like:

Sunrise DJ Set to **hard synth solo**:

0:00 / 1:42

Detroit Rap to **Jazz**:

0:00 / 1:42

Cinematic New York City in a Dust Storm to **Golden hour vibes**:

0:00 / 1:42

Techno beat to **Jamaican rap**:

0:00 / 0:51

Try it!

Fantasy ballad, female voice to **teen boy pop star**:

0:00 / 0:51

Citation

If you build on this work, please cite it as follows:

```
@software{Forsgren_Martiros_2022,
  author = {Forsgren, Seth*
    and Martiros, Hayk*},
  title = {{Riffusion -
    Stable diffusion for
    real-time music generation}},
  url = {https://riffusion.com/about},
  year = {2022}
}
```