

CLAUDE CODE · AI · AUDIO

Claude is deaf, let it hear your video with earshot

earshot is an open-source tool that turns sound, which agents like Claude Code can't hear, into a report they can read. Installing it, getting a video's sound scorecard (critique), having Claude explain the report in plain words, bringing the audio to -14 LUFS, and its limits.

TIME	LEVEL	STEPS	PROMPTS
20 min	Beginner	6	3

Claude can edit your video, but it can't hear it. It drops in a keyboard sound effect without knowing whether it fits the text or your tone of voice. The volume rises and falls, and it doesn't notice. I edited my last few videos with Claude. When I let earshot listen, my audio turned out to be ~10 dB quieter than it should be. In this guide I wrote down, step by step, how I installed earshot, measured my own video, had Claude explain the report and fixed the audio. Press the **Copy** button at the top right of each command and prompt, paste it, carry on.

What we're making

We'll install earshot, get a video's sound scorecard and have Claude explain it in plain words. Then we'll fix the biggest problem: the volume.

What is earshot? [earshot](#) is an open-source tool that gives coding agents "ears". It takes an audio or video file and turns it into a report an agent can read: loudness (LUFS), peaks, what the sound is like, how well the sound effects line up with the cuts in the picture. It's written in Python, MIT licensed, and has ~3.3k stars on GitHub. It runs entirely on your computer's CPU; no graphics card needed.

How does it hear? Two open-source models run in the background:

- **CLAP:** places sound and text on the same map. That's how it measures how close a description like "mechanical key press" is to a sound.
- **AudioSet tagger:** compares the sound against 527 sound types and says what can be heard: speech, music, clicks and so on.

On top of these come measurements (loudness, peaks, frequency balance), which use no model.

At the end you'll have three things:

1. Your video's sound scorecard: which values are in range, which aren't, and what to fix first
2. A plain summary of that scorecard written by Claude
3. A video with fixed loudness

INGREDIENTS

- [Claude Code](#)
- **Python 3.10 or newer** and **ffmpeg**. If you don't have ffmpeg, Claude can install it.
- **~1.1 GB of free space.** Two models download on first use.
- A video or audio file to measure. I measured my own Instagram video.

TOOLS

01 Install earshot

Give Claude Code the repo link and ask it to install it:

PROMPT

Install this repo: <https://github.com/Leonxlnx/earshot>

1. Follow the steps in the README's "Install" section; use its own virtual environment (.venv).
2. Install torch with the CPU build, as the README says.
3. Before installing anything, tell me what you're about to do.
4. When done, run "earshot doctor --load" and tell me the result.

The install clones the repo, creates a virtual environment, and installs the CPU build of torch and earshot. On Windows the virtual environment's path is `.venv/Scripts/python`; on Mac and Linux it's `.venv/bin/python`.

02 Check the install

TERMINAL

```
earshot doctor --load
```

This checks the install, the models and the CPU settings. On the first run it downloads the two models (~1.1 GB). After that it reads them from disk every time and never goes online.

If the `earshot` command isn't found, the virtual environment isn't active. Telling Claude "run earshot from its virtual environment" is enough.

03 Get your video's sound scorecard

In the folder with your video:

TERMINAL

```
earshot critique video.mp4 --fast
```

`critique` compares your video with a profile built from 7 professional launch films. It says whether each metric is in range and puts a "fix this first" list at the top.

`--fast` skips two heavy checks, whether the sound is a real recording or synthesized and whether there's music, so the result comes in seconds instead of minutes. Run it once without `--fast` before you publish.

For my own video the summary was: 16 of 21 metrics in range, 1 critical and 1 major issue. The critical one was loudness:

- **My video:** -23.9 LUFS
- **Web and YouTube target:** around -14 LUFS

- **Difference:** ~10 dB. Viewers turn their phone up to hear the video, then down again for the next one.

Peaks reached -4.4 dB, so the audio was a whisper in one place and a bang in another.

What is LUFS? The average loudness as the ear perceives it. The closer the number is to zero, the louder: -14 is louder than -24.

04 Have Claude explain the report

The report is full of technical terms. Instead of working through all of them, I copied the output and pasted it into Claude Code:

PROMPT

Can you explain this technical analysis simply enough for me to understand? Start with the 3 most important problems and, for each, tell me in one sentence what I should do.
<paste the earshot critique output here>

What Claude pulled out for me: the audio is very quiet, there are too many emphasis sounds, the audio is fully mono, there's almost no bass, and the mids are a bit harsh. It also reminded me that two things aren't measured in `-fast` mode.

Mono audio isn't a problem for me. I make content for phones, and a phone speaker plays more or less as one channel anyway. You don't have to fix every finding; you decide which ones matter to you.

05 Fix the loudness

Loudness was the biggest problem, and it's also the easiest to fix. Ask Claude to bring the audio to -14 LUFS with ffmpeg:

PROMPT

Bring the audio of video.mp4 to -14 LUFS with ffmpeg, with true peak no higher than -1.5 dB. Don't touch or re-encode the picture. Name the result video_14lufs.mp4. When done, measure the new loudness with earshot measure and tell me.

`earshot measure` only measures and loads no models, so it takes seconds. That's how you see the fix worked.

The audio of this guide's own video was fixed the same way: from -21.2 LUFS to -14.1.

06 Optional: connect it as an MCP server

Every `earshot` command starts a new process and loads the models from scratch: 20-40 seconds each time. If you'll measure a lot on one video, connect earshot as an MCP server. The models load once and stay loaded:

TERMINAL

```
claude mcp add earshot --scope user -e EARSHOT_THREADS=3 --  
/path/to/earshot/.venv/bin/python -m earshot mcp
```

Replace the path with your own earshot folder. On Windows it ends in `.venv/Scripts/python`. On a slow machine, start Claude Code with `MCP_TIMEOUT=60000`.

What it hears and what it doesn't

Topic	Reliable?	Note
Loudness, peaks, fluctuation	Yes	Measurement, not a model
Sound effects in sync with picture cuts	Yes	
"Is this the sound I described?"	Yes for single, clear sounds	Only a hint inside a busy mix
Music genre, emotion, taste	No	The README says so too
What the speech says	No	It doesn't understand words

It mistook my emphasis for sound effects. When I imitated typing with my voice in the video, it read that as a sound effect. Expect this in speech-heavy videos.

The scorecard is based on launch films. `critique` compares your video with the style of 7 professional launch films. An Instagram video where you talk to the camera isn't a launch film, so some metrics may not fit you. Loudness and peak values apply to every video; start with those.

Who it's for

If your microphone is bad, your sound effects never sit right, or you can't get your video's overall level right, it's a good free starting point. If you're doing professional

work, still work with a sound engineer; earshot doesn't replace one.

TIPS

Every command waits half a minute. That's normal: the models load on every call. If you'll measure a lot, use the MCP server from step 6.

ffmpeg not found. earshot uses ffmpeg and ffprobe. Tell Claude "install ffmpeg and add it to PATH".

earshot: command not found. The virtual environment isn't active. Run it as `.venv/bin/python -m earshot ...` (on Windows `.venv/Scripts/python -m earshot ...`).

The result says "partial". You ran it with `--fast`, so the realness and music checks were skipped. Run it without `--fast` before you publish.

The report is too long. Put a limit in the prompt: "3 most important problems, one sentence each".

DONE?

- ☐ I installed earshot with the README's steps, in its own virtual environment
- ☐ `earshot doctor --load` finished without errors
- ☐ I got my video's scorecard with `earshot critique video.mp4 --fast`
- ☐ I had Claude explain the report in plain words

- ☐ I brought the audio to -14 LUFS and measured it with `earshot measure`
- ☐ I ran it once without `--fast` before publishing

If you get stuck, message me on Instagram and I'll add it to the guide.