DOT Music Production

Copilot conversation export

Open-source, headless music analysis, reference matching, mastering, and perceptual-quality architecture for an AI agent.

Generated from the current conversation thread.

User

Are there any open source public domain professional quality music analyzing comparison production mixing equilizing apps and models that I can instruct my AI agent to download install and learn?

Copilot

Yes. There are several strong open-source projects spanning research-grade music analysis, production, source separation, loudness measurement, and reference mastering.

Essentia was initially recommended for music-information retrieval and features such as tempo, key, loudness, spectral descriptors, similarity, and machine-learning inference. Sonic Visualiser was identified for detailed visualization and analysis. Matchering was highlighted for reference-based mastering. Ardour and Audacity were mentioned as production/editing environments, while CLAP/OpenL3 were suggested for learned audio representations, Demucs for stem separation, FFmpeg for loudness/processing, and librosa for Python-based spectral and rhythm analysis.

A proposed autonomous workflow was: import a reference, separate stems, analyze features, measure loudness, compare spectra, suggest EQ changes, master, score similarity, and generate a production report.

User

My agent needs open source free to use for personal or commercial tools that run without a desktop or audio... I am using DOT on openai and need to upgrade its capabilities for music production asap

Copilot

That changes the recommendation substantially. DOT should favor command-line tools and Python/C libraries rather than desktop DAWs, and licensing needs to be treated separately from the fact that a project is open source.

Recommended core:
- FFmpeg as the headless audio engine for decoding, encoding, resampling, channel conversion, filtering, loudness analysis, normalization, statistics, and rendering. Most FFmpeg is LGPL 2.1+, while some optional components make a build GPL.
- libebur128 for EBU R128 loudness and true-peak measurement; its source is MIT licensed.
- SoX as a command-line DSP and batch-processing engine; its executable is GPL while libSoX is LGPL.
- Matchering as an optional reference-mastering worker. It is a Python/container system that matches a target against a reference using measures including RMS, frequency response, peak level, and stereo width; it is GPLv3.

Essentia was removed from the commercial-safe core because its official licensing distinguishes non-commercial open use from separate commercial licensing, and its pretrained models have non-commercial restrictions. Aubio was similarly excluded from the commercial-safe core because its project documentation calls out GPL licensing and commercial-product considerations.

The key architecture proposed for DOT was file-based rather than dependent on audio hardware:

```
INPUT -> inspect -> structured measurements -> compare target/reference -> DOT reasoning ->
processing recipe -> render -> reanalyze -> score improvement
```

Suggested atomic tools included inspect_audio, measure_loudness, analyze_spectrum, analyze_dynamics, analyze_stereo, compare_tracks, apply_eq, apply_dynamics, normalize_master, and render_audio. The central idea is to make the OpenAI model the reasoning layer while deterministic DSP tools become DOT's meters, console, and mastering rack.

User

Any way to teach DOT how to judge music quality and compare songs to analyze how to shift one song sound toward another reference style?

Copilot

Yes. The key is not to use one vague quality score. DOT should learn a multidimensional definition of production quality and treat reference matching as a measurable optimization problem.

Two separate concepts were proposed:

Technical quality: clipping/true peak, loudness, crest factor, dynamics, frequency balance, brightness/harshness proxies, sub/low/mid/high energy, stereo width/correlation, phase behavior, transient behavior, noise/silence, and section-to-section spectral consistency.

Perceptual/style similarity: timbre, instrumentation, density, texture, energy, production character, stem similarity, and overall sonic character.

LAION CLAP was identified as a promising learned perceptual representation. DOT can embed a target and reference and use embedding similarity as one signal, while keeping deterministic DSP measurements separate. Recent research also supports combining pretrained audio embeddings with source separation and instrument-wise similarity.

The strongest approach is stem-aware comparison: compare vocals to vocals, drums to drums, bass to bass, and other material to its counterpart rather than relying only on the finished stereo mix. DOT can then report differences such as vocal tonal balance, drum character, bass similarity, master dynamics, and high-frequency energy.

DOT should build a structured reference-delta vector covering spectral bands, loudness, width, dynamics, and perceptual similarity. The model then translates those deltas into engineering hypotheses rather than blindly copying a reference curve.

```
Most importantly, DOT should experiment: BASELINE -> analyze -> form hypotheses -> generate
multiple candidates -> analyze candidates -> reject artifacts -> score reference-distance
changes -> choose the best candidate -> refine -> repeat.
```

Quality and similarity should remain independent objectives. A useful conceptual score is: score = technical quality + weighted style similarity - artifact penalty - destructive-processing penalty. Hard safeguards should prevent clipping, phase damage, excessive pumping, unwanted distortion, destroyed transients, collapsed stereo image, and excessive limiting.

Human preference can become another training signal. If DOT makes A/B/C masters and the user ranks them while giving reasons such as 'better punch', 'too bright', or 'overcompressed', those preferences can be stored alongside target features, reference features, processing settings, and rankings. Over time DOT can retrieve successful past decisions and adapt to the user's production aesthetic without necessarily retraining the language model.

RefMatch was identified as a project close to this idea: its package documentation describes DSP matching across spectral balance, loudness, dynamics, rhythm and harmony combined with CLAP embeddings, offline operation, local reference libraries, and dimension-specific matching. Music2Latent was not recommended for a commercial-safe core because its published license is CC BY-NC 4.0.

User

Is there a way to share this as a public link without a copilot sign in?

Copilot

Not reliably as an anonymous public Copilot link. Microsoft's sharing documentation for personal Microsoft accounts says recipients can open shared content by signing in with a Microsoft account. A separate conversation-history page says a generated link can be accessed by anyone with the link, creating ambiguity, so the recommendation was not to rely on native Copilot sharing as a no-login public webpage.

For genuinely public, no-login distribution, the suggested approach was to publish a sanitized copy as a GitHub Gist/repository, static webpage, publicly shared document, or publicly hosted PDF. A consolidated DOT Music Engineer architecture document was also suggested as a cleaner public artifact than the raw chat.

User

Can you output this thread as a PDF?