

Инструменты для детекции смеха, крика и резких звуков

Для анализа аудио со стримов можно использовать как готовые ML-модели общего назначения, так и специализированные инструменты. Так как скорость не критична и важен результат, стоит обратить внимание на уже обученные модели и библиотеки «из коробки». Ниже приведены основные варианты.

Предобученные нейросети на аудио (AudioSet, YAMNet, PANNs и др.)

- **YAMNet (TensorFlow)** – глубокая модель от Google, предобученная на базе AudioSet (521 класс звуков). YAMNet умеет распознавать такие события, как смех, лай собак, сирена и др. ¹. Модель принимает моно-волновой сигнал (16 кГц) и выдает вероятности наличия каждой из 521 категории. Среди меток AudioSet есть *Laughter*, *Baby laughter*, *Giggle* и *Screaming* ². Это готовый модуль (есть на TensorFlow Hub), легко установить через `pip` и запустить на ЦП (модель компактна, основана на MobileNet ¹). Для разных форматов аудио можно сначала конвертировать файлы в WAV (16 кГц) с помощью `ffmpeg` или библиотек типа TensorFlow I/O/ Librosa.
- **PANNs (PyTorch Audio Neural Networks)** – семейство предобученных CNN-моделей (например, Cnn14) автора Q. Kong, обученных на тех же 5000 часах AudioSet (527 меток) ³. PANNs позволяет делать *аудио-теггинг* и *детектирование звуковых событий* (sound event detection) ³. Модели доступны на GitHub; в репозитории есть скрипты командной строки для инференса по любому аудио-файлу. Вы скачиваете чекпоинт модели (публикуется на Zenodo), а затем запускаете, например, `python inference.py audio_tagging --model_type Cnn14 --audio_path=input.wav` – на выходе получите вероятности разных звуков (включая «Speech», «Music», «Animal» и т.д.). PANNs работают на ЦП (хотя имеют порядка десятков МБ и требуют Librosa и PyTorch), и могут распознавать акустические события, близкие к смеху и крику.
- **VGGish / AudioSet-выборки для смеха** – есть проекты, которые сами тонко балансируют AudioSet для смеха. Например, репозиторий `ideo/LaughDetection` использует предобученный векторизатор VGGish (AudioSet) и свою небольшую сеть для детекции смеха ⁴. Там предоставлены модели и скрипты для «живого» инференса: команда `python live_inference.py` слушает микрофон и помечает сегменты со смехом ⁵. Хотя это решение чуть сложнее в настройке (требуется получить `vggish_model.ckpt` и установить PortAudio), оно уже обучено специально на разделении «смех – не смех» ⁴.
- **Фундаментальные модели (HuggingFace и др.)** – существуют многофункциональные модели для аудио (разработка Tencent). Например, SenseVoice-Small – модель для распознавания речи

и «звукособытий» (audio event detection). Она умеет детектировать разные звуки HCI (фон, аплодисменты, смех, плач, чих и др.) ⁶. Эти модели открыты и могут запускаться локально, но они тяжелы по вычислениям. Если нужна «коробочная» система без доп. тренировки, можно рассмотреть так называемые аудио-пайплайны на HuggingFace (например, `pipeline("audio-classification")`) и выбрать модель, обученную на AudioSet. Но наиболее простые варианты – YAMNet/PANNs и специализированные библиотеки ниже.

Специализированные библиотеки для смеха и звуковых событий

- **jrgillick/Laughter-detection** – open-source библиотека на PyTorch для детекции и сегментации человеческого смеха в аудио ⁷. Включает готовые модели, обученные на реальных данных (Switchboard, AudioSet). Запуск прост: `python segment_laughter.py --input_audio_file=input.wav --output_dir=out/ ...`. На выходе скрипт выдаёт список интервалов (начало, конец) со смехом и опционально сохраняет соответствующие WAV-файлы ⁸. Проект поддерживает запуск на CPU (CUDA отключается) и требует только Librosa/PyTorch. В документации есть подробности по порогам, минимальной длительности фрагмента и т.д., но «из коробки» оно найдёт большинство случаев смеха ⁷ ⁸.
- **ideo/LaughDetection** – библиотека на Keras/TensorFlow для детекции смеха, основанная на AudioSet и VGGish ⁴. Здесь также есть скрипты для обработки готовых файлов: `python live_inference.py` позволяет анализировать аудио с микрофона или файла. Это более комплексный вариант (надо установить зависимости, скачать веса VGGish). Если нужен GUI-дашборд и визуализация локальных данных, в репозитории есть демка с Dash. В целом, проект показывает: предварительная обработка с VGGish + небольшой классификатор даёт рабочий детектор смеха ⁴ ⁵.
- **Другие библиотеки** – можно упомянуть *Speechmatics* API или другие коммерческие сервисы, но они не локальные. Среди открытых фреймворков: *pyAudioAnalysis* умеет выделять сегменты событий и строить классификаторы, однако требует самостоятельного обучения под ваши классы. Ближайший к «сделал из коробки» вариант – перечисленные выше проекты.

Простые эвристические методы («всплески» звука)

- **Порог по уровню громкости** – если нужны просто резкие звуки (всплески, шумы, хлопки), можно взять короткие фреймы аудио и находить резкие пики энергии. Например, вычислять амплитуду (или RMS) в скользящем окне и фиксировать моменты, когда она резко возрастает. Для этого можно использовать `scipy.signal.find_peaks` (по амплитуде) или функции из Librosa (например, `librosa.onset.onset_detect` для поиска начала звука) ⁹. Такой подход не требует ML и легко работает на CPU с любым форматом (после конвертации), но он просто находит *любые* громкие скачки, а не «смысловые» события. Тем не менее, может помочь обнаружить крики/шумы по превышению амплитуды.
- **Методы на основе частотных признаков** – более точные результаты даст анализ спектра: например, длинные резкие звуки или голоса (смех, крик) имеют характерные спектральные признаки (форманты, шумовые составляющие). Инструменты вроде OpenSMILE могут

извлекать сотни аудиофич, и затем можно обучить простую SVM/нейронку классифицировать «смех/не смех», «крик/не крик». Однако это уже сложнее в «коробочной» реализации. Если надо именно выделять любые всплески (без классификации), порог по энергии обычно достаточно.

Рекомендации и выводы

- **Поддержка форматов:** большинство инструментов любят WAV (16 кГц). Для MP3/Opus можно конвертировать через FFmpeg или библиотеки (`pydub`, `torchaudio`, `tensorflow_io` и др.). Например, `ffmpeg -i input.mp3 -ar 16000 output.wav`. Это просто предусмотреть в конвейере.
- **Приоритет смеха:** специализированные детекторы смеха (Gillick, ideo) покажут лучшие результаты на смехе, чем общие модели, потому что заточены на этот класс. Если нужно просто «есть ли смех» – можно сразу запустить сегментацию этих библиотек ⁷. YAMNet/PANNs тоже дадут метку «смех» с вероятностью, но иногда путают со схожими звуками.
- **Оценка результата:** у предобученных моделей возможны ложные срабатывания (например, YAMNet может принять громкий хохот за «бели-laughter» или не заметить тихий). Поэтому рекомендуется слушать/проверять примеры, при необходимости фильтровать по порогу вероятности.
- **Работа на CPU:** все перечисленные решения могут работать на процессоре. YAMNet небольшая; PANNs – несколько сотен МБ и дают классические вероятности; специальные библиотеки требуют зависимости, но без GPU работают (просто медленнее). Зато не надо собирать и обучать модели с нуля.

Источники: предобученные модели и их применение описаны в документации и статьях (TensorFlow/TensorFlow Hub о YAMNet ¹, GitHub PANNs ³). Готовые библиотеки для смеха (Gillick et al., ideo) – в их репозиториях на GitHub ⁷ ⁴. Эти решения открыты и могут быть использованы «из коробки» для анализа аудио со стримов.

¹ ² Transfer learning with YAMNet for environmental sound classification | TensorFlow Core
https://www.tensorflow.org/tutorials/audio/transfer_learning_audio

³ GitHub - qiuqiangkong/audioset_tagging_cnn
https://github.com/qiuqiangkong/audioset_tagging_cnn

⁴ ⁵ GitHub - ideo/LaughDetection
<https://github.com/ideo/LaughDetection>

⁶ FunAudioLLM/SenseVoiceSmall · Hugging Face
<https://huggingface.co/FunAudioLLM/SenseVoiceSmall>

⁷ ⁸ GitHub - jrgillick/laughter-detection
<https://github.com/jrgillick/laughter-detection>

⁹ librosa.onset.onset_detect — librosa 0.11.0 documentation
https://librosa.org/doc/main/generated/librosa.onset.onset_detect.html