openai / **whisper**

Code | Pull requests 93 | Discussions | Actions | Security | Insights

Your recovery codes have not been saved in the past year. Make sure you still have them stored somewhere safe by viewing and downloading them again.

View recovery codes

Robust Speech Recognition via Large-Scale Weak Supervision

MIT license

88.1k stars **10.9k** forks **673** watching Branches Activity Custom properties Tags

Public repository

14 Branches **13** Tags Go to file t Go to file Add file Code

jongwook Release 20250625 ✓ c0d2f62 · 3 months ago

.github	Release 20250625	3 months ago
data	fix typo data/README.md (#2433)	10 months ago
notebooks	Fix: GitHub display errors for Jupyter …	3 months ago
tests	large-v3 (#1761)	2 years ago
whisper	Release 20250625	3 months ago
.flake8	apply formatting with `black` (#1038)	2 years ago
.gitattributes	fix github language stats getting domi…	2 years ago
.gitignore	initial commit	3 years ago
.pre-commit-config.yaml	pre-commit: Upgrade black v25.1.0 a…	4 months ago
CHANGELOG.md	Release 20250625	3 months ago
LICENSE	initial commit	3 years ago
MANIFEST.in	Use tiktoken (#1044)	2 years ago
README.md	docs: updated README to specify tra…	3 months ago
approach.png	initial commit	3 years ago
language-breakdown.svg	large-v3 (#1761)	2 years ago
model-card.md	large-v3-turbo model (#2361)	last year
pyproject.toml	PEP 621: Migrate from setup.py to py…	8 months ago
requirements.txt	Relax triton requirements for compati…	last year

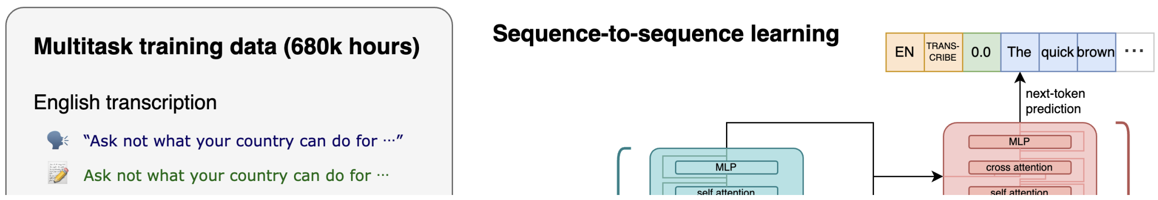
README MIT license

Whisper

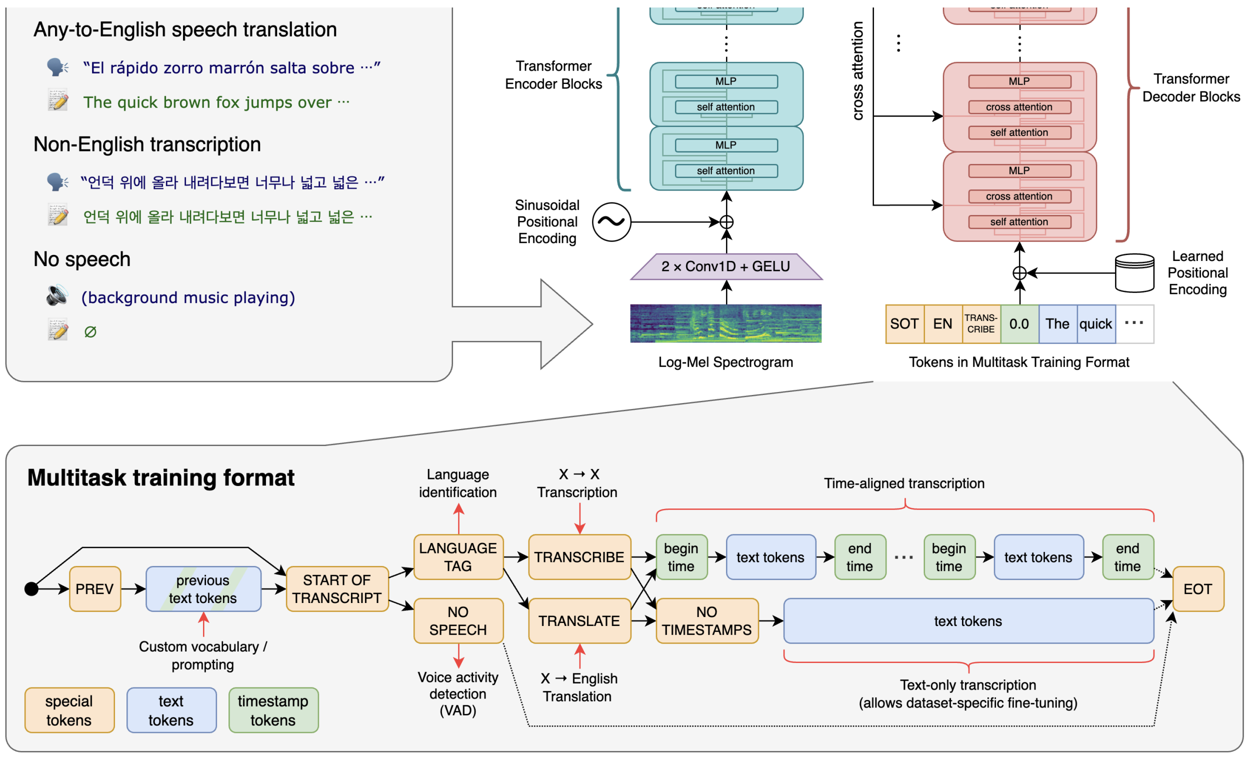
[Blog] [Paper] [Model card] [Colab example]

Whisper is a general-purpose speech recognition model. It is trained on a large dataset of diverse audio and is also a multitasking model that can perform multilingual speech recognition, speech translation, and language identification.

Approach



Multitask training data (680k hours)
English transcription
"Ask not what your country can do for ⋯"
Ask not what your country can do for ⋯
Sequence-to-sequence learning
EN
TRANS-CRIBE
0.0
The
quick
brown
⋯
next-token prediction
MLP
cross attention
self attention
MLP
self attention


Any-to-English speech translation
"El rápido zorro marrón salta sobre ···"
The quick brown fox jumps over ···
Non-English transcription
"언덕 위에 올라 내려다보면 너무나 넓고 넓은 ···"
언덕 위에 올라 내려다보면 너무나 넓고 넓은 ···
No speech
(background music playing)
∅
Transformer Encoder Blocks
MLP
self attention
Sinusoidal Positional Encoding
2 × Conv1D + GELU
Log-Mel Spectrogram
cross attention
Transformer Decoder Blocks
cross attention
Learned Positional Encoding
SOT
EN
TRANS-CRIBE
0.0
The
quick
Tokens in Multitask Training Format
Multitask training format
Language identification
X → X Transcription
Time-aligned transcription
PREV
previous text tokens
START OF TRANSCRIPT
LANGUAGE TAG
TRANSCRIBE
begin time
text tokens
end time
EOT
NO SPEECH
TRANSLATE
NO TIMESTAMPS
Custom vocabulary / prompting
Voice activity detection (VAD)
X → English Translation
Text-only transcription (allows dataset-specific fine-tuning)
special tokens
text tokens
timestamp tokens

A Transformer sequence-to-sequence model is trained on various speech processing tasks, including multilingual speech recognition, speech translation, spoken language identification, and voice activity detection. These tasks are jointly represented as a sequence of tokens to be predicted by the decoder, allowing a single model to replace many stages of a traditional speech-processing pipeline. The multitask training format uses a set of special tokens that serve as task specifiers or classification targets.

Setup

We used Python 3.9.9 and PyTorch 1.10.1 to train and test our models, but the codebase is expected to be compatible with Python 3.8-3.11 and recent PyTorch versions. The codebase also depends on a few Python packages, most notably OpenAI's tiktoken for their fast tokenizer implementation. You can download and install (or update to) the latest release of Whisper with the following command:

```
pip install -U openai-whisper
```

Alternatively, the following command will pull and install the latest commit from this repository, along with its Python dependencies:

```
pip install git+https://github.com/openai/whisper.git
```

To update the package to the latest version of this repository, please run:

```
pip install --upgrade --no-deps --force-reinstall git+https://github.com/openai/whisper.git
```

It also requires the command-line tool `ffmpeg` to be installed on your system, which is available from most package managers:

```
# on Ubuntu or Debian
sudo apt update && sudo apt install ffmpeg

# on Arch Linux
sudo pacman -S ffmpeg
```

```
# on MacOS using Homebrew (https://brew.sh/)
brew install ffmpeg

# on Windows using Chocolatey (https://chocolatey.org/)
choco install ffmpeg

# on Windows using Scoop (https://scoop.sh/)
scoop install ffmpeg
```

You may need `rust` installed as well, in case tiktoken does not provide a pre-built wheel for your platform. If you see installation errors during the `pip install` command above, please follow the Getting started page to install Rust development environment. Additionally, you may need to configure the `PATH` environment variable, e.g. `export PATH="$HOME/.cargo/bin:$PATH"`. If the installation fails with `No module named 'setuptools_rust'`, you need to install `setuptools_rust`, e.g. by running:

```
pip install setuptools-rust
```

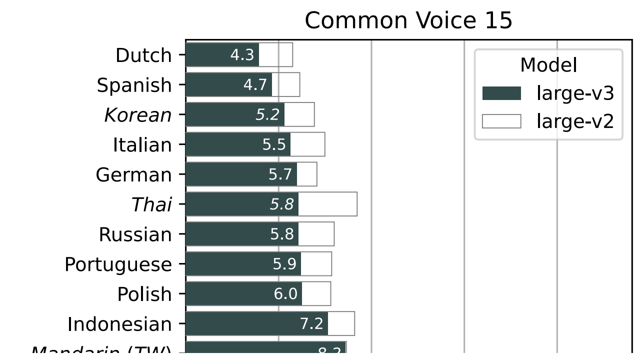
Available models and languages

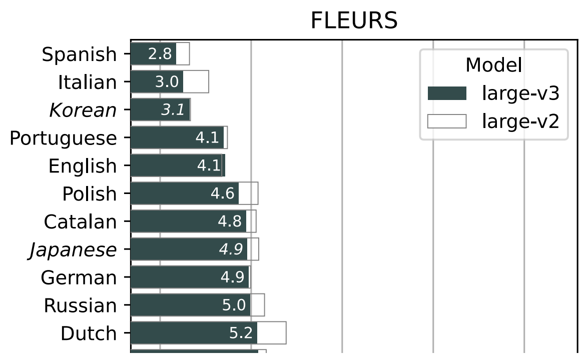
There are six model sizes, four with English-only versions, offering speed and accuracy tradeoffs. Below are the names of the available models and their approximate memory requirements and inference speed relative to the large model. The relative speeds below are measured by transcribing English speech on a A100, and the real-world speed may vary significantly depending on many factors including the language, the speaking speed, and the available hardware.

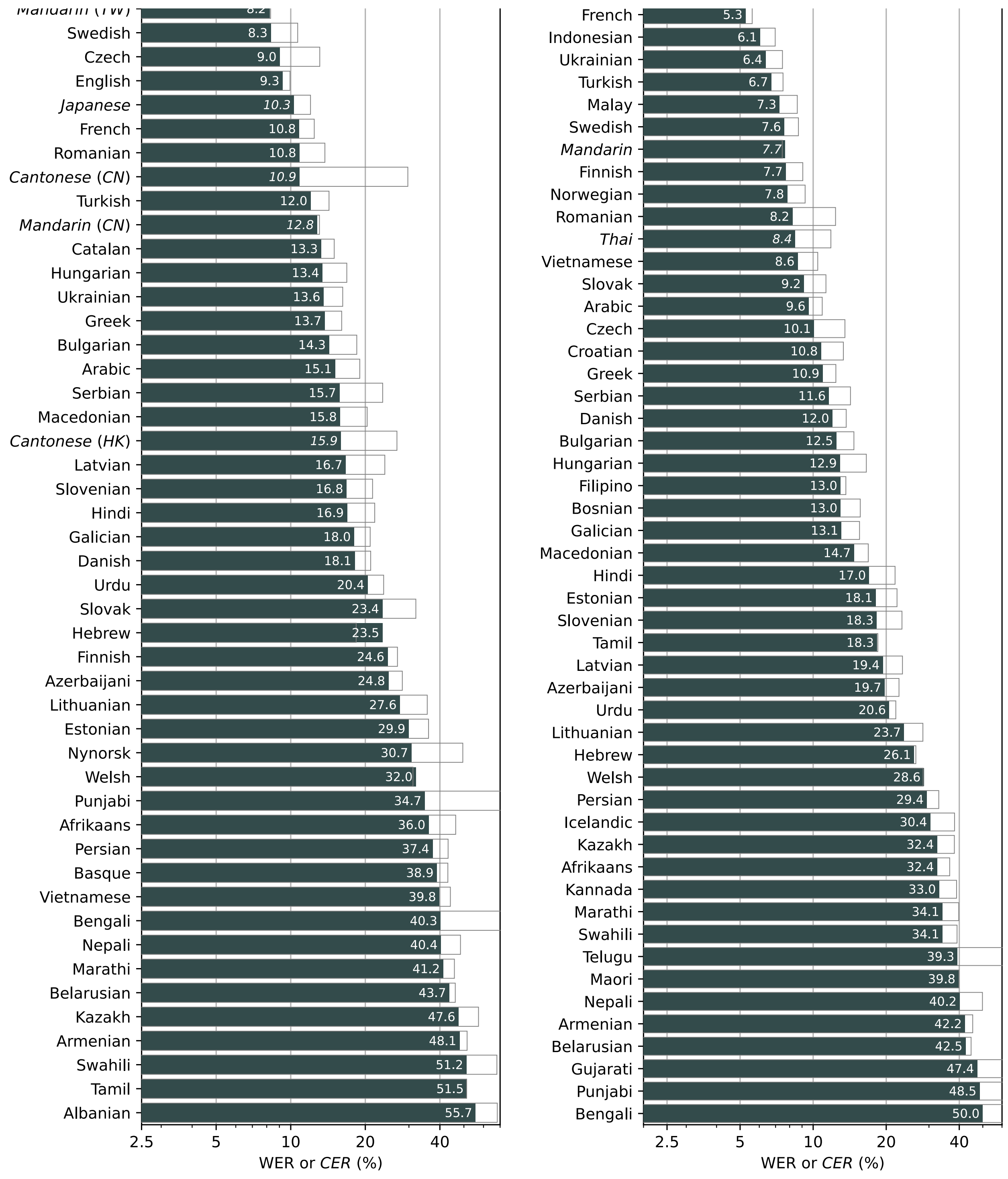
Size	Parameters	English-only model	Multilingual model	Required VRAM	Relative speed
tiny	39 M	`tiny.en`	`tiny`	~1 GB	~10x
base	74 M	`base.en`	`base`	~1 GB	~7x
small	244 M	`small.en`	`small`	~2 GB	~4x
medium	769 M	`medium.en`	`medium`	~5 GB	~2x
large	1550 M	N/A	`large`	~10 GB	1x
turbo	809 M	N/A	`turbo`	~6 GB	~8x

The `.en` models for English-only applications tend to perform better, especially for the `tiny.en` and `base.en` models. We observed that the difference becomes less significant for the `small.en` and `medium.en` models. Additionally, the `turbo` model is an optimized version of `large-v3` that offers faster transcription speed with a minimal degradation in accuracy.

Whisper's performance varies widely depending on the language. The figure below shows a performance breakdown of `large-v3` and `large-v2` models by language, using WERs (word error rates) or CER (character error rates, shown in *Italic*) evaluated on the Common Voice 15 and Fleurs datasets. Additional WER/CER metrics corresponding to the other models and datasets can be found in Appendix D.1, D.2, and D.4 of the paper, as well as the BLEU (Bilingual Evaluation Understudy) scores for translation in Appendix D.3.







Command-line usage

The following command will transcribe speech in audio files, using the `turbo` model:

```
whisper audio.flac audio.mp3 audio.wav --model turbo
```

The default setting (which selects the `turbo` model) works well for transcribing English. However, **the `turbo` model is not trained for translation tasks**. If you need to **translate non-English speech into English**, use one of the **multilingual models** (`tiny` , `base` , `small` , `medium` , `large`) instead of `turbo` .

For example, to transcribe an audio file containing non-English speech, you can specify the language:

```
whisper japanese.wav --language Japanese
```

To **translate** speech into English, use:

```
whisper japanese.wav --model medium --language Japanese --task translate
```

> **Note:** The `turbo` model will return the original language even if `--task translate` is specified. Use `medium` or `large` for the best translation results.

Run the following to view all available options:

```
whisper --help
```

See tokenizer.py for the list of all available languages.

Python usage

Transcription can also be performed within Python:

```
import whisper

model = whisper.load_model("turbo")
result = model.transcribe("audio.mp3")
print(result["text"])
```

Internally, the `transcribe()` method reads the entire file and processes the audio with a sliding 30-second window, performing autoregressive sequence-to-sequence predictions on each window.

Below is an example usage of `whisper.detect_language()` and `whisper.decode()` which provide lower-level access to the model.

```
import whisper

model = whisper.load_model("turbo")

# load audio and pad/trim it to fit 30 seconds
audio = whisper.load_audio("audio.mp3")
audio = whisper.pad_or_trim(audio)

# make log-Mel spectrogram and move to the same device as the model
mel = whisper.log_mel_spectrogram(audio, n_mels=model.dims.n_mels).to(model.device)

# detect the spoken language
_, probs = model.detect_language(mel)
print(f"Detected language: {max(probs, key=probs.get)}")

# decode the audio
options = whisper.DecodingOptions()
result = whisper.decode(model, mel, options)

# print the recognized text
print(result.text)
```

More examples

Please use the 🙌 Show and tell category in Discussions for sharing more example usages of Whisper and third-party extensions such as web demos, integrations with other tools, ports for different platforms, etc.

License

Releases 13

v20250625 Latest
on Jun 25

+ 12 releases

Contributors 81

+ 67 contributors

Languages

- ● **Python** 100.0%