Video Transcription

- **Primary Model: faster-whisper** with the large-v3 model.

  Whisper models are **large pre-trained models**. Training a model from scratch to understand human speech requires massive amounts of data and computing power. Using a pre-trained model like Whisper allows project to leverage its existing knowledge of dozens of languages and accents instantly, without any training needed. I'm here **fine-tuning the output**, not the model itself.

- faster-whisper is a highly optimized version of OpenAI's Whisper. It's much faster and uses less memory because it's built on CTranslate2. The large-v3 model I chosed because it is the most accurate for multilingual transcription and translation.

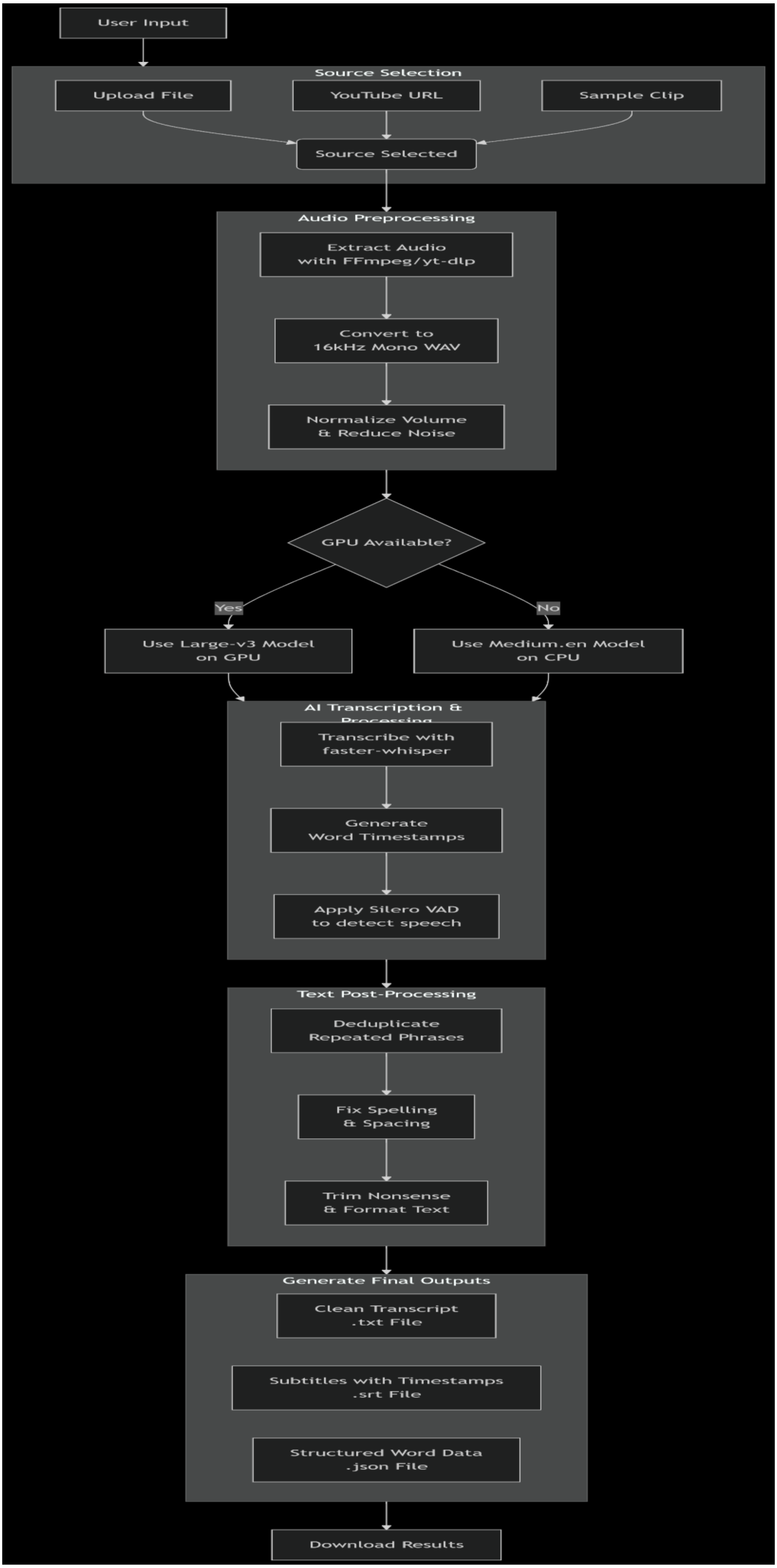
- On newer Python versions or if my GPU runs out of memory, it falls back to the standard openai-whisper library with a medium.en model, which is a good balance of speed and accuracy.

Key Algorithms & Techniques

- Audio Pre-processing (**FFmpeg, Pydub, noisereduce**): Used before sending audio to the model. It ***converts any audio/video file to a standardized 16kHz mono WAV format***, normalizes its volume, and reduces background noise to improve transcription accuracy.
- **Automatic Speech Recognition (ASR):** The core task performed by the Whisper model. It ***converts audio signals into text.***
- Voice Activity Detection (**VAD - Silero VAD**): Used during transcription to identify and only transcribe parts of the audio where someone is speaking, ***ignoring silent gaps and reducing errors***. This is a key feature of faster-whisper.
- Beam Search: Used during decoding (the model's text generation step). It keeps several likely options (beam_size=5) open at each step, leading to more accurate and coherent transcriptions than simpler greedy methods.
- Text Post-processing (Custom Algorithms): Used after transcription to clean the raw output. This includes:
- Jaccard Similarity: To detect and remove repetitive sentences.
- Rule-based Filters: To collapse repeated words and trim nonsensical endings (e.g., "आपको आपको आपको").
- Context-aware Corrections: To fix common informal shorthand (e.g., "u" -> "you").

The Main Steps (Pipeline)

1. **Input:** User provides a video/audio file or a YouTube URL.
2. **Audio Extraction:** FFmpeg extracts the audio track.
3. **Pre-processing:** Audio is converted to 16kHz mono, normalized, and cleaned of noise.
4. **Transcription:** The pre-processed audio is fed to the Whisper model, which uses VAD and Beam Search to generate raw text with word-level timestamps.
5. **Post-processing:** Custom algorithms clean the text, remove repetitions, and improve formatting.
6. **Output:** The final text is saved, along with an SRT file (for subtitles) and a JSON file (with precise word timestamps).


User Input
Source Selection
Upload File
YouTube URL
Sample Clip
Source Selected
Audio Preprocessing
Extract Audio
with FFmpeg/yt-dlp
Convert to
16kHz Mono WAV
Normalize Volume
& Reduce Noise
GPU Available?
Yes
No
Use Large-v3 Model
on GPU
Use Medium.en Model
on CPU
AI Transcription &
Processing
Transcribe with
faster-whisper
Generate
Word Timestamps
Apply Silero VAD
to detect speech
Text Post-Processing
Deduplicate
Repeated Phrases
Fix Spelling
& Spacing
Trim Nonsense
& Format Text
Generate Final Outputs
Clean Transcript
.txt File
Subtitles with Timestamps
.srt File
Structured Word Data
.json File
Download Results