Weekly Report 9

Google Chirp Streaming Speech Recognition

Timothé Berland

December 3, 2025

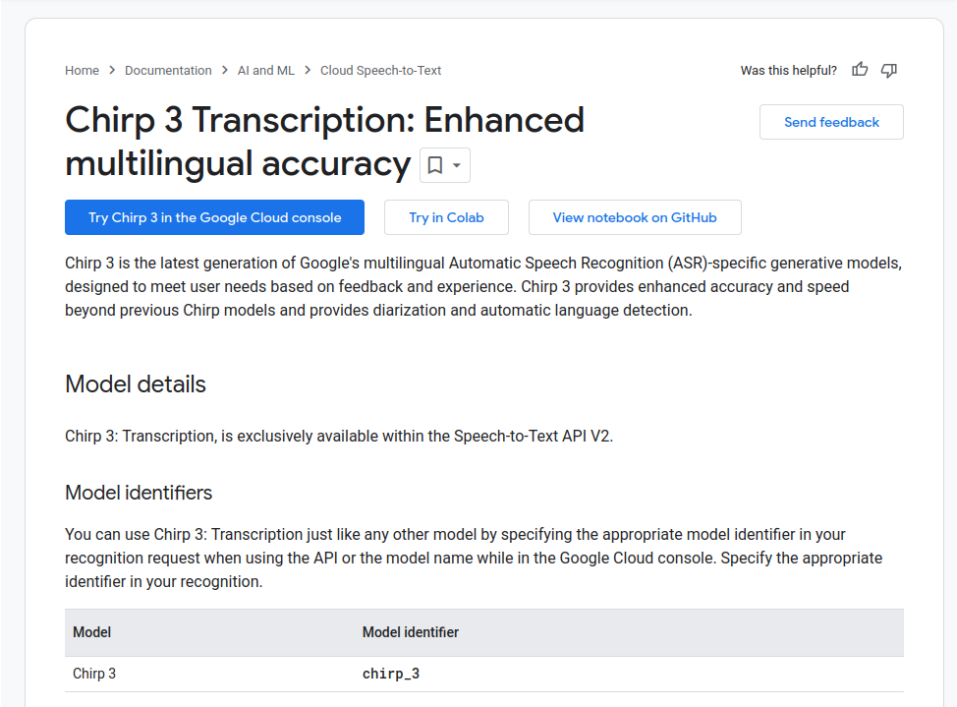
Introduction

- Study Google Cloud Speech-to-Text V2 streaming API
- Explore the **Chirp model** for real-time transcription
- Integrate streaming transcription into previous Whisper-based system

Summary

Chirp streaming modes

- Synchronous recognition
- Batch recognition
- **Streaming recognition**



Home › Documentation › AI and ML › Cloud Speech-to-Text

Was this helpful?

Chirp 3 Transcription: Enhanced multilingual accuracy

Send feedback

Try Chirp 3 in the Google Cloud console | Try in Colab | View notebook on GitHub

Chirp 3 is the latest generation of Google's multilingual Automatic Speech Recognition (ASR)-specific generative models, designed to meet user needs based on feedback and experience. Chirp 3 provides enhanced accuracy and speed beyond previous Chirp models and provides diarization and automatic language detection.

Model details

Chirp 3: Transcription, is exclusively available within the Speech-to-Text API V2.

Model identifiers

You can use Chirp 3: Transcription just like any other model by specifying the appropriate model identifier in your recognition request when using the API or the model name while in the Google Cloud console. Specify the appropriate identifier in your recognition.

Model	Model identifier
Chirp 3	`chirp_3`

- Streaming mode allows **real-time partial and final transcripts**

MINES Saint-Étienne | Institut Mines-Télécom

Summary

Audio capture workflow

- Continuous audio capture from microphone
- Audio split into **small chunks** sent to API
- Same structure as previous Whisper Streamlit implementation

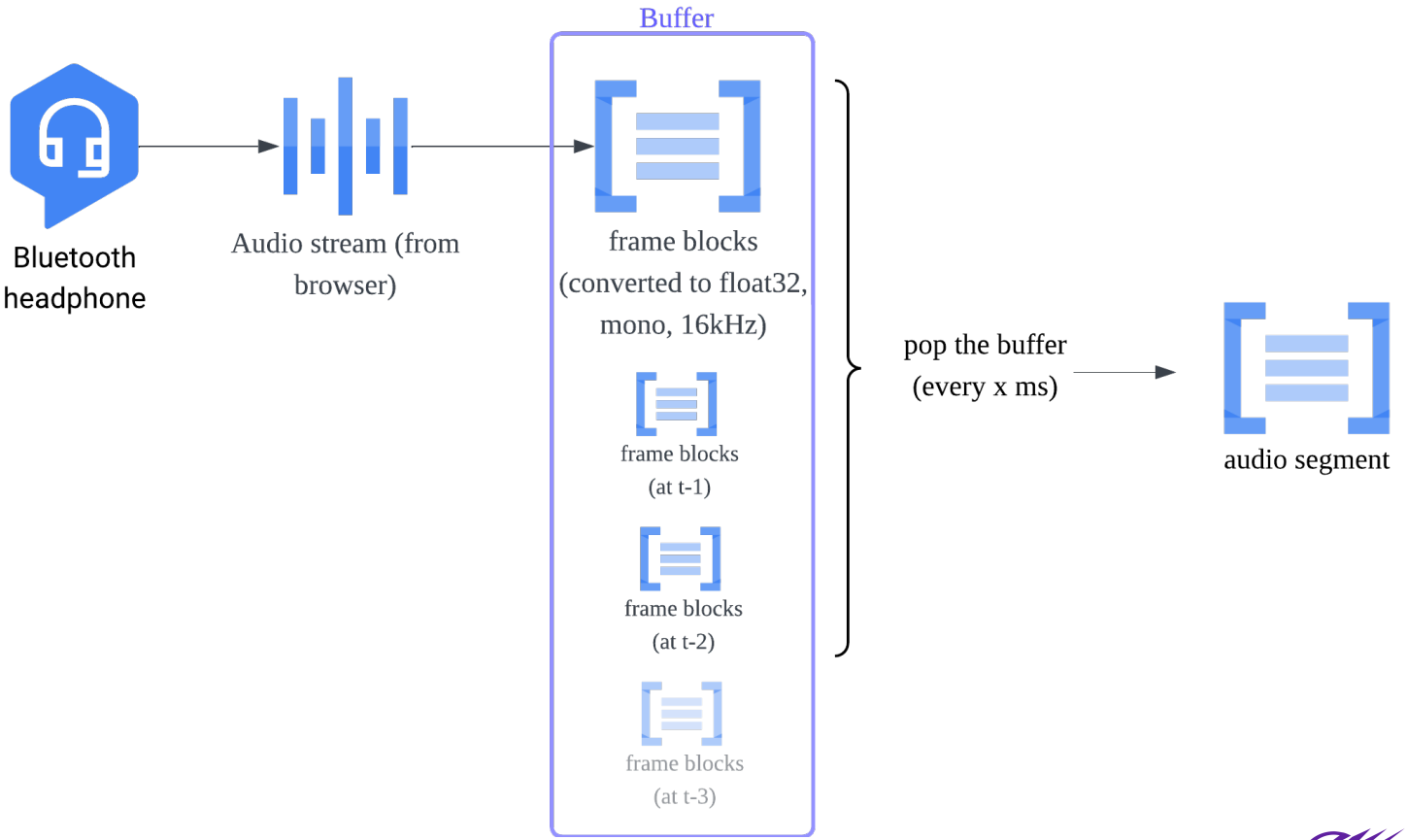


Figure: Real-time audio capture

Streaming request workflow

- First message: recognizer name + streaming configuration
- Following messages: only audio bytes
- Chunk size: $<$15 KB per request
- Model returns:
  - **Interim results** (can change)
  - **Final results** (stable)

```python
config_request = cloud_speech.StreamingRecognizeRequest(
    recognizer=f"projects/{PROJECT_ID}/locations/{REGION}/recognizers/_",
    streaming_config=streaming_config,
)

def request_generator():
    yield config_request
    yield from google_audio_generator(ctx)

responses = client.streaming_recognize(requests=request_generator())
```

Figure: Chirp streaming inference

Summary

Chirp features

- Punctuation, capitalisation, timestamps
- Language identification
- Stability score (0–1) for interim results
- Audio quality crucial: prefer clean PCM input

Conclusion & Next steps

Next steps

- Complete API implementation
- Optimize audio buffer
- Add optional translation (with a model like M2M100)

MINES Saint-Étienne | Institut Mines-Télécom