Clase 147 — Whisper: ASR, transcripción, traducción de audio

Parte: 2 — Deep Learning · Fuente: Radford et al. (2022) Whisper + OpenAI release notes.

Duración estimada: 70 min.

Clase 147 — Whisper: ASR, transcripción, traducción de audio

> *Parte: 2 — Deep Learning · Fuente: Radford et al. (2022) Whisper + OpenAI release notes. Duración estimada: 70 min.*

Objetivo

Usar Whisper (OpenAI 2022, open-source) — el modelo de ASR (Automatic Speech Recognition) multilenguaje que destronó a Google STT y AWS Transcribe en accuracy. Cubrir transcripción, traducción a inglés, timestamps, word-level timing. Alternativas modernas: Whisper-large-v3, distil-whisper (4× más rápido), insanely-fast-whisper.

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Cargar Whisper con transformers (openai/whisper-large-v3) o openai-whisper (lib oficial).
- Transcribir audio en cualquier idioma (99+ soportados).
- Aplicar task='translate' para traducir directo a inglés.
- Obtener timestamps a nivel palabra para subtítulos.
- Usar distil-whisper para inference 4-6× más rápida con calidad similar.

Temas

- Arquitectura: encoder-decoder Transformer + spectrogram input.
- Tamaños: tiny, base, small, medium, large-v3.
- Languages: detectado automático o explícito.
- Tareas: transcribe (en idioma origen), translate (→ inglés).
- Diarization (quién habla): no built-in, requiere pyannote.audio separado.
- Long-form audio: chunking con overlap.

Definiciones y características

- Whisper: encoder-decoder Transformer entrenado en 680k horas multi-lenguaje.
- pipeline('automatic-speech-recognition'): API HF más simple.
- Distil-Whisper: destillación 4× más rápida; calidad casi igual.
- WER (Word Error Rate): métrica estándar ASR.
- Timestamps: por segmento (default) o por word (con flag).

Dataset / recursos

- Cualquier audio: voz, podcasts, llamadas.
- HuggingFace: openai/whisper-large-v3, distil-whisper/distil-large-v3.
- Librerías: transformers, torch, librosa (procesamiento audio).

Ejercicios

1. Transcripción básica: cargar pipeline('asr', model='openai/whisper-base'); pasar un audio.
2. Multilenguaje: audio en español → transcribir; verificar.
3. Traducción: pipe(audio, task='translate') → texto en inglés.
4. Timestamps: pipe(audio, return_timestamps='word') → palabras con start/end seconds.
5. Distil-Whisper: comparar tiempo y WER vs full Whisper.

Homework verificable

Sistema de subtítulos:

1. Audio de 5-10 min (clase grabada, podcast).
2. Whisper-large-v3 con return_timestamps='word'.
3. Generar SRT (formato subtítulos) a partir del output.
4. Verificar manualmente la calidad en 1 minuto random.

Criterio de aceptación: SRT bien formado; timestamps razonables; ≥ 90 % de palabras correctas.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
Audio en formato no soportado	Whisper espera 16kHz mono. Fix: librosa.lo
Lento en CPU	Whisper-large es big. Fix: medium o distil
OOM en GPU	Fix: chunk_length_s=30 para procesar por c
Hallucinations en silencios	Whisper a veces "inventa" texto en silenci
Idioma detectado mal	Fix: language='es' explícito.

Preguntas frecuentes

Whisper vs Google STT / Azure?

Whisper es gratis y open. Calidad comparable o mejor en muchos idiomas. Google/Azure mejor en real-time streaming.

Distil-Whisper cuándo?

Cuando latencia importa o tenés CPU. Calidad ~1-2 WER points peor que full Whisper-large. Worth it.

Diarization (multiple speakers)?

Whisper NO la hace. Combinar con pyannote.audio.

Real-time streaming?

Whisper es batch (audio completo). Para streaming: faster-whisper con VAD, o insanely-fast-whisper (BetterTransformer + Flash Attention).

Hallucination prevention?

condition_on_previous_text=False, temperature alta cuando no hay confianza, VAD para skip silencios.

Referencias

- Radford et al. (2022), Robust Speech Recognition via Large-Scale Weak Supervision, OpenAI.
- Whisper repo.
- Distil-Whisper.
- Insanely Fast Whisper.

Siguiente clase

Clase 148 — LLMs aplicados: fine-tuning, prompting (+ LoRA / QLoRA, DPO, vLLM)

Apéndice: notebook (primer bloque)

Whisper (OpenAI 2022) = encoder-decoder transformer entrenado en 680k h de audio multilingüe. Pipeline: audio → mel-spectrogram → encoder → decoder autoregressive → texto. Fallback completo si whisper no está instalado.

```
USE_WH = False
try:
    import whisper
    USE_WH = True
    print('whisper disponible')
except Exception as e:
    print('whisper no disponible. Fallback: mel-spec desde scratch + API conceptual. Motivo:', type(e).__name__)

import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import stft
np.random.seed(42)
```

Archivos complementarios

- notebook.ipynb