IACADEMY

DE FUNDAMENTOS A ARQUITECTURA DE IA

MÓDULO 27

Voice AI y multimodal

Avanzado

iacedemy.com — 2026

MÓDULO 27

Voice AI y multimodal

Nivel: Avanzado

Autor: David Moya

Publicación: Mayo 2026

Plataforma: iacedemy.com

Text-to-Speech en 2026

Text-to-Speech (TTS) ha dado un salto brutal en los ultimos dos anos. Ya no estamos hablando de voces roboticas tipo GPS de 2010. Los modelos actuales generan audio con entonacion natural, pausas correctas y hasta emociones. La diferencia entre una voz sintetica premium y una voz humana es cada vez mas dificil de detectar.

El ecosistema TTS en 2026 se divide en tres categorias claras:

- **APIs cloud premium**: ElevenLabs, Google Cloud TTS, Amazon Polly. Calidad alta, coste por caracter.
- **Modelos open-source locales**: Kokoro, Voxtral 4B, Coqui TTS, Piper. Sin coste por uso, requieren GPU.
- **Herramientas gratuitas limitadas**: Edge TTS (via Microsoft), Google Translate TTS. Gratis, calidad media, ideal para prototipos.

La pregunta correcta no es "cual es la mejor voz" sino "cual es la mejor voz para mi caso de uso y presupuesto". Un prototipo interno no necesita ElevenLabs. Un producto B2C con miles de usuarios no puede depender de Edge TTS.

Edge TTS: el punto de entrada gratuito

Edge TTS usa las mismas voces de Microsoft Edge. Es gratuito, no requiere API key y tiene voces decentes en espanol. Perfecto para prototipos y proyectos internos.

```
# Instalar edge-tts
pip install edge-tts

# Uso basico desde linea de comandos
edge-tts --voice "es-ES-AlvaroNeural" --text "Hola, esto es una prueba de voz." --write-media

# Uso programatico en Python
import edge_tts
import asyncio

async def generar_audio(texto: str, voz: str = "es-ES-AlvaroNeural"):
    communicate = edge_tts.Communicate(texto, voz)
    await communicate.save("output.mp3")

asyncio.run(generar_audio("Este es un ejemplo de TTS gratuito."))
```

Voces recomendadas en espanol (Edge TTS)

- **es-ES-AlvaroNeural**: voz masculina espanola, tono claro
- **es-ES-ElviraNeural**: voz femenina espanola
- **es-MX-DaliaNeural**: voz femenina mexicana
- **es-MX-JorgeNeural**: voz masculina mexicana

Listar todas las voces: `edge-tts --list-voices | grep "es-"`

Speech-to-Text con Whisper

Whisper de OpenAI es el estandar de facto para Speech-to-Text (STT). El modelo large-v3 alcanza mas del 95% de precision en espanol, incluso con acentos regionales y ruido de fondo moderado. Es open-source, puedes ejecutarlo en local y no envias datos a ningun servidor externo.

Whisper estandar vs Faster Whisper

Whisper original funciona, pero es lento. Para produccion necesitas Faster Whisper, que usa CTranslate2 y es entre 4x y 8x mas rapido con la misma precision.

```
# Instalar faster-whisper
pip install faster-whisper

from faster_whisper import WhisperModel

# Cargar modelo (large-v3 para maxima precision)
model = WhisperModel("large-v3", device="cuda", compute_type="float16")

# Transcribir un archivo de audio
segments, info = model.transcribe("reunion.mp3", language="es")

print(f"Idioma detectado: {info.language} (probabilidad: {info.language_probability:.2f})")

for segment in segments:
    print(f"[{segment.start:.2f}s -> {segment.end:.2f}s] {segment.text}")
```

Para equipos sin GPU, el modelo `medium` funciona bien en CPU con Faster Whisper. Tarda mas, pero la precision en espanol sigue siendo superior al 90%.

Whisper en tiempo real

```
# Pipeline de transcripcion en tiempo real con chunks
import sounddevice as sd
import numpy as np
from faster_whisper import WhisperModel

model = WhisperModel("medium", device="cpu", compute_type="int8")

def transcribe_chunk(audio_data, sample_rate=16000):
    """Transcribe un chunk de audio."""
    segments, _ = model.transcribe(
        audio_data,
        language="es",
        vad_filter=True,  # Filtra silencios
        vad_parameters=dict(min_silence_duration_ms=500)
    )
    return " ".join([s.text for s in segments])

# Grabar y transcribir en chunks de 5 segundos
duration = 5  # segundos
sample_rate = 16000

print("Grabando... (Ctrl+C para parar)")
while True:
    audio = sd.rec(int(duration * sample_rate), samplerate=sample_rate, channels=1, dtype="flo
    sd.wait()
    text = transcribe_chunk(audio.flatten())
    if text.strip():
        print(f"> {text}")
```

Vision y analisis de imagenes

Los modelos multimodal combinan vision y lenguaje. Puedes enviar una imagen y preguntar sobre ella: que contiene, que texto tiene, que datos se ven en un grafico. Esto abre casos de uso que antes requerianOCR dedicado, pipelines de procesamiento de imagen y mucho codigo custom.

Qwen-VL para analisis visual

Qwen-VL es el modelo de vision mas potente que puedes ejecutar en local. Analiza imagenes con precision comparable a GPT-4V pero sin enviar datos fuera de tu infraestructura.

```
# Usando Qwen-VL via vLLM (self-hosted)
import requests

def analizar_imagen(image_url: str, pregunta: str) -> str:
    """Analiza una imagen con Qwen-VL via vLLM."""
    response = requests.post(
        "http://localhost:8000/v1/chat/completions",
        json={
            "model": "Qwen/Qwen2.5-VL-7B-Instruct",
            "messages": [{
                "role": "user",
                "content": [
                    {"type": "image_url", "image_url": {"url": image_url}},
                    {"type": "text", "text": pregunta}
                ]
            }],
            "max_tokens": 1024
        }
    )
    return response.json()["choices"][0]["message"]["content"]

# Ejemplos de uso
resultado = analizar_imagen(
    "file:///tmp/factura.png",
    "Extrae el numero de factura, fecha, importe total e IVA de esta factura."
)
print(resultado)
```

Casos de uso de vision multimodal

- **Extraccion de facturas**: analizar PDFs/imagenes y extraer campos estructurados
- **Analisis de dashboards**: enviar capturas de Grafana/Analytics y pedir resumen
- **Control de calidad**: detectar defectos en imagenes de producto
- **Accesibilidad**: generar descripciones alt-text automaticas para imagenes
- **Digitalizacion de documentos**: convertir documentos escaneados en texto estructurado

Pipeline de audio a video

El pipeline completo para generar video con IA desde texto tiene 5 etapas. Cada etapa usa una herramienta especializada:

1. **Texto**: guion escrito o generado por LLM
2. **Audio**: TTS convierte el guion en voz (ElevenLabs, Voxtral, Edge TTS)
3. **Avatar/Imagen**: imagen base del presentador (foto real o generada)
4. **Lip-sync**: Hedra o HyperFrames sincroniza labios del avatar con el audio

5. **Post-produccion**: FFmpeg anade intro, outro, subtitulos, musica de fondo

```
# Pipeline completo: texto -> audio -> video
import subprocess
import edge_tts
import asyncio
import requests

async def pipeline_video(guion: str, avatar_path: str, output_path: str):
    """Pipeline completo de texto a video."""

    # Paso 1: Generar audio con TTS
    audio_path = "/tmp/narration.mp3"
    communicate = edge_tts.Communicate(guion, "es-ES-AlvaroNeural")
    await communicate.save(audio_path)
    print(f"Audio generado: {audio_path}")

    # Paso 2: Lip-sync con Hedra API
    video_raw = "/tmp/video_raw.mp4"
    hedra_response = requests.post(
        "https://api.hedra.com/v1/characters",
        headers={"Authorization": "Bearer YOUR_API_KEY"},
        files={
            "audio": open(audio_path, "rb"),
            "image": open(avatar_path, "rb")
        }
    )
    video_url = hedra_response.json()["video_url"]
    # Descargar video generado
    with open(video_raw, "wb") as f:
        f.write(requests.get(video_url).content)

    # Paso 3: Post-produccion con FFmpeg
    subprocess.run([
        "ffmpeg", "-i", video_raw,
        "-i", "assets/intro.mp4",
        "-i", "assets/outro.mp4",
        "-filter_complex",
        "[1:v][0:v][2:v]concat=n=3:v=1:a=1[outv][outa]",
        "-map", "[outv]", "-map", "[outa]",
        output_path
    ], check=True)
    print(f"Video final: {output_path}")

asyncio.run(pipeline_video(
    guion="Bienvenidos al modulo 27 de IAcademy...",
    avatar_path="assets/presenter.jpg",
    output_path="output/m27_video.mp4"
))
```

Herramientas TTS: Edge TTS, ElevenLabs, Kokoro, Voxtral

Cada herramienta TTS tiene su nicho. No hay una "mejor" universal. Depende de tu caso de uso, presupuesto y requisitos de privacidad.

ElevenLabs: calidad premium

ElevenLabs es la referencia en calidad. Voces hiperrealistas, clonacion de voz, control de emociones. Precio: desde 5 EUR/mes (30 min) hasta 99 EUR/mes (uso intensivo). Ideal para productos B2C donde la calidad de voz es diferencial.

```
# ElevenLabs API
from elevenlabs import ElevenLabs

client = ElevenLabs(api_key="your-api-key")

audio = client.text_to_speech.convert(
    voice_id="pNInz6obpgDQGcFmaJgB",  # Adam
    text="Este es un ejemplo de voz premium con ElevenLabs.",
    model_id="eleven_multilingual_v2"
)

with open("elevenlabs_output.mp3", "wb") as f:
    for chunk in audio:
        f.write(chunk)
```

Kokoro: open-source, limitado en espanol

Kokoro es un modelo TTS open-source ligero. Funciona bien en ingles, pero su soporte de espanol es limitado. Util para prototipos en ingles o como base para fine-tuning.

Voxtral 4B: la mejor voz en espanol local

Voxtral 4B de Mistral es actualmente la mejor opcion para TTS en espanol ejecutado en local. La voz masculina en espanol (`es_male`) es clara, natural y sin artefactos. La voz femenina tiene ruido y acento brasileno, evitala para espanol.

> **Recomendacion practica para espanol**
>
> **Voxtral 4B es_male** es la voz preferida para espanol en local. Para produccion premium, usa ElevenLabs. Para prototipos rapidos, usa Edge TTS. Kokoro solo para prototipos en ingles.

Lip-sync con Hedra y HyperFrames

Lip-sync es la tecnologia que sincroniza los movimientos labiales de un avatar con un audio. Dos opciones principales:

Hedra

API cloud para lip-sync. Subes una imagen y un audio, y devuelve un video con el avatar "hablando". Calidad alta, coste por minuto de video generado. Ideal cuando necesitas calidad maxima y no te importa el coste.

HyperFrames (HeyGen open-source)

HyperFrames es un framework open-source de HeyGen que convierte HTML animado en MP4 usando GSAP, Puppeteer y FFmpeg. No es lip-sync puro, sino generacion de video con animaciones. Coste por render: 0 EUR. Apache 2.0.

```
# HyperFrames: HTML animado -> MP4
# 1. Crear template HTML con animaciones GSAP
# 2. Renderizar con Puppeteer
# 3. Combinar con FFmpeg

# Ejemplo de template HyperFrames
cat > template.html << 'HTMLEOF'
<html>
<body style="background:#0f0f18; color:white; font-family:Inter">
  <h1 id="title" style="opacity:0">Voice AI y Multimodal</h1>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/gsap/3.12.5/gsap.min.js"></script>
  <script>
    gsap.to("#title", {opacity:1, y:-20, duration:1, ease:"power2.out"});
  </script>
</body>
</html>
HTMLEOF

# Renderizar con Puppeteer + FFmpeg (via HyperFrames CLI)
npx hyperframes render template.html --output video.mp4 --duration 5
```

Casos de uso para PYMEs

Voice AI y multimodal no son solo para startups tech. Las PYMEs pueden aplicar estas tecnologias hoy con presupuestos minimos:

- **Soporte telefonico automatizado**: Whisper transcribe la llamada en tiempo real, un LLM genera respuestas, TTS las convierte en voz. Coste: ~0.02 EUR por minuto con modelos locales.

- **Resumenes de reuniones**: graba con cualquier herramienta, transcribe con Faster Whisper, resume con un LLM local. Sin enviar datos a terceros.
- **Videos de formacion**: genera guiones con LLM, audio con Voxtral, video con HyperFrames. Una PYME puede crear contenido de formacion interna sin equipo de video.
- **Accesibilidad web**: convierte articulos del blog en audio automaticamente con Edge TTS. Anade un boton "Escuchar articulo" sin coste.
- **Procesamiento de documentos**: Qwen-VL extrae datos de facturas, albaranes, formularios escaneados. Elimina la entrada manual de datos.

Costes y comparativa

Comparativa real de costes para generar 1 hora de audio TTS:

HERRAMIENTA	COSTE/HORA AUDIO	CALIDAD ESPANOL	PRIVACIDAD	GPU NECESARIA
Edge TTS	0 EUR	Media-Alta	Cloud (Microsoft)	No
ElevenLabs	~15-30 EUR	Muy alta	Cloud	No
Kokoro	0 EUR (local)	Baja	Local	Si (4GB+)
Voxtral 4B	0 EUR (local)	Alta (es_male)	Local	Si (8GB+)
Google Cloud TTS	~4-16 EUR	Alta	Cloud (Google)	No

Para STT, la comparativa es mas simple:

HERRAMIENTA	PRECISION ESPANOL	VELOCIDAD	COSTE
Whisper large-v3	95%+	Lento (sin GPU)	0 EUR (local)
Faster Whisper large-v3	95%+	4-8x mas rapido	0 EUR (local)
Faster Whisper medium	90%+	Rapido en CPU	0 EUR (local)
OpenAI Whisper API	95%+	Rapido	~0.36 EUR/hora

Ejercicio practico

Ejercicio M27: Pipeline de voz completo

1. **Instala edge-tts** y genera un audio de 30 segundos con una voz en espanol. Prueba al menos 3 voces diferentes y elige la que suene mas natural.
2. **Instala faster-whisper** y transcribe el audio que acabas de generar. Compara el texto original con la transcripcion. Calcula el porcentaje de precision.
3. **Crea un script Python** que:

```
# Pipeline: texto -> audio -> transcripcion -> verificacion
import edge_tts
import asyncio
from faster_whisper import WhisperModel

texto_original = "La inteligencia artificial multimodal combina texto, audio e imagen

# 1. Generar audio
# 2. Transcribir con Whisper
# 3. Comparar texto original vs transcripcion
# 4. Calcular precision (palabras correctas / total palabras)
```

4. **Anade vision**: toma una captura de pantalla de cualquier dashboard o factura. Usa la API de un modelo multimodal (Claude, GPT-4V o Qwen-VL local) para extraer informacion estructurada de la imagen.
5. **Documenta costes**: si tuvieras que procesar 100 horas de audio al mes, calcula el coste con cada herramienta TTS de la tabla anterior.

Bonus: Crea un boton "Escuchar articulo" para una pagina web usando Edge TTS. El audio se genera bajo demanda via una API FastAPI.

Conclusiones clave

Key takeaways del M27

1. TTS en 2026 tiene opciones para cada presupuesto: Edge TTS (gratis, prototipos), Voxtral 4B es_male (mejor espanol local), ElevenLabs (premium cloud).
2. Faster Whisper large-v3 es el estandar para STT en espanol: 95%+ precision, open-source, ejecutable en local sin enviar datos fuera.
3. Los modelos multimodal como Qwen-VL permiten analizar imagenes en local: facturas, dashboards, documentos escaneados.
4. El pipeline completo texto-audio-video es viable hoy: LLM genera guion, TTS genera voz, Hedra/HyperFrames genera video.
5. Para PYMEs, los casos de uso mas inmediatos son: resumenes de reuniones, videos de formacion internos, procesamiento de documentos y accesibilidad web.
6. Siempre evalua privacidad: modelos locales (Whisper, Voxtral, Qwen-VL) permiten procesar datos sensibles sin enviarlos a terceros.