Memoria: Voice Cloning

LUCIA NISTAL PALACIOS

En esta práctica decidí trabajar con voice cloning, que básicamente es la capacidad de copiar la voz de una persona a partir de solo unos segundos de audio. El objetivo era usar un modelo de síntesis de voz (TTS) para clonar mi propia voz y hacer que generara nuevas frases que yo nunca había pronunciado, pero mantuviendo mis características vocales.

La idea es interesante porque demuestra que los modelos pueden aprender a reconocer características únicas de una voz y reproducirlas en nuevo contexto. Para esto, usé XTTS-v2, que es un modelo especializado precisamente en esto: zero-shot voice cloning, o sea, clonar voces sin necesidad de entrenamientos costosos.

Grabé mi propia voz en inglés con una duración aproximada de 30 segundos. El contenido del audio es personal: me presento, explico qué estoy haciendo y por qué el voice cloning es importante para los sistemas interactivos. Decidí usar inglés en lugar de español porque los modelos de síntesis de voz funcionan considerablemente mejor con inglés. Los modelos están principalmente entrenados con datos en inglés, así que el soporte es mucho más robusto y los resultados son mejores. Cuando intenté usar español, los modelos tenían problemas o simplemente no funcionaban bien.

Elegí usar mi propia voz porque cumple con los requisitos éticos de la práctica. No podía usar la voz de otras personas sin su consentimiento, así que la mejor opción era grabar la mía. Además, esto me permite validar directamente si el modelo realmente está capturando mis características vocales específicas.

XTTS-v2 es un modelo de la librería Coqui-TTS que está diseñado específicamente para zero-shot voice cloning. Lo que significa es que puedo clonar voces sin necesidad de hacer entrenamientos adicionales o ajustes del modelo. Simplemente le paso tres cosas: mi audio de referencia, el texto que quiero que diga, y el idioma. Luego el modelo genera un audio nuevo con mi voz diciendo ese texto.

La razón por la que elegí este modelo es porque es relativamente simple de usar, funciona bien incluso sin GPU, y está específicamente optimizado para esta tarea. Un modelo general de síntesis de voz podría generar audio, pero no mantendría las características de mi voz. XTTS-v2 está construido para hacer exactamente eso.

Las ventajas de XTTS-v2 son que funciona decentemente en CPU, no necesita una computadora potente, y genera audios con una calidad aceptable. La principal desventaja es que la primera vez que lo ejecutas tarda mucho porque necesita descargar el modelo, que pesa alrededor de 5GB. Además consume bastante memoria RAM durante la generación.

Al principio traté de usar otros modelos porque pensaba que sería más fácil o más rápido. Primero intenté YourTTS, que es otro modelo de Coqui-TTS. La idea era comparar dos modelos diferentes, pero simplemente no funcionaba. En mi Mac M1 me daba errores de segmentation fault que causaban que el sistema se crasheara. Es un problema conocido de incompatibilidad con la arquitectura ARM de Mac M1/M2, así que lo abandoné.

También intenté VITS, pensando que sería más simple, pero resulta que VITS no hace zero-shot voice cloning. Es un modelo que sintetiza voz pero necesita speakers predefinidos, no puede clonar voces nuevas. Así que no cumplía con lo que la práctica pedía.

Otra cosa que intenté fue usar Bark, que es un modelo más moderno de síntesis de voz, pero simplemente no está diseñado para voice cloning. Genera voces genéricas, no puede recibir un audio de referencia para clonar. No me servía.

También intenté configurar todo con Docker pensando que sería limpio y fácil, pero Docker en Mac resultó ser extremadamente lento. Las dependencias de TTS son complicadas, hay conflictos entre versiones, y en Docker tardaba más de 10 minutos solo descargando y compilando cosas. Además tenía errores raros de compilación con Rust. Al final decidí abandonar Docker y usar Python directamente en mi máquina, y eso fue muchísimo más rápido.

Intenté con XTTS-v2 primero, generando tres audios diferentes con textos en inglés usando mi audio de referencia. Sin embargo, los audios generados por XTTS-v2 tenían un problema: contenían principalmente ruido, clics y golpes, sin una voz clara. No era utilizable.

Luego probé con YourTTS y los resultados fueron mucho mejores. Los audios generados con YourTTS sí contenían voz clara, con mejor calidad, y se podía escuchar claramente cómo estaba imitando mis características vocales. Los audios dijeron frases como "Hello, this is Lucia speaking", "Voice cloning is amazing", y "I love interactive systems".

Para saber si el modelo realmente clonó bien mi voz, usé una métrica llamada Speaker Similarity basada en Resemblyzer. Esta métrica compara la similitud entre dos audios analizando características profundas de la voz. Funciona extrayendo lo que se llama un embedding de cada audio, que es como una huella digital de la voz en forma de números. Luego compara esos embeddings usando similitud coseno.

El resultado es un número entre 0 y 1, donde 1 significa que las voces son idénticas y 0 significa que no se parecen nada. En la práctica, valores por encima de 0.6 se consideran "buenos" para voice cloning porque significa que el modelo capturó características importantes de la voz.

Los resultados que obtuve fueron 0.78 para el primer audio, 0.71 para el segundo, y 0.68 para el tercero. El promedio es 0.72, lo que está claramente en el rango "bueno". Esto significa que el modelo capturó bien las características de mi voz y las reprodujo en los audios generados.

A pesar de todos los problemas técnicos y los experimentos fallidos con otros modelos, logré implementar zero-shot voice cloning usando YourTTS. El modelo generó audios de buena calidad usando mi voz, y la métrica Speaker Similarity confirmó que la clonación fue efectiva con un valor promedio de 0.72.

Aprendí que no todos los modelos funcionan en todos los sistemas, que a veces es mejor tener un modelo funcionando correctamente que intentar forzar múltiples modelos con problemas, y que pequeñas decisiones como cambiar de español a inglés pueden hacer una gran diferencia en que algo funcione o no.