Google Cloud 语音转文字

我最近尝试了 Google Cloud 的语音转文字 API。以下是我用来执行转录的 Python 函数。

```python
import os
import json
import time
import argparse
from google.cloud import speech
from pydub import AudioSegment
import tempfile

# 固定的输出目录
OUTPUT_DIRECTORY = "assets/transcriptions"


def speech_to_text(audio_file, output_filename):
    print(f" 正在生成转录文件：{output_filename}")
    try:
        client = speech.SpeechClient()

        # 使用 pydub 加载音频文件以确定参数
        audio_segment = AudioSegment.from_file(audio_file)
        sample_rate = audio_segment.frame_rate
        channels = audio_segment.channels

        # 根据文件扩展名确定编码
        file_extension = os.path.splitext(audio_file)[1].lower()
        if file_extension == '.mp3':
            encoding = speech.RecognitionConfig.AudioEncoding.MP3
        elif file_extension in ['.wav', '.wave']:
            encoding = speech.RecognitionConfig.AudioEncoding.LINEAR16
        elif file_extension == '.flac':
            encoding = speech.RecognitionConfig.AudioEncoding.FLAC
        else:
            print(f" 不支持的文件格式：{file_extension}")
            return

        # 配置识别
        config = speech.RecognitionConfig(
            encoding=encoding,
            sample_rate_hertz=sample_rate,
            audio_channel_count=channels,
            language_code="en-US",  # 根据你的逻辑设置
        )

        with open(audio_file, "rb") as f:
            audio_content = f.read()

        audio = speech.RecognitionAudio(content=audio_content)

        # 执行长时间运行的语音识别
        try:
            operation = client.long_running_recognize(config=config, audio=audio)
            response = operation.result(timeout=300)  # 根据需要调整超时时间
        except Exception as e:
            print(f"转录过程中出错: {e}")
            return

        print(response.results)

        transcription = ""
        for result in response.results:
            transcription += result.alternatives[0].transcript + "\n"

        with open(output_filename, "w", encoding="utf-8") as f:
            f.write(transcription)
        print(f"转录已写入 {output_filename}")

    except Exception as e:
        print(f"生成转录文件 {output_filename} 时出错: {e}")


def process_audio_files(input_dir, output_dir):
    os.makedirs(output_dir, exist_ok=True)

    all_audio_files = [f for f in os.listdir(input_dir) if f.endswith(('.mp3', '.wav', '.m4a'))]
    total_files = len(all_audio_files)
    print(f"待处理的音频文件总数: {total_files}")
    if total_files == 0:
        print(f" 在 '{input_dir}' 目录中未找到音频文件。")
        return

    files_processed = 0



    for filename in all_audio_files:
        audio_file_path = os.path.join(input_dir, filename)
        output_filename = os.path.join(output_dir, f"{os.path.splitext(filename)[0]}.txt")
        if os.path.exists(output_filename):
            print(f" 跳过 {filename}: {output_filename} 已存在。")
            continue
        print(f"\n正在处理 {files_processed + 1}/{total_files}: {filename}")
        try:
            # 根据文件名后缀确定语言
            if filename.endswith('-zh.mp3') or filename.endswith('-zh.wav') or filename.endswith('-zh.m4a'):
                language_code = "cmn-CN"
            else:
                language_code = "en-US"

            # 如果需要，可以在 speech_to_text 中更新配置
            # 为简化起见，我们将在 speech_to_text 中设置 language_code

            speech_to_text(
                audio_file=audio_file_path,
                output_filename=output_filename,
            )
            files_processed += 1
            print(f" 文件 {files_processed}/{total_files} 已处理。\n")
        except Exception as e:
            print(f" 处理 {filename} 失败: {e}")
            continue

    print(f" 处理完成! 已处理 {files_processed}/{total_files} 个文件。")

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description=" 处理音频文件以生成转录。")
    parser.add_argument('--input_dir', type=str, default="assets/audios", help=" 音频文件的输入目录。")
```

```
args = parser.parse_args()


process_audio_files(
    input_dir=args.input_dir,
    output_dir=OUTPUT_DIRECTORY,
)
```