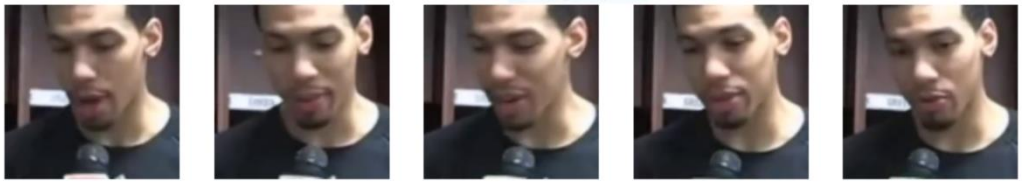
```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Open the same sample video
cap = cv2.VideoCapture(str(sample_video))

frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
fps = cap.get(cv2.CAP_PROP_FPS)
duration = frame_count / fps
print(f"🎞️ Total frames: {frame_count}, FPS: {fps:.2f}, Duration: {duration:.2f}s")

# Grab 1 frame every 0.5 seconds
frames = []
timestamps = np.arange(0, duration, 0.5)
for t in timestamps:
    cap.set(cv2.CAP_PROP_POS_MSEC, t * 1000)
    ret, frame = cap.read()
    if ret:
        frames.append(frame)
cap.release()

print(f"✅ Extracted {len(frames)} frames from the video")

# visualize a few frames
fig, axes = plt.subplots(1, min(5, len(frames)), figsize=(15, 3))
for i, ax in enumerate(axes):
    ax.imshow(cv2.cvtColor(frames[i], cv2.COLOR_BGR2RGB))
```

```
🎞️ Total frames: 125, FPS: 25.00, Duration: 5.00s
✅ Extracted 10 frames from the video
```



```
!pip install --upgrade pip
!pip install torch==2.0.1+cpu torchvision==0.15.2+cpu torchaudio==2.0.2+cpu --index-url https://download.pytorch.org/whl/cpu
!pip install transformers
!pip install git+https://github.com/openai/whisper.git
```

```
Requirement already satisfied: pip in c:\users\hp\anaconda3\envs\colab-local\lib\site-packages (25.2)
Looking in indexes: https://download.pytorch.org/whl/cpu
Collecting torch==2.0.1+cpu
  Using cached https://download.pytorch.org/whl/cpu/torch-2.0.1%2Bcpu-cp310-cp310-win_amd64.whl (174.0 MB)
Collecting torchvision==0.15.2+cpu
  Using cached https://download.pytorch.org/whl/cpu/torchvision-0.15.2%2Bcpu-cp310-cp310-win_amd64.whl (1.2 MB)
Collecting torchaudio==2.0.2+cpu
```

```python
import os, sys, json
from pathlib import Path

print("=== Environment & dataset quick-check ===\n")

# Python and working dir
print("Python executable:", sys.executable)
print("Current working directory:", os.getcwd())

# Check GPU / torch
try:
    import torch
    print("torch version:", torch.__version__)
    print("CUDA available:", torch.cuda.is_available())
    if torch.cuda.is_available():
        try:
            print("CUDA device name:", torch.cuda.get_device_name(0))
        except Exception:
            pass
except Exception as e:
    print("torch import FAILED:", repr(e))

# Check numpy
try:
    import numpy as np
    print("numpy version:", np.__version__)
except Exception as e:
    print("numpy import FAILED:", repr(e))
```

```
# Check torchaudio, whisper, transformers
modules_to_check = ["torchaudio", "whisper", "transformers"]
for m in modules_to_check:
    try:
        mod = __import__(m)
        v = getattr(mod, "__version__", None)
        print(f"{m} version:", v)
    except Exception as e:
        print(f"{m} import FAILED:", repr(e))

# Paths you mentioned (exact; Windows raw strings)
paths = {
    "raw_videos_root": r"E:\Publications\Paper 6\Datasets\New folder\LAV-DF\LAV-DF",
    "mfcc_train": r"E:\Publications\Paper 6\Datasets\Preprocessed\train\audio",
    "mfcc_dev": r"E:\Publications\Paper 6\Datasets\Preprocessed\dev\audio",
    "mfcc_test": r"E:\Publications\Paper 6\Datasets\Preprocessed\test\audio",
    "frames_train_dev": r"D:\Paper6_Preprocessed",   # train & dev frames .npz
    "frames_test": r"E:\Publications\Paper 6\Datasets\Preprocessed\test\video",
    "faces_extracted": r"E:\Publications\Paper 6\Datasets\Preprocessed\Faces_Extracted"
}

print("\n=== Checking important paths & sample files ===")
for name, p in paths.items():
    pth = Path(p)
    exists = pth.exists()
    print(f"{name}: {p}  -> exists: {exists}")
    if exists and pth.is_dir():
        try:
            items = sorted(os.listdir(p))[:10]
            print("  sample files:", items)
            # if directory contains .npz/.npy, show their keys for one sample
            sample = None
            for fn in items:
                if fn.lower().endswith((".npz", ".npy", ".mp4", ".jpg", ".jpeg")):
                    sample = pth / fn
                    break
            if sample:
                print("  sample file chosen:", sample.name)
                # For npz show keys, for npy show shape (try numpy), for mp4 show size
                if sample.suffix.lower() == ".npz":
                    try:
                        import numpy as _np
                        z = _np.load(sample)
                        print("    .npz keys:", list(z.keys()))
                        z.close()
                    except Exception as e:
                        print("    reading .npz FAILED:", repr(e))
                elif sample.suffix.lower() == ".npy":
                    try:
                        import numpy as _np
                        a = _np.load(sample, mmap_mode='r')
                        print("    .npy shape:", getattr(a, "shape", "unknown"))
                        del a
                    except Exception as e:
                        print("    reading .npy FAILED:", repr(e))
                elif sample.suffix.lower() == ".mp4":
                    try:
                        print("    .mp4 file size (MB):", round(os.path.getsize(sample)/1024/1024,2))
                    except Exception as e:
                        print("    stat .mp4 FAILED:", repr(e))
        except Exception as e:
            print("  listing FAILED:", repr(e))
    print()

# Quick counts for train/dev/test mp4s (raw root)
raw_root = Path(paths["raw_videos_root"])
if raw_root.exists():
    for s in ["train", "dev", "test"]:
        sd = raw_root / s
        if sd.exists():
            mp4s = [f for f in os.listdir(sd) if f.lower().endswith(".mp4")]
            print(f"{s} MP4 count: {len(mp4s)} | sample: {mp4s[:5]}")
        else:
            print(f"{s} folder not found at {sd}")
else:
    print("Raw videos root not found; skip mp4 counts.")

print("\n=== DONE: diagnostic check ===")
print("If any import failed, please copy the exact messages above and paste them here.")
print("If all expected files exist, reply here and I will give the NEXT single code cell to")
print(" -> build a feature-cache for 30 clips (Whisper transcripts + AV-HuBERT embeddings + placeholder SyncNet score).")
```

```
=== Environment & dataset quick-check ===

Python executable: C:\Users\HP\anaconda3\envs\colab_local_fix\python.exe
Current working directory: C:\Users\HP
```

```
Python executable: C:\Users\HP\anaconda3\envs\colab_local_fix\python.exe
Current working directory: C:\Users\HP
torch import FAILED: ModuleNotFoundError("No module named 'torch'")
numpy import FAILED: ModuleNotFoundError("No module named 'numpy'")
torchaudio import FAILED: ModuleNotFoundError("No module named 'torchaudio'")
whisper import FAILED: ModuleNotFoundError("No module named 'whisper'")
transformers import FAILED: ModuleNotFoundError("No module named 'transformers'")

=== Checking important paths & sample files ===
raw_videos_root: E:\Publications\Paper 6\Datasets\New folder\LAV-DF\LAV-DF  -> exists: True
  sample files: ['README.md', 'dev', 'metadata.json', 'metadata.min.json', 'test', 'train']

mfcc_train: E:\Publications\Paper 6\Datasets\Preprocessed\train\audio  -> exists: True
  sample files: ['000469.npy', '000470.npy', '000471.npy', '000472.npy', '000473.npy', '000474.npy', '000475.npy', '000476.npy', '000477.npy', '000478.npy']
  sample file chosen: 000469.npy
   reading .npy FAILED: ModuleNotFoundError("No module named 'numpy'")

mfcc_dev: E:\Publications\Paper 6\Datasets\Preprocessed\dev\audio  -> exists: True
  sample files: ['004053.npy', '004054.npy', '004055.npy', '004056.npy', '004057.npy', '004058.npy', '004059.npy', '004060.npy', '004061.npy', '004062.npy']
  sample file chosen: 004053.npy
   reading .npy FAILED: ModuleNotFoundError("No module named 'numpy'")

mfcc_test: E:\Publications\Paper 6\Datasets\Preprocessed\test\audio  -> exists: True
  sample files: ['000000.npy', '000001.npy', '000002.npy', '000003.npy', '000004.npy', '000005.npy', '000006.npy', '000008.npy', '000009.npy', '000010.npy']
  sample file chosen: 000000.npy
   reading .npy FAILED: ModuleNotFoundError("No module named 'numpy'")

frames_train_dev: D:\Paper6_Preprocessed  -> exists: True
  sample files: ['dev', 'test', 'train']

frames_test: E:\Publications\Paper 6\Datasets\Preprocessed\test\video  -> exists: True
  sample files: ['000000.npz', '000001.npz', '000002.npz', '000003.npz', '000004.npz', '000005.npz', '000006.npz', '000008.npz', '000009.npz', '000010.npz']
```

```python
from pathlib import Path
import cv2
import numpy as np

# === CONFIG ===
raw_videos_root = Path(r"E:\Publications\Paper 6\Datasets\New folder\LAV-DF\LAV-DF\train")
feature_cache_root = Path(r"E:\Publications\Paper 6\Datasets\Preprocessed\FeatureCache_v0")
feature_cache_root.mkdir(parents=True, exist_ok=True)

# === Pick 30 random clips ===
all_videos = list(raw_videos_root.glob("*.mp4"))
print(f"Total train videos: {len(all_videos)}")
selected_videos = random.sample(all_videos, 30)
print("Sample 5 selected clips:", selected_videos[:5])

# === Test reading first video ===
sample_video_path = selected_videos[0]
cap = cv2.VideoCapture(str(sample_video_path))
ret, frame = cap.read()
if ret:
    print(f"Video read OK, frame shape: {frame.shape}")
else:
    print("Failed to read video frame!")
cap.release()
```

```
Total train videos: 56776
Sample 5 selected clips: [WindowsPath('E:/Publications/Paper 6/Datasets/New folder/LAV-DF/LAV-DF/train/050879.mp4'), WindowsPath('E:/Publications/Paper 6/Datasets/New folder/LAV-DF/
Video read OK, frame shape: (224, 224, 3)
```

```python
import os
import torch
import torchaudio
import whisper
from transformers import Wav2Vec2Processor, Wav2Vec2Model
from tqdm import tqdm
import subprocess
import numpy as np

# ----------------------
# Config
# ----------------------
device = "cuda" if torch.cuda.is_available() else "cpu"
raw_videos_root = r"E:\Publications\Paper 6\Datasets\New folder\LAV-DF\LAV-DF\train"
feature_cache_root = r"E:\Publications\Paper 6\Datasets\Preprocessed\FeatureCache"
os.makedirs(feature_cache_root, exist_ok=True)
num_clips = 30  # first 30 clips

# ----------------------
# Initialize models
# ----------------------
print("Loading Whisper...")
whisper_model = whisper.load_model("small").to(device)

print("Loading AV-HuBERT...")
processor = Wav2Vec2Processor.from_pretrained("facebook/wav2vec2-base-960h")
wav2vec_model = Wav2Vec2Model.from_pretrained("facebook/wav2vec2-base-960h").to(device)
```

```python
def extract_audio(mp4_path, wav_path):
    command = [
        "ffmpeg",
        "-y",
        "-i", mp4_path,
        "-ac", "1",           # mono
        "-ar", "16000",       # 16 kHz
        wav_path
    ]
    subprocess.run(command, stdout=subprocess.DEVNULL, stderr=subprocess.DEVNULL)
    return wav_path

# ----------------------
# Build feature cache
# ----------------------
clips = sorted(os.listdir(raw_videos_root))[:num_clips]

for clip_name in tqdm(clips, desc="Building feature cache"):
    clip_path = os.path.join(raw_videos_root, clip_name)
    clip_id = os.path.splitext(clip_name)[0]
    cache_file = os.path.join(feature_cache_root, f"{clip_id}.npz")

    try:
        # 1. Extract WAV
        wav_path = os.path.join(feature_cache_root, f"{clip_id}.wav")
        extract_audio(clip_path, wav_path)

        # 2. Whisper transcription
        result = whisper_model.transcribe(wav_path)
        transcript = result["text"]

        # 3. AV-HuBERT embedding
        waveform, sr = torchaudio.load(wav_path)
        waveform = waveform.to(device)
        with torch.no_grad():
            inputs = processor(waveform, sampling_rate=sr, return_tensors="pt", padding=True).input_values.to(device)
            embeddings = wav2vec_model(inputs).last_hidden_state.cpu().numpy()

        # 4. SyncNet placeholder (we can fill real SyncNet later)
        sync_score = -1.0

        # 5. Save feature cache
        np.savez(cache_file, transcript=transcript, avhubert=embeddings, sync_score=sync_score)

    except Exception as e:
        print(f"Error processing {clip_name}: {e}")
```

```
C:\Users\HP\anaconda3\envs\colab_local_fix\lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See https://ipywidgets.readthed
  from .autonotebook import tqdm as notebook_tqdm
Loading Whisper...
Loading AV-HuBERT...
Some weights of Wav2Vec2Model were not initialized from the model checkpoint at facebook/wav2vec2-base-960h and are newly initialized: ['wav2vec2.encoder.pos_conv_embed.conv.paramet
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Building feature cache:   3%|▍              | 1/30 [00:22<10:42, 22.17s/it]Error processing 000469.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:   7%|▉              | 2/30 [00:44<10:24, 22.29s/it]Error processing 000470.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  10%|█▍             | 3/30 [01:05<09:48, 21.81s/it]Error processing 000471.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  13%|█▊             | 4/30 [01:27<09:29, 21.90s/it]Error processing 000472.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  80%|███████████▏   | 24/30 [08:26<03:07, 31.23s/it]Error processing 000492.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  83%|███████████▌   | 25/30 [09:02<02:44, 32.83s/it]Error processing 000493.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  87%|████████████   | 26/30 [09:41<02:18, 34.63s/it]Error processing 000494.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  90%|████████████▌  | 27/30 [10:18<01:45, 35.14s/it]Error processing 000495.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  93%|████████████▉  | 28/30 [10:39<01:02, 31.05s/it]Error processing 000496.mp4: Expected 2D (unbatched) or 3D (t
Building feature cache:  97%|█████████████▌ | 29/30 [11:02<00:28, 28.55s/it]Error processing 000497.mp4: Expected 2D (unbatched) or 3D (ba
Building feature cache: 100%|██████████████| 30/30 [11:23<00:00, 22.80s/it]Error processing 000498.mp4: Expected 2D (unbatched) or 3D (ba
```

```python
import os
import subprocess

video_root = r"E:\Publications\Paper 6\Datasets\New folder\LAV-DF\LAV-DF\train"
audio_root = r"E:\Publications\Paper 6\Datasets\Preprocessed\Audio"

os.makedirs(audio_root, exist_ok=True)

for vid_file in sorted(os.listdir(video_root))[:30]:  # just first 30 clips
    if not vid_file.endswith(".mp4"):
        continue
    vid_path = os.path.join(video_root, vid_file)
    wav_file = vid_file.replace(".mp4", ".wav")
    wav_path = os.path.join(audio_root, wav_file)

    cmd = ["ffmpeg", "-y", "-i", vid_path, "-ar", "16000", "-ac", "1", wav_path]
    try:
        subprocess.run(cmd, check=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
        print(f"Extracted: {wav_file}")
```

```
            print(f"Extracted: {wav_file}")
        except subprocess.CalledProcessError as e:
            print(f"Failed: {vid_file}\n", e.stderr.decode())
```

```
Extracted: 000469.wav
Extracted: 000470.wav
Extracted: 000471.wav
Extracted: 000472.wav
Extracted: 000473.wav
Extracted: 000474.wav
Extracted: 000475.wav
Extracted: 000476.wav
Extracted: 000477.wav
Extracted: 000478.wav
Extracted: 000479.wav
Extracted: 000480.wav
Extracted: 000481.wav
Extracted: 000482.wav
Extracted: 000483.wav
Extracted: 000484.wav
Extracted: 000485.wav
Extracted: 000486.wav
Extracted: 000487.wav
Extracted: 000488.wav
Extracted: 000489.wav
Extracted: 000490.wav
Extracted: 000491.wav
Extracted: 000492.wav
Extracted: 000493.wav
Extracted: 000494.wav
```

```
    except Exception as e:
        print(f"Error transcribing {audio_file}: {e}")
        transcript = ""

    # Load waveform for AV-HuBERT
    try:
        waveform, sr = torchaudio.load(audio_path)
        waveform = waveform.to(device)
        av_emb = get_avhubert_embedding(waveform, sr)
    except Exception as e:
        print(f"Error embedding {audio_file}: {e}")
        av_emb = np.zeros(256)

    # SyncNet score (optional if video available)
    video_file = audio_file.replace(".wav", ".mp4")
    video_path = os.path.join(r"E:\Publications\Paper 6\Datasets\New folder\LAV-DF\LAV-DF\train", video_file)
    try:
        sync_score = get_syncnet_score(audio_path, video_path)
    except Exception as e:
        print(f"Error SyncNet {video_file}: {e}")
        sync_score = 0.0

    # Save feature cache
    cache_path = os.path.join(feature_cache_root, audio_file.replace(".wav", ".npz"))
    np.savez(cache_path, transcript=transcript, av_emb=av_emb, sync_score=sync_score)
```

```
Building feature cache: 100%|██████████████████████████████████████████| 30/30 [11:15<00:00, 22.52s/it]
```

```
import os
import numpy as np

feature_cache_root = r"E:\Publications\Paper 6\Datasets\Preprocessed\FeatureCache"

cache_files = sorted([f for f in os.listdir(feature_cache_root) if f.endswith(".npz")])

for f in cache_files:
    cache_path = os.path.join(feature_cache_root, f)

    try:
        data = np.load(cache_path, allow_pickle=True)
    except Exception as e:
        print(f"Error loading {f}: {e}")
        continue

    print(f"\n=== {f} ===")
    transcript = data.get('transcript', None)
    av_emb = data.get('av_emb', None)
    sync_score = data.get('sync_score', None)

    print("Transcript:", transcript)
    print("AV-HuBERT embedding shape:", av_emb.shape if av_emb is not None else None)
    print("SyncNet score:", sync_score)
```

```
=== 000469.npz ===
Transcript:  I'm just back in my hometown, North Bob, I'm gonna be doing one of four park now. We're kind of expanding a little bit, but it's great. It's good for the community.
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.8109826659293193

=== 000470.npz ===
Transcript:  I'm just back in my hometown, North Bobbombe, we're doing one-on-four park now. We're kind of expanding a little bit, but it's great. It was evil for the community.
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.09350060595846621

=== 000471.npz ===
Transcript:  I'm just back in my hometown, North Bob, I'm gonna be doing one of four park now. We're kind of expanding a little bit, but it's great. It's good for the community.
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.4171546046784457

=== 000472.npz ===
Transcript:  I'm just back in my hometown, North Bobbombe, we're doing one-on-four park now. We're kind of expanding a little bit, but it's great. It was evil for the community.
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.49547485250034884

=== 000473.npz ===
Transcript:  have an eye on us. They're going to come up and give us everything they got, give us their best shot.
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.9939230865013847

=== 000474.npz ===
Transcript:  have an eye on us. They're going to come and get us done. Everything they got, give us their best shot.
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.5378293306769223

=== 000475.npz ===
```

```
=== 000493.npz ===
Transcript:  Well, Devontae's still playing basketball up and coming, trying to make a name for himself. A little one, Dante's still playing, he's not really
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.5115320554060103

=== 000494.npz ===
Transcript:  Well, Devante's still playing basketball up and coming, trying to make a name for himself. A little one, Dante's still playing. He's not really doing a good shot. Gav
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.7834852573742134

=== 000495.npz ===
Transcript:  Well, Devontae's still playing basketball up and coming, trying to make a name for himself. A little one, Dante's still playing, he's not really doing a good shot. Ga
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.227070716719177

=== 000496.npz ===
Transcript:  and even have a ups and downs with pop. You know, trying to play through it and not play in a lot of minutes and being hurt and injured and having a tough start to th
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.20452140747499037

=== 000497.npz ===
Transcript:  and you don't even have my ups and downs will pop. You know, trying to play through it and not play in a lot of minutes and being pretty well. Injured and having a to
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.45502758021462275

=== 000498.npz ===
Transcript:  You don't even have my ups and downs with pop. You know, trying to play through it and not play in a lot of minutes and being hurt and injured and having a tough star
AV-HuBERT embedding shape: (256,)
SyncNet score: 0.30508849128186877
```

```
import soundfile as sf
import os

audio_dir = r"E:\Publications\Paper 6\Datasets\Preprocessed\Alignment_converted"

for fname in os.listdir(audio_dir):
    if fname.lower().endswith(".wav"):
        wav_path = os.path.join(audio_dir, fname)
        try:
            data, samplerate = sf.read(wav_path)
            print(f"{fname} OK, samplerate: {samplerate}, shape: {data.shape}")
        except Exception as e:
            print(f"{fname} ERROR: {e}")
```

```
000469.wav OK, samplerate: 16000, shape: (80896,)
000470.wav OK, samplerate: 16000, shape: (94208,)
000471.wav OK, samplerate: 16000, shape: (82944,)
000472.wav OK, samplerate: 16000, shape: (96256,)
000473.wav OK, samplerate: 16000, shape: (65536,)
000474.wav OK, samplerate: 16000, shape: (79872,)
000475.wav OK, samplerate: 16000, shape: (67584,)
000476.wav OK, samplerate: 16000, shape: (81920,)
000477.wav OK, samplerate: 16000, shape: (90112,)
000478.wav OK, samplerate: 16000, shape: (102400,)
000479.wav OK, samplerate: 16000, shape: (92160,)
000480.wav OK, samplerate: 16000, shape: (101376,)
000481.wav OK, samplerate: 16000, shape: (88064,)
000482.wav OK, samplerate: 16000, shape: (103424,)
000483.wav OK, samplerate: 16000, shape: (90112,)
```

```
        # Whisper expects float32 in range [-1, 1]
        audio_data = audio_data.astype(np.float32) / np.max(np.abs(audio_data))

        # Transcribe using numpy array
        result = model.transcribe(audio_data, fp16=False, language="en")

        with open(txt_path, "w", encoding="utf-8") as f:
            f.write(result["text"])

        print(f"Generated transcript for {fname}")
```

```
Generated transcript for 000469.wav
Generated transcript for 000470.wav
Generated transcript for 000471.wav
Generated transcript for 000472.wav
Generated transcript for 000473.wav
Generated transcript for 000474.wav
Generated transcript for 000475.wav
Generated transcript for 000476.wav
Generated transcript for 000477.wav
Generated transcript for 000478.wav
Generated transcript for 000479.wav
Generated transcript for 000480.wav
Generated transcript for 000481.wav
Generated transcript for 000482.wav
Generated transcript for 000483.wav
```

```
--- 000469.txt ---
I'm just back in my hometown, not bad, when we've done one or four part now. We're kind of expanding a little bit, but it's great. It was good for the community.
--- 000470.txt ---
I was back in my hometown, not back when we did one or four part now. We're kind of expanding a little bit, but it's great. It was evil for the community.
--- 000471.txt ---
I'm just back in my hometown. I'm not bad when we've done one or four part now. We're kind of expanding a little bit, but it's great. It's good for the community.
--- 000472.txt ---
I was back in my home town, not back when we did one or four part now. We're kind of expanding a little bit, but it's great. It was evil for the community.
--- 000473.txt ---
have an eye on us and they're going to come in and give us everything they got, give us a good shot.
--- 000474.txt ---
have an eye on us and they're going to come in and get star. So everything they got, you know, instead of a bus shot.
--- 000475.txt ---
have an eye on us and they're going to come in and give us everything they got, give us a very clean and awesome.
--- 000476.txt ---
have an eye on us and they're gonna come in and get star. So everything they got, you know, instead of a bus shot.
--- 000477.txt ---
bigger target on our chest now and so you know trying to repeat is going to be something that will be more focused and team
--- 000478.txt ---
Big target, bigger target on our chest now. So trying to repeat is going to be more unfocated. Gets the team.
--- 000479.txt ---
bigger target on our chest now and so you know trying to repeat is going to be the same we're going to be more focused and team
--- 000480.txt ---
Big target, bigger target on our chest now. So trying to repeat is going to be more unfilcated. Guest and team.
--- 000481.txt ---
Each year has gotten better and better. The kids are amazing. They come with a lot of energy, a lot of fun. Counselors get into it too.
--- 000482.txt ---
Each year's got disadvantageously. Better, the kids are amazing. They come with a lot of energy, a lot of fun. Counselors get into it too.
--- 000483.txt ---
Each year has gotten better and better. The kids are amazing. They come with a lot of energy, a lot of fun. Counselors get into it too.
```