STORYFORGE

SUMMER OF INNOVATION

OUR TEAM

AI AVENGERS

- MAHESH KRISHNAM (MEMBER 1)
- ANMOL SHARMA (MEMBER 2)

- ADARSH GUPTA (MEMBER 3)
- AMAN PUSHKAR (MEMBER 4)

STORY GENERATION DATASET

STORY GENERATION DATASET

The GPT-2 model was fine-tuned using a large corpus of 100,000 stories and narratives. The dataset includes various genres and writing styles to ensure the model can generate diverse and engaging stories.

Resource :
https://www.kaggle.com/datasets/cuddlefish/fairy-tales

SPEECH SYNTHESIS DATASET

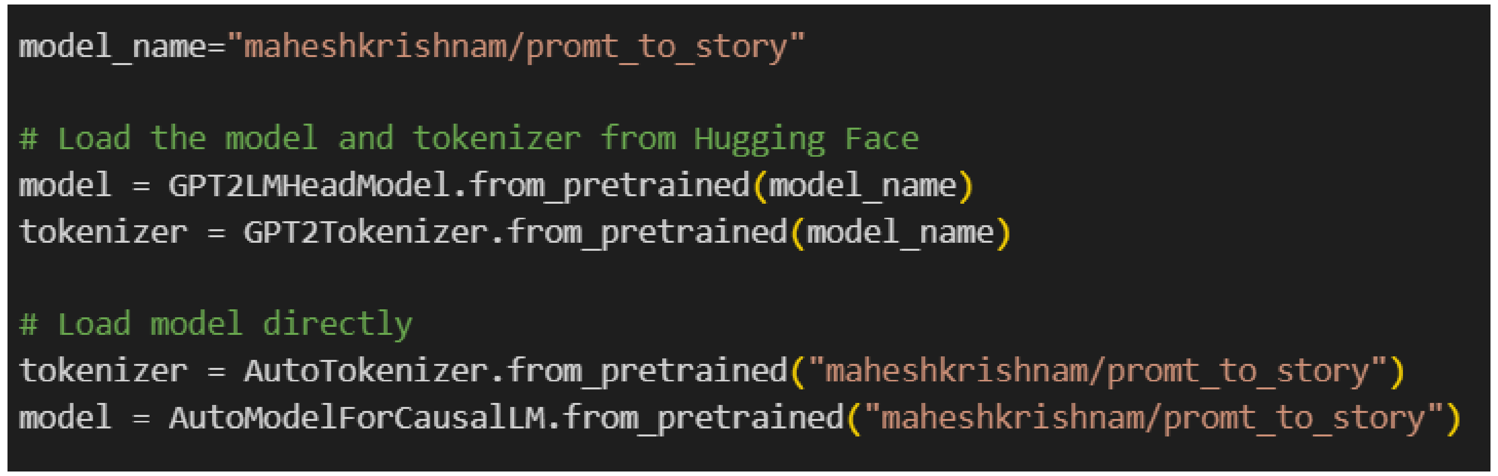
The SpeechT5 model for text-to-speech synthesis was trained using the CMU Arctic dataset, which contains 1,135 recordings of multiple speakers with different accents and speech characteristics

Resource :
load_dataset("Matthijs/cmu-arctic-xvectors", split="validation")

MODELS

GPT-2 FOR STORY GENERATION

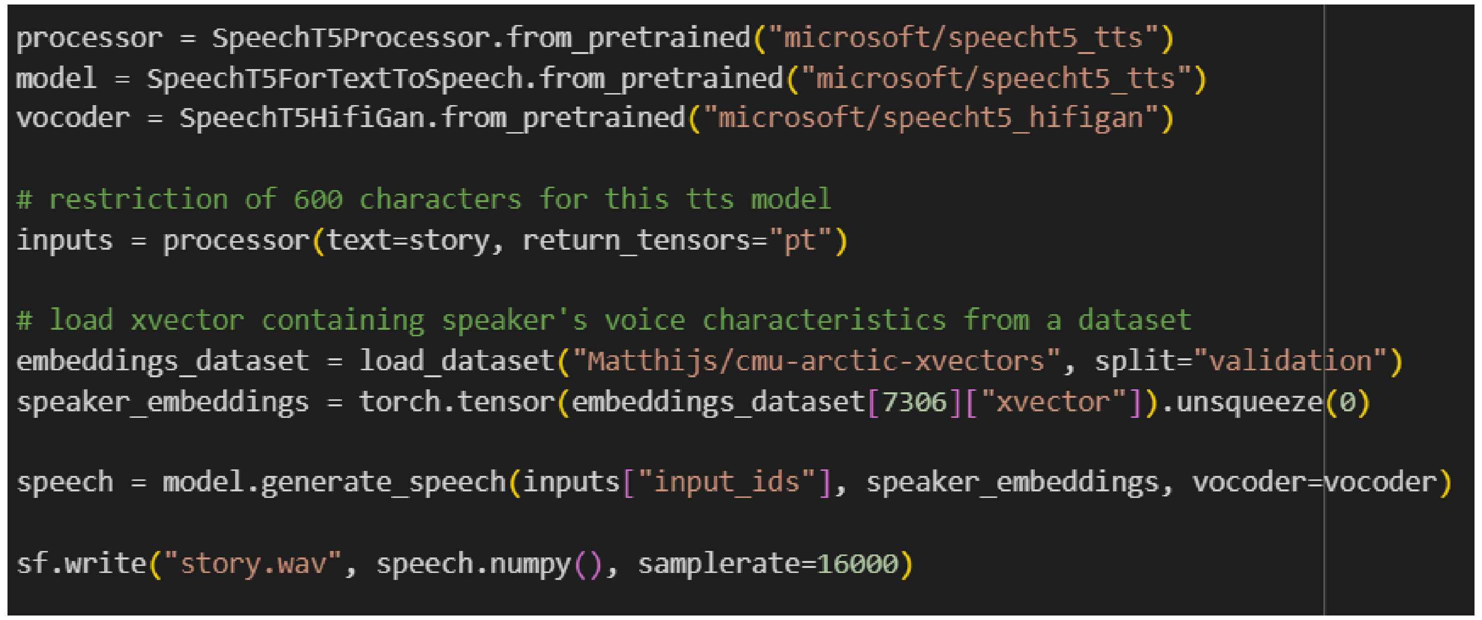
The GPT-2 model was fine-tuned using a large corpus of 100,000 stories and narratives. The dataset includes various genres and writing styles to ensure the model can generate diverse and engaging stories.



```
model_name="maheshkrishnam/promt_to_story"

# Load the model and tokenizer from Hugging Face
model = GPT2LMHeadModel.from_pretrained(model_name)
tokenizer = GPT2Tokenizer.from_pretrained(model_name)

# Load model directly
tokenizer = AutoTokenizer.from_pretrained("maheshkrishnam/promt_to_story")
model = AutoModelForCausalLM.from_pretrained("maheshkrishnam/promt_to_story")
```

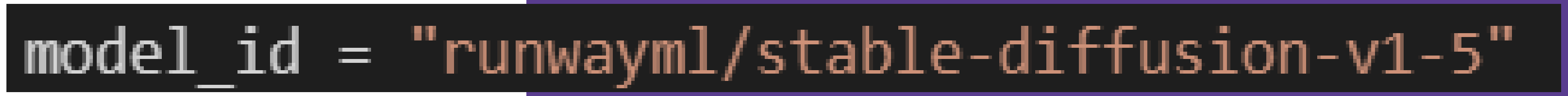
SPEECHT5 FOR TEXT-TO-SPEECH

The SpeechT5 model for text-to-speech synthesis was trained using the CMU Arctic dataset, which contains 1,135 recordings of multiple speakers with different accents and speech characteristics

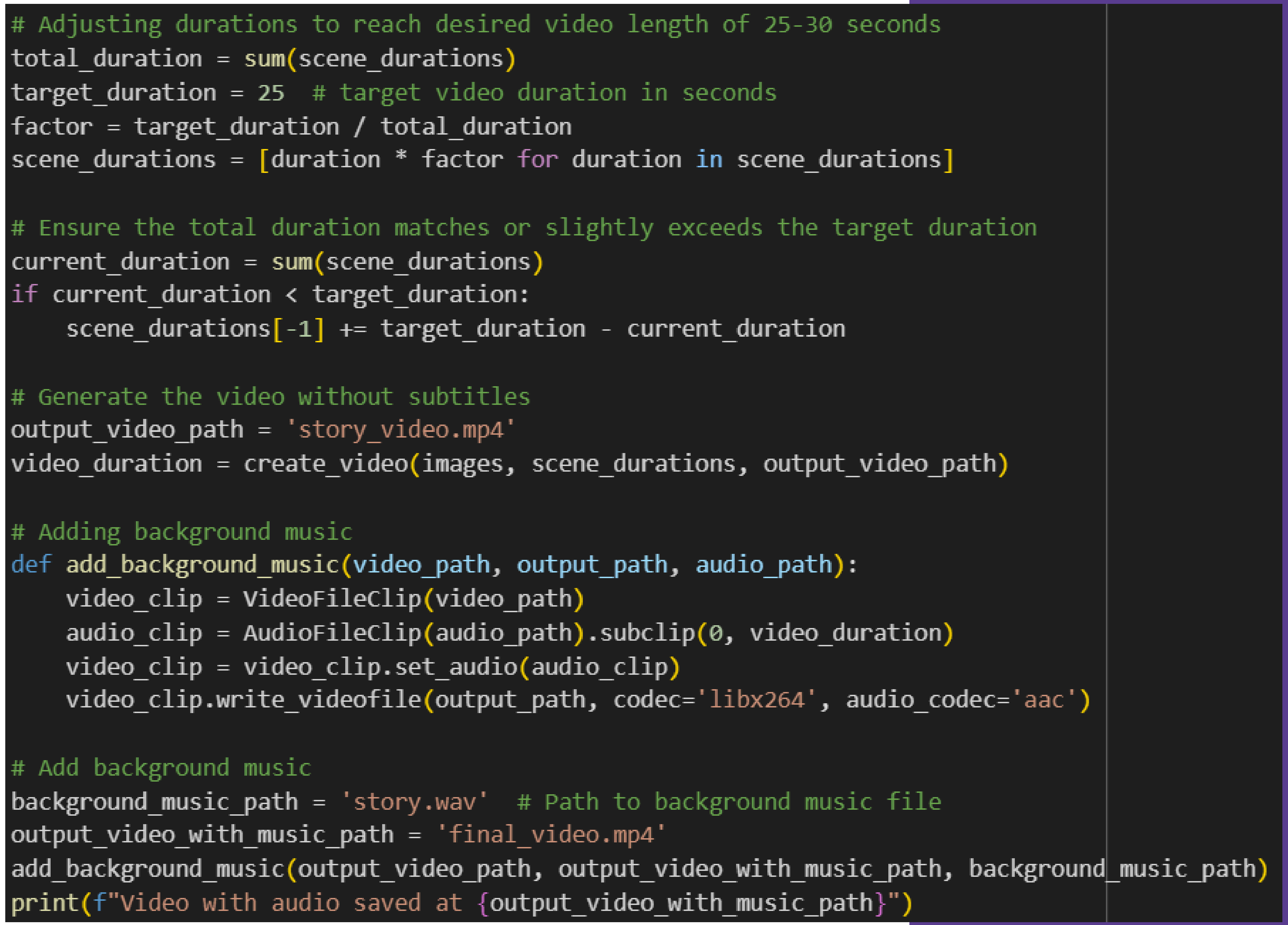


```
processor = SpeechT5Processor.from_pretrained("microsoft/speecht5_tts")
model = SpeechT5ForTextToSpeech.from_pretrained("microsoft/speecht5_tts")
vocoder = SpeechT5HifiGan.from_pretrained("microsoft/speecht5_hifigan")

# restriction of 600 characters for this tts model
inputs = processor(text=story, return_tensors="pt")

# load xvector containing speaker's voice characteristics from a dataset
embeddings_dataset = load_dataset("Matthijs/cmu-arctic-xvectors", split="validation")
speaker_embeddings = torch.tensor(embeddings_dataset[7306]["xvector"]).unsqueeze(0)

speech = model.generate_speech(inputs["input_ids"], speaker_embeddings, vocoder=vocoder)

sf.write("story.wav", speech.numpy(), samplerate=16000)
```

STABLE DIFFUSION FOR VIDEO GENERATION



```
model_id = "runwayml/stable-diffusion-v1-5"
```



```
# Adjusting durations to reach desired video length of 25-30 seconds
total_duration = sum(scene_durations)
target_duration = 25  # target video duration in seconds
factor = target_duration / total_duration
scene_durations = [duration * factor for duration in scene_durations]

# Ensure the total duration matches or slightly exceeds the target duration
current_duration = sum(scene_durations)
if current_duration < target_duration:
    scene_durations[-1] += target_duration - current_duration

# Generate the video without subtitles
output_video_path = 'story_video.mp4'
video_duration = create_video(images, scene_durations, output_video_path)

# Adding background music
def add_background_music(video_path, output_path, audio_path):
    video_clip = VideoFileClip(video_path)
    audio_clip = AudioFileClip(audio_path).subclip(0, video_duration)
    video_clip = video_clip.set_audio(audio_clip)
    video_clip.write_videofile(output_path, codec='libx264', audio_codec='aac')

# Add background music
background_music_path = 'story.wav'  # Path to background music file
output_video_with_music_path = 'final_video.mp4'
add_background_music(output_video_path, output_video_with_music_path, background_music_path)
print(f"Video with audio saved at {output_video_with_music_path}")
```

COMPUTATIONAL RESOURCES

HARDWARE

The models were trained and fine-tuned on an NVIDIA Tesla T4 GPU cluster with 12 GB of memory. Training the models required approximately 5-7 GPU hours in total (Google Colab Notebook)

SOFTWARE

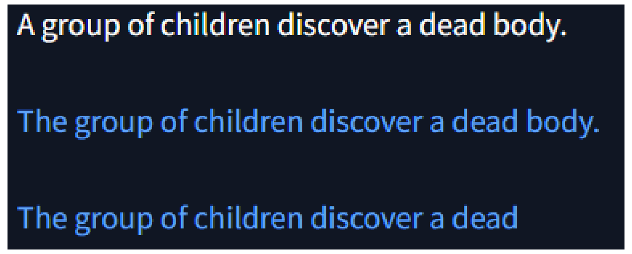
We used the PyTorch framework for model training and inference. Additional libraries such as Transformers, MoviePy, and Diffusers were employed for various tasks within the pipeline.

OUTPUT COMPARISON BEFORE AND AFTER FINE-TUNING

GPT-2 STORY GENERATION

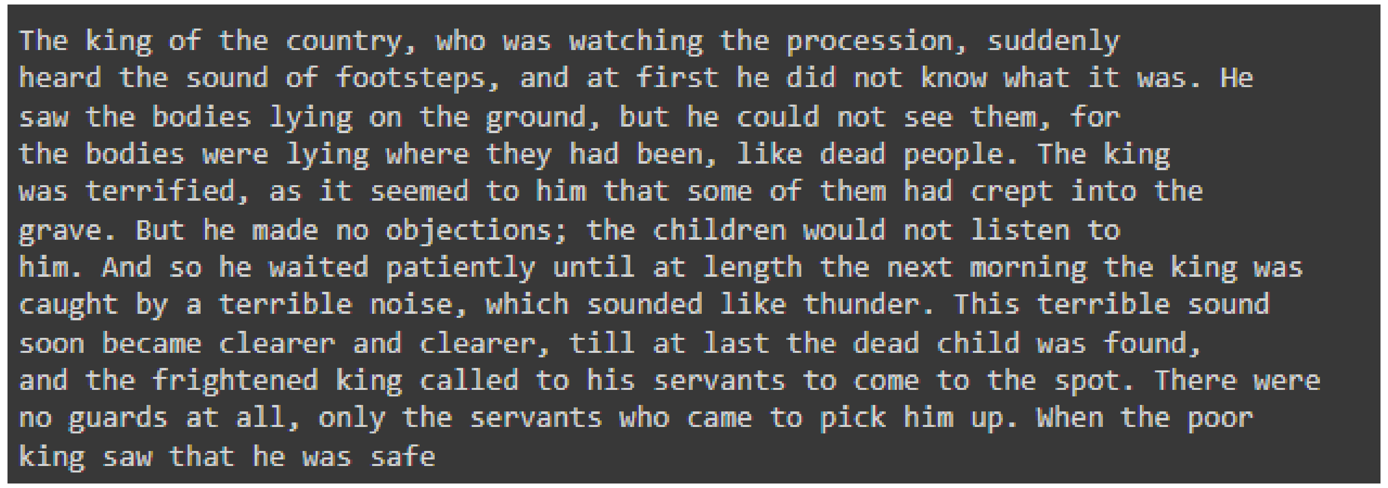
• Before Fine-tuning: Generated stories were less coherent and often deviated from the prompt. • After Fine-tuning: The stories became more engaging and aligned closely with the given prompts.

BEFORE FINE-TUNING



```
A group of children discover a dead body.

The group of children discover a dead body.

The group of children discover a dead
```

AFTER FINE-TUNING



```
The king of the country, who was watching the procession, suddenly
heard the sound of footsteps, and at first he did not know what it was. He
saw the bodies lying on the ground, but he could not see them, for
the bodies were lying where they had been, like dead people. The king
was terrified, as it seemed to him that some of them had crept into the
grave. But he made no objections; the children would not listen to
him. And so he waited patiently until at length the next morning the king was
caught by a terrible noise, which sounded like thunder. This terrible sound
soon became clearer and clearer, till at last the dead child was found,
and the frightened king called to his servants to come to the spot. There were
no guards at all, only the servants who came to pick him up. When the poor
king saw that he was safe
```

SPEECHT5 TEXT-TO-SPEECH

• Before Fine-tuning: Speech outputs were less natural and had noticeable artifacts. • After Fine-tuning: The speech became more natural and pleasant to listen to, closely mimicking human speech.

STABLE DIFFUSION VIDEO GENERATION

• Before Training: Initial visual outputs were less coherent and lacked detail. • After Training: Generated videos became more detailed and visually consistent with the story content.

CONTRIBUTION OF EACH MEMBER

MAHESH KRISHNAM

- Fine-tuned GPT-2 for story generation
- Fine-tuned SpeechT5 for Audio generation
- Stable Diffusion Video generation Model
- Final Integration of all Models

ANMOL SHARMA

- Collecting Dataset
- Fine-tuning GPT-2 for story generation
- Fine-tuned SpeechT5 for Audio generation

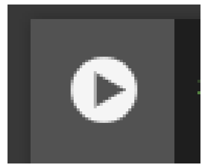
ADARSH GUPTA

- Collecting Dataset
- Fine-tuning GPT-2 for story generation
- Final Integration of all Models

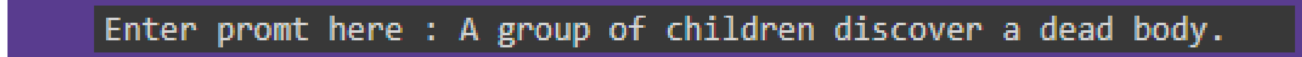
AMAN PUSHKAR

- Model Selection
- Stable Diffusion Video generation Model

WORKFLOW OF MODEL

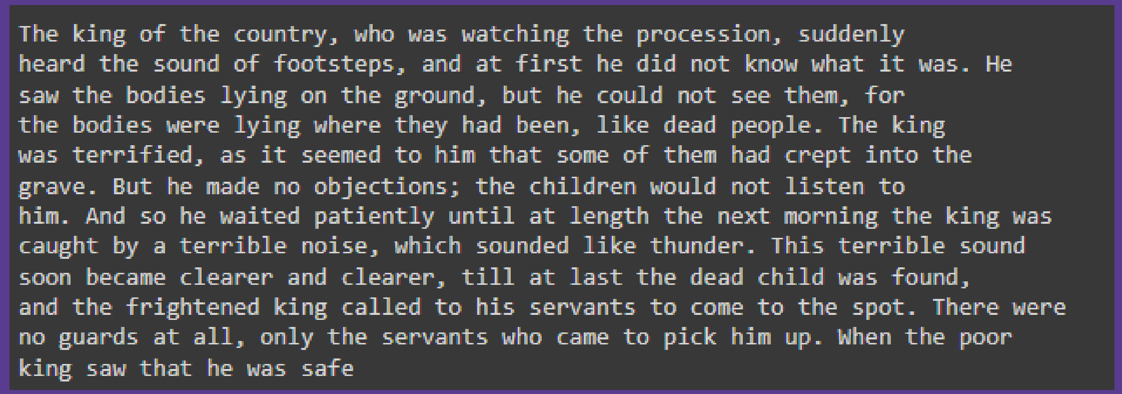


RUN THE CODE SHELL (START)



```
Enter promt here : A group of children discover a dead body.
```

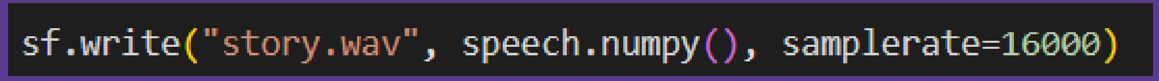
TEXT PROMPT



```
The king of the country, who was watching the procession, suddenly
heard the sound of footsteps, and at first he did not know what it was. He
saw the bodies lying on the ground, but he could not see them, for
the bodies were lying where they had been, like dead people. The king
was terrified, as it seemed to him that some of them had crept into the
grave. But he made no objections; the children would not listen to
him. And so he waited patiently until at length the next morning the king was
caught by a terrible noise, which sounded like thunder. This terrible sound
soon became clearer and clearer, till at last the dead child was found,
and the frightened king called to his servants to come to the spot. There were
no guards at all, only the servants who came to pick him up. When the poor
king saw that he was safe
```

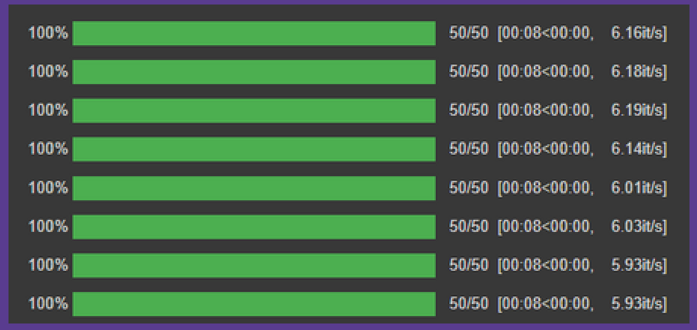
STORY GENERATION

STORY TO AUDIO

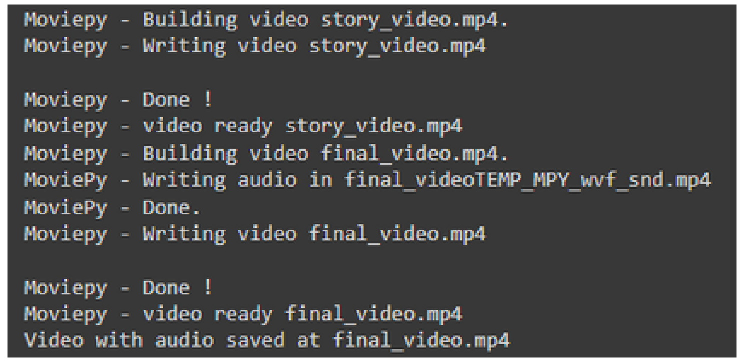


```
sf.write("story.wav", speech.numpy(), samplerate=16000)
```

STORY TO VIDEO



```
100% 50/50 [00:08<00:00,   6.16it/s]
100% 50/50 [00:08<00:00,   6.18it/s]
100% 50/50 [00:08<00:00,   6.19it/s]
100% 50/50 [00:08<00:00,   6.14it/s]
100% 50/50 [00:08<00:00,   6.01it/s]
100% 50/50 [00:08<00:00,   6.03it/s]
100% 50/50 [00:08<00:00,   5.93it/s]
100% 50/50 [00:08<00:00,   5.93it/s]
```



```
Moviepy - Building video story_video.mp4.
Moviepy - Writing video story_video.mp4

Moviepy - Done !
Moviepy - video ready story_video.mp4
Moviepy - Building video final_video.mp4.
MoviePy - Writing audio in final_videoTEMP_MPY_wvf_snd.mp4
MoviePy - Done.
Moviepy - Writing video final_video.mp4

Moviepy - Done !
Moviepy - video ready final_video.mp4
Video with audio saved at final_video.mp4
```

AUDIO AND VIDEO INTEGRATION

final_video.mp4

DOWNLOAD FINAL VIDEO (END)

THE END

THANK YOU !!