**STEPS TO RUN THE CODE**

SpeechT5 TTS Fine- Tuned Model for Technical Terms like CUDA, API, OAuth

* The fine-tuned text-to-speech model has been trained on the technical terms dataset and have been uploaded on HuggingFace.
* In order to run fine-tuned model, the “DEMO” part in the code (ipynb file uploaded on GitHub) can be copied and run.
* Steps:

***Process the model:***

model **=** SpeechT5ForTextToSpeech**.**from\_pretrained(

"Niha14/speecht5\_finetuned\_techincal\_data\_V2")

Process the dataset:

example **=** dataset["test"][60]

speaker\_embeddings **=** torch**.**tensor(example["speaker\_embeddings"])**.**unsqueeze(0)

give the input text:

text **=** "To accelerate deep learning models, CUDA is used for GPU computation."

***Preprocess the input data:***

number\_words **=** {

0: "zero", 1: "one", 2: "two", 3: "three", 4: "four", 5: "five", 6: "six", 7: "seven", 8: "eight", 9: "nine", '-': " ",

}

**def** number\_to\_words(number):

**if** number **<** 20:

**return** number\_words[number]

**elif** number **<** 100:

tens, unit **=** divmod(number, 10)

**return** number\_words[tens **\*** 10] **+** (" " **+** number\_words[unit] **if** unit **else** "")

**elif** number **<** 1000:

hundreds, remainder **=** divmod(number, 100)

**return** (number\_words[hundreds] **+** " yüz" **if** hundreds **>** 1 **else** "yüz") **+** (" " **+** number\_to\_words(remainder) **if** remainder **else** "")

**elif** number **<** 1000000:

thousands, remainder **=** divmod(number, 1000)

**return** (number\_to\_words(thousands) **+** " bin" **if** thousands **>** 1 **else** "bin") **+** (" " **+** number\_to\_words(remainder) **if** remainder **else** "")

**elif** number **<** 1000000000:

millions, remainder **=** divmod(number, 1000000)

**return** number\_to\_words(millions) **+** " milyon" **+** (" " **+** number\_to\_words(remainder) **if** remainder **else** "")

**elif** number **<** 1000000000000:

billions, remainder **=** divmod(number, 1000000000)

**return** number\_to\_words(billions) **+** " milyar" **+** (" " **+** number\_to\_words(remainder) **if** remainder **else** "")

**else**:

**return** str(number)

**def** replace\_numbers\_with\_words(text):

**def** replace(match):

number **=** int(match**.**group())

**return** number\_to\_words(number)

*# Find the numbers and change with words.*

result **=** re**.**sub(r'\b\d+\b', replace, text)

**return** result

In [ ]:

*# Function to clean up text using the replacement pairs*

**def** cleanup\_text(text):

**for** src, dst **in** replacements:

text **=** text**.**replace(src, dst)

**return** text

In [ ]:

converted\_text **=** replace\_numbers\_with\_words(text)

cleaned\_text **=** cleanup\_text(converted\_text)

final\_text **=** normalize\_text(cleaned\_text)

final\_text

***processing the inputs:***

inputs **=** processor(text**=**final\_text, return\_tensors**=**"pt")

***process the speechT5HifiGan model:***

**from** transformers **import** SpeechT5HifiGan

vocoder **=** SpeechT5HifiGan**.**from\_pretrained("microsoft/speecht5\_hifigan")

speech **=** model**.**generate\_speech(inputs["input\_ids"], speaker\_embeddings, vocoder**=**vocoder)

***Run and get the Output:***

**from** IPython.display **import** Audio

Audio(speech**.**numpy(), rate**=**16000)

* In order to run the code which is uploaded on GitHub:

1. You can easily download the .ipynb file from the Github.
2. After downloaded, run the file till ‘collator’ part, which is before the codes where model is trained.
3. After running till “collator” part, you can easily run the “DEMO” part so as to get the output.