Text to speech – speech synthesis

BLOCKADERY

SOFIJA STOJANOVIĆ 17462
KRISTINA STANOJEVIĆ 17432
STRAHINJA STAMENKOVIĆ 17413

SADRŽAJ

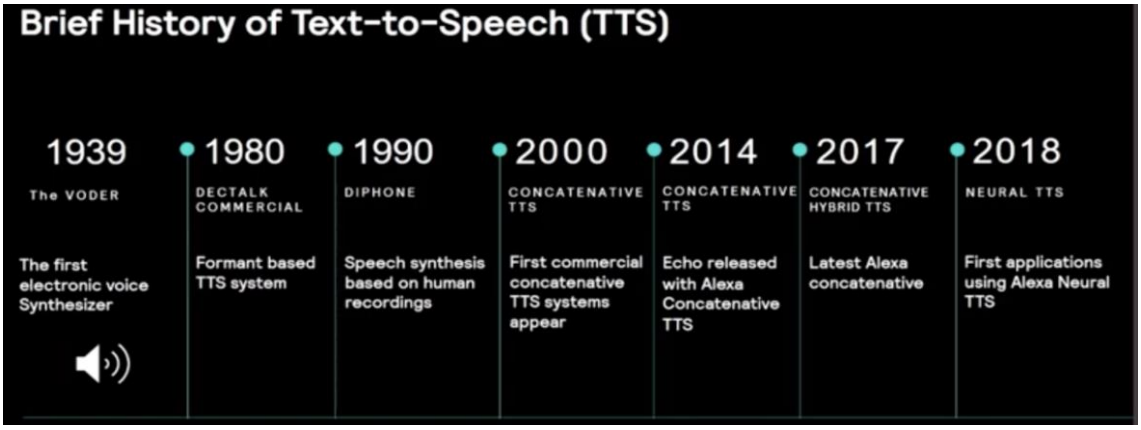
TEORIJSKI UVOD

Sinteza govora predstavlja pretvaranje pisanog ulaza u govorni izlaz odnosno veštačko proizvođenje ljudskog govora. Često se sreće i pojam Text-to-Speech(TTS). Sistem koji se koristi za sintezu govora naziva se speech synthesizer i može biti ugrađen u hardver ili softver.

Proces suprotan ovom naziva se Prepoznavanje govora (Speech recognition).

Ovi sistemi imaju veliku primenu u svakodnevnom životu. Najčešće ih koriste osobe sa invaliditetom, za koje su prvobitno i dizajnirani ovi sistemi – oni čiji je govor oštećen i koji pomoću ovih sistema mogu gotovo normalno da komuniciraju (Na primer, veoma poznati Stephen Hawking koristio je tts za komunikaciju sa ljudima oko sebe), takođe i osobe sa oštećenim vidom, kojima ovi sistemi omogućavaju samostalno korišćenje računara. Pored toga koriste ih osobe sa disleksijom i drugim problemima sa čitanjem, mogu se naći u video igrama, često ih koriste online streameri kako bi tekstualne donacije pretvorili u audio sadržaj, a takođe su korisne i deci koja uče da čitaju.

Prva mašina koja je imala cilj da imitira ljudski glas napravljena je 1769. godine. S napretkom razvoja ove oblasti akcenat TTS synthesizer-a se stavlja na sintezu glasa tako da on zvuči što prirodnije i da što više podseća na ljudski glas. Iako synthesizer glasa ima zadatak da obradi mnogo tipova ljudskog glasa, mnogo karakteristika, trajanje, glasnoću, akcentovanje i mnogo drugih atributa, pojedine deepfake tehnike uspevaju da sintetišu govor koji je gotovo nemoguće razlikovati od ljudskog.



Pored „ljudskosti“ sintetisanog glasa, kvalitet rezultata se može meriti i količinom uzoraka glasa koje je potrebno dati. Što je veći broj podataka, bolji će rezultati biti. Najnaprednije tehnologije koriste neuronske mreže koje procesiraju 10000 glasovnih uzoraka u sekundi.

Neki eksperti kažu da će do 2024.godine biti više glasovnih asistenata nego ljudi na planeti.

Načini za sintezu glasa

Danas postoji mnogo načina da se dobije dobra sinteza govora.

Sve veći akcenat se stavlja na generisanju govora različitih jezika, pa zbog toga postoje biblioteke na skoro svim svetskim jezicima. Brzina kojom se prikupljaju uzorci govora da bi se dobio što prirodniji glas je ogromna i sinteza govora na osnovu što manjeg broja uzorka je pravac u kom treba razvijati ovu oblast.

1. Online sajtovi za TTS

Postoji veliki broj online sajtova koji pružaju ovakve usluge sa velikim izborom različitih kategorija glasova, likova iz crtaća, stvarnih ljudi ili kompjuterski generisanih glasova (likovi iz Star Wars franšize, Marvel stripova...).

FakeYou. Deep Fake Text to Speech.

2. Easy-to-use biblioteke

U samo par linija koda generiše se jako dobar human-like govor. Neke od biblioteka imaju mogućnost odabira različitih glasova.

a. google
   a. instalacija
      sudo pip install gTTS
   b. kod
      from gtts import gTTS
      import os
      tts = gTTS(text='How are you', lang='en')

```
tts.save("hello.mp3")
os.system("mpg321 hello.mp3")
```

b. Python library
    a. instalacija
    ```
    sudo pip install pyttsx
    import pyttsx
    ```
    b. kod
    ```
    engine = pyttsx.init()
    engine.say(' How are you ')
    engine.runAndWait()
    ```

c. Espeach
    a. Instalacija
    ```
    sudo apt-get install espeak
    ```
    b. kod
    ```
    import os
    os.system("espeak 'How are you'")
    ```

3. Pretvaranje svog glasa u tekst open source aplikacijama

Osnovu obe aplikacije predstavlja Tacotron End-to-end TTS model. Osnova tacotrona je seq2seq model. Generiše tekst direktno iz karaktera, pretvara ga u spektogram, a kasnije u waveform. Ima 3.82 subjective 5scale MOS (mean opinion score) za US Engleski.

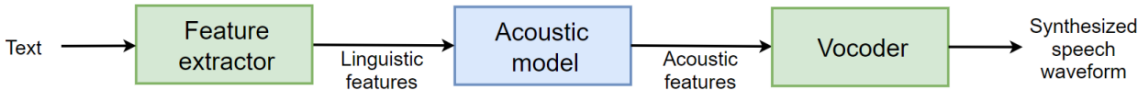
NVIDIA/tacotron2: Tacotron 2 - PyTorch implementation with faster-than-realtime inference (github.com)

Napredniji AI za TTSS je pravljen tako da se postigne najbolji odnos brzine učenja, brzine sinteze glasa i kvaliteta glasa.

coqui-ai/TTS: 🐸💬 - a deep learning toolkit for Text-to-Speech, battle-tested in research and production (github.com)

Mi ćemo se najviše baviti ovom poslednjom aplikacijom.

CorentinJ/Real-Time-Voice-Cloning: Clone a voice in 5 seconds to generate arbitrary speech in real-time (github.com)



Slika 1. The general Statistical parametric speech synthesis (SPSS) pipeline

Iako se moderni TTS ne odnosi direktno na SPSS, tok procesa je gotovo isti.

Feature extraction se odnosi na izdvajanje podataka koji će biti korisniji po nekoj metrici. Sinteza govora je jako složen proces i jednostavno davanje izgovora karaktera nije dovoljno da bi se postigao „human-like“ efekat.

Acoustic model čita dobijene podatke. Najčitljiviji oblik podataka za model je time-frequency. Spektogrami su jednostavniji i manje zbijeni u odnosu na waveform.

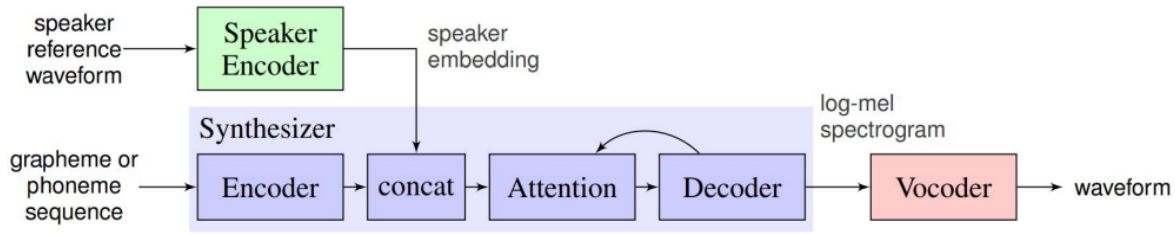
Vocoder predstavlja funkciju generisanja govora. Izbor vocodera je jedan od najvažnijih faktora za postizanje što boljeg kvaliteta.

Tri faze korišćena konkretnog frameworka su:

- Speaker encoder koji vrši izradu glasova na osnovu kratkih zvukova korisnika. Izrada glasova predstavlja značajnu reprezentaciju glasa, tako da su slični glasovi bliži u latentnom prostoru.

- Synthesizer korišćenjem Tacotrona generiše spektograme iz teksta. Tacotron obično radi jako brzo, 5 do 10 puta brže od onoga što znamo kao „real time“

- Vocoder koji korišćenjem WaveNet (pogodan za rad sa Tacotronom) generiše waveform od spektograma dobijenog od Synthesizer-a. WaveNet koristi tehnike deep learninga i teži da postigne što veću prirodnost glasa, ali je takođe poznat kao najsporija deep learning arhitektura.



Kroz interface se encoderu zadaje uzorak glasa korisnika (biće primera u odeljku Rad aplikacije). Predstavićemo ga sa u_{ij} to su zapisi u waveform domenu. Encoder ε pravi ulazni zvuk kao $\varepsilon(x_{ij}, w_\varepsilon)$ gde su w_ε parametri dekodera. Tako da se izrada glasa (embedding) definiše kao centaroid izrađenih glasova na osnovu datih uzoraka.

$$\mathbf{c}_i = \frac{1}{n}\sum_{j=1}^{n}\mathbf{e}_{ij}$$

Synthesizer S, sa parametrima w_{ε} , treba da definiše x_{ij} sa zadatim u_{ij} i c_i tako da dobijamo $x_{ij}=S(u_{ij}, t_{ij}; w_s)$. Vocoder V parametrizovan sa w_v, treba da definiše u_{ij} sa datim x_{ij} tako da dobijamo $u_{ij}=V(x_{ij};w_v)$. Framwork se može trenirati korišćenjem end2end sa sledećom funkcijom:

$$min_{\mathbf{w}_{\mathcal{E}},\mathbf{w}_{\mathcal{S}},\mathbf{w}_{\mathcal{V}}} L_{\mathcal{V}}(\mathbf{u}_{ij}, \mathcal{V}(\mathcal{S}(\mathcal{E}(\mathbf{x}_{ij};\mathbf{w}_{\mathcal{E}}), \mathbf{t}_{ij};\mathbf{w}_{\mathcal{S}});\mathbf{w}_{\mathcal{V}}))$$

Gde je L_v loss function waveform domena.

IDEJA PROJEKTA

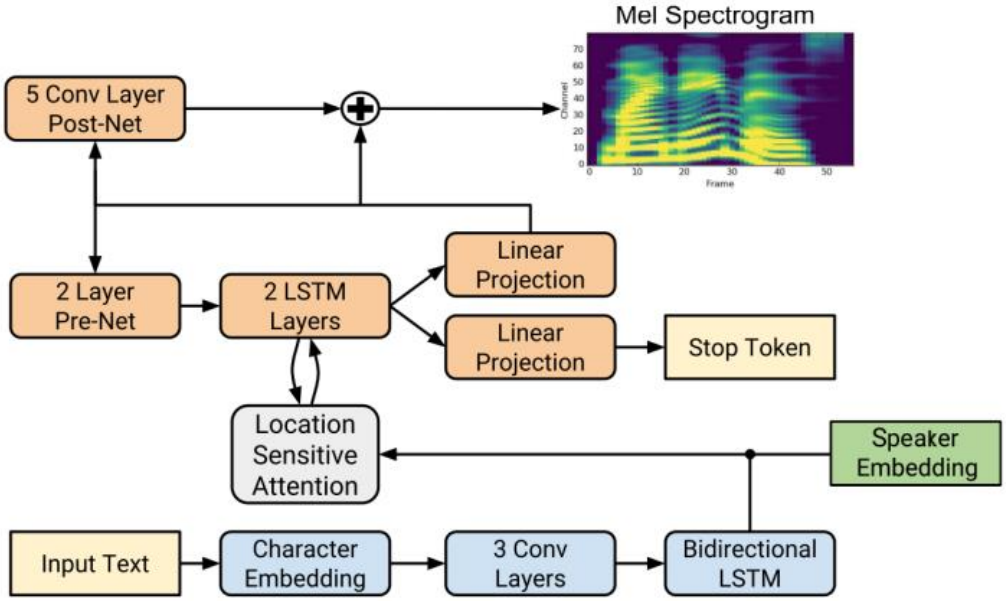
Prva ideja projekta je bila napraviti Text to speech syntetiser, ali s obzirom da je bez korišćenja već gotovih biblioteka to bio preobiman i prezahtevan posao, a korišćenjem gore opisanih biblioteka to predstavljalo prejednostavan zadatak, odlučili smo se da jednu od pomenutih TTS alata proširimo tako da se glas koji se dobije sintezom datog govora (koji može biti i u tom trenutku snimljen) koristi kao glas „virtual assistent-a“ koji će odgovarati na ponovo glasom postavljena pitanja. Tako da će projekat prestavljati kombinaciju text to speech, speech recognition i speech to text oblasti sa akcentom na sintezu govora.

IMPLEMENTACIJA

Implementacija RTVC (Real Time Voice Cloning) sastoji se od vocodera, encodera, i synthesizera.

Enkoder predstavlja 3 slojni LSTM (Long Short-Term Memory) sa 768 čvora sa projektantnim nivoom od 256 jedinica. Ulaz u enkoder predstavlja 40-o kanalni spektogram sa 25ms prozorom i 10ms korakom. Izlaz je L2 normalizovano stanje-vektor od 256 elementa.

Synthesizera predstavlja modifikivani Tacotron.



Plavi pravougaonici predstavljaju deo enkodera a narandžasti deo dekodera. Spektogram koji ulazi u synthesizer imaju više atributa od onih koji se dovode u enkoder. Generisani su u prozoru od 50ms, sa 12.5ms korakom i imaju 80 kanala. Ulaznom tekstu se ne proverava izgovor i on se prosleđuje takav kakav jeste. Postoji par procesa „peglanja“: zamena znakova i brojeva potpunom tekstualnom formom, prebacivanje svih karaktera u ASCII, skraćivanje razmaka na 1 blanko znak i prebacivanje svih slova u mala slova. Interpunkcijski znakovi se ne tumače. Taj efekat se može postići prebacivanjem teksta u novi red.

Aplikaciji RTVC dodate su biblioteke potrebne za speech recognition i speech recording.

Neophodni moduli PyAudio i PySpeech, koji se mogu instalirati pomoću pip-a.

U fajlu __init__.py u klasi Toolbox funkciji setup_event dodato je dugme „Talk to me“ i event koji se generiše klikom na zadato dugme. Event poziva funkciju javis(self) koja sluša naredbu, prepoznaje govor, pretvara ga u tekst i sintetiše taj tekst u glas sa onim uzorcima glasa koje je prethodno dobio.

```
133
134     def javis(self):
135
136         #!/usr/bin/env python3
137         # Requires PyAudio and PySpeech.
138         r = sr.Recognizer()
139
140         time.sleep(2)
141         self.ui.text_prompt.clear()
142         self.ui.text_prompt.appendPlainText( "Hello, can i know your name?")
143         self.synthesize()
144         self.vocode()
145
146
147         r = sr.Recognizer()
148
149
150         with sr.Microphone(device_index=0) as source:
151             print("Say something!")
152             audio = r.listen(source)
153
154         data = ""
155         try:
156                 # Uses the default API key
157                 # To use another API key: `r.recognize_google(audio, key="GOOGLE_SPEECH_RECOGNITION_API_KEY")`
158             data = r.recognize_google(audio)
159             print("You said: " + data)
160         except sr.UnknownValueError:
161             print("Google Speech Recognition could not understand audio")
162         except sr.RequestError as e:
163             print("Could not request results from Google Speech Recognition service; {0}".format(e))
164
165         self.ui.text_prompt.clear()
166         self.ui.text_prompt.appendPlainText( "Hi " +data+ " how can i help you")
167         self.synthesize()
168         self.vocode()
169
```

Pitanja koja možete postaviti su statički zadata.

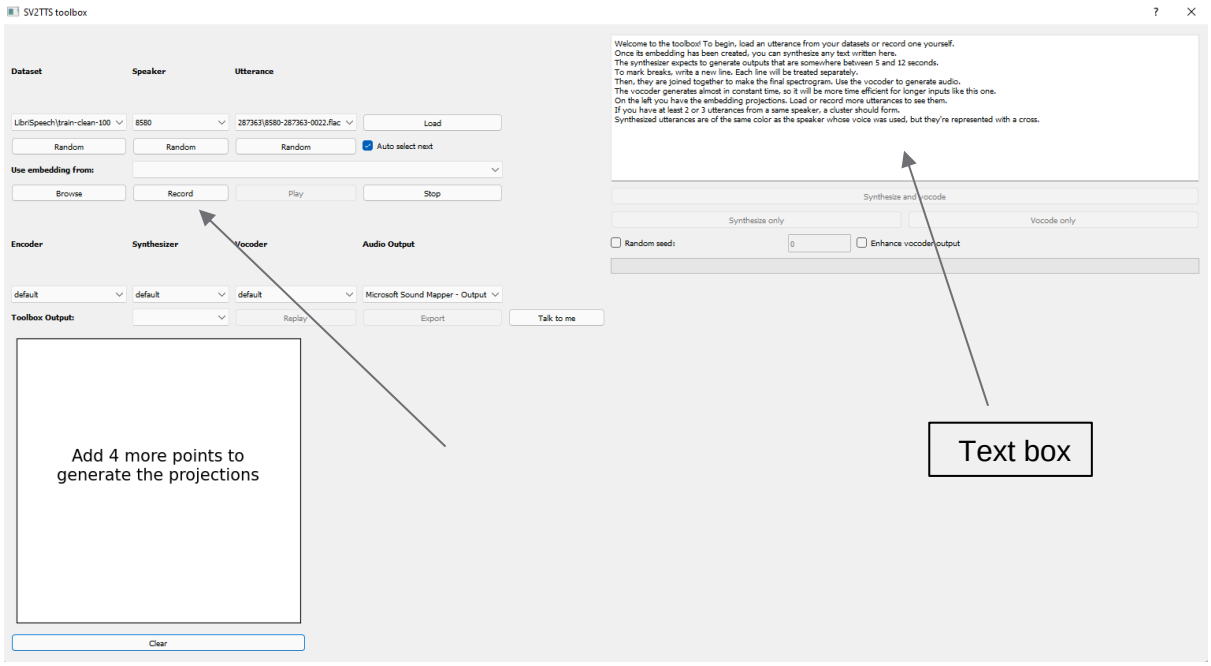
Za prepoznavanje glasa i pretvaranje govora u tekst u ovom delu zadužen je google speech recognition.

sr – speech recognition svojom funkcijom Recognizier() prepoznaje reči i ako se neka prepozna kao komanda sintetiše se glas funkcijom synthesize(), za zadati tekst koji predstavlja odgovor na komandu. Funkcija vocode() poziva vocoder i čuće se odgovor.

```
170         while 1:
171
172             # Record Audio
173             r = sr.Recognizer()
174
175
176             with sr.Microphone(device_index=0) as source:
177             #with DuckTypedMicrophone() as source:
178                 print("Say something!")
179                 audio = r.listen(source)
180
181             # Speech recognition using Google Speech Recognition
182             data = ""
183             try:
184                 # Uses the default API key
185                 # To use another API key: `r.recognize_google(audio, key="GOOGLE_SPEECH_RECOGNITION_API_KEY")`
186                 data = r.recognize_google(audio)
187                 print("You said: " + data)
188             except sr.UnknownValueError:
189                 print("Google Speech Recognition could not understand audio")
190             except sr.RequestError as e:
191                 print("Could not request results from Google Speech Recognition service; {0}".format(e))
192
193
194             if "how are you" in data:
195
196                 self.ui.text_prompt.clear()
197                 self.ui.text_prompt.appendPlainText("I am fine thank you!")
198                 self.synthesize()
199                 self.vocode()
200
201
202
203             if "what time is it" in data:
204
205                 self.ui.text_prompt.clear()
206                 self.ui.text_prompt.appendPlainText(ctime())
207                 self.synthesize()
208                 self.vocode()
209
210
211
212             if "where is" in data:
213                 data = data.split(" ")
214                 location = data[2]
215
216             time.sleep(10)
```

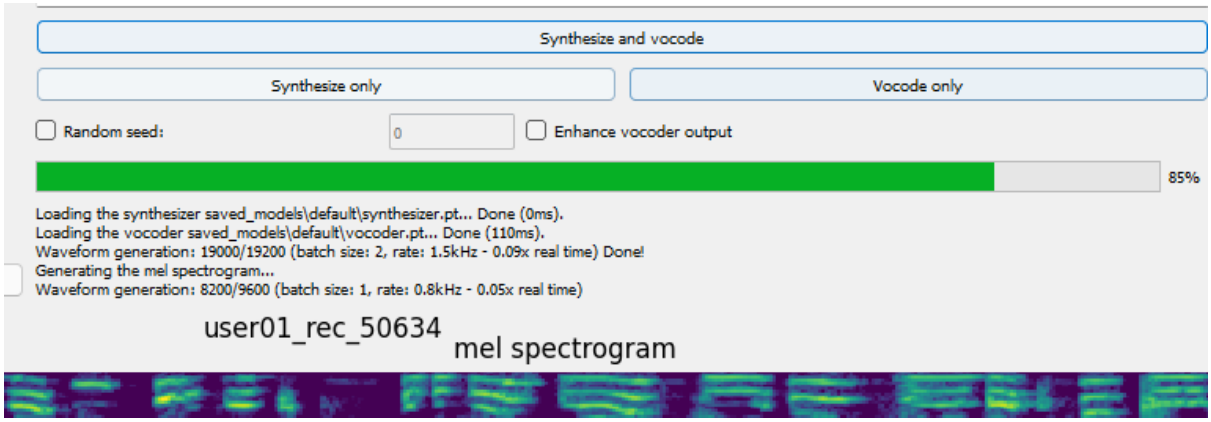
RAD APLIKACIJE

Pre pokretanja aplikacije potrebno je instalirati Pytorch i Apex. Pokretanjem aplikacije komandom python demo_tolbox.py dobija se sledeći interfejs.

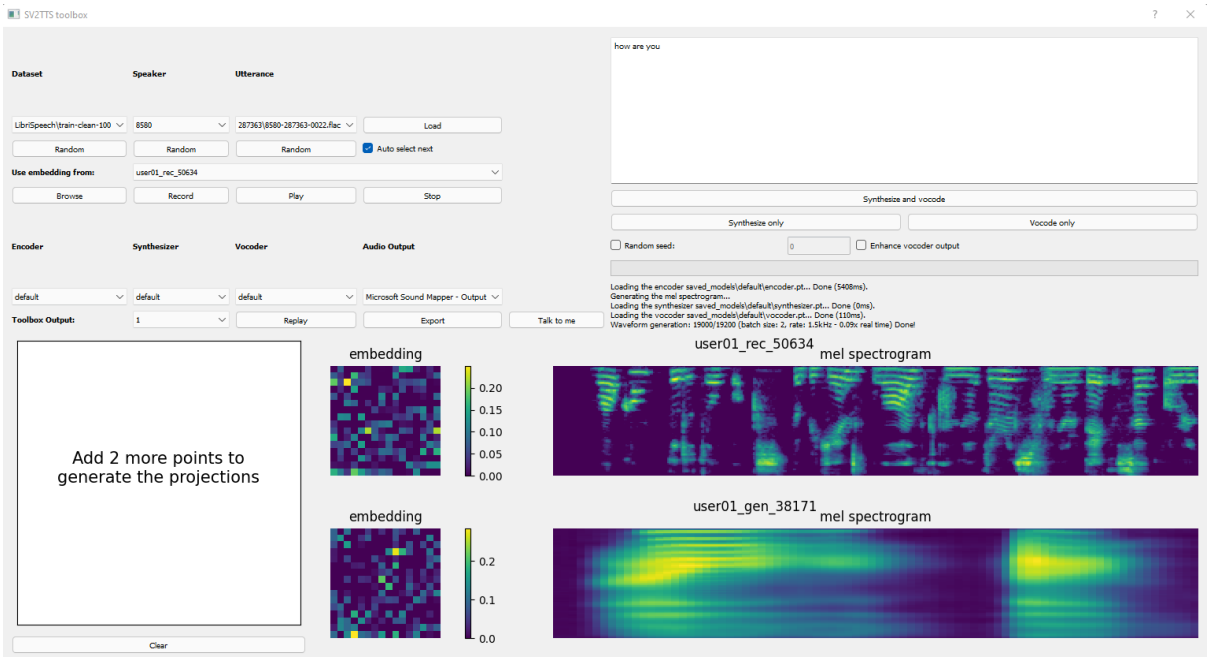


U odeljku dataset može se odabrati neki od prethodno generisanih setova uzoraka zvukova na osnovu kog će se sintetisati govor. Klikom na Record dugme započinje se snimanje svog glasa u trajanju od 5 sec. Nakon snimanja glasa treba odabrati opciju Synthesize and vocode ili synthesize only ukoliko taj glas tek kasnije želimo koristiti, gotovo odmah se sintetiše glas.

Praćenje rada vocodera i synthesizer-a se može vršiti ovde:



Tekst koji će izgovoriti zadaje se u text box-u. Vocoder pokreće izgovor i glas se može čuti. Rezultati nakon snimanja 2 uzorka izgledaju ovako:



Preporučen minimalan broj unosa uzorka je 4.

Klikom na Talk to me pokreće se asistent koji će odgovarati na postavljena pitanja tim sintetisanim glasom.

Za sintezu glasa je ostavljeno 10sec tako da se asistent aktivira tek nakon tog vremena.

Snimci glasova:

SynthesizedText.flac je sledeći tekst izgovoren sintetisanim glasom.

“Welcome to the toolbox! To begin, load an utterance from your datasets or record one yourself.
Once its embedding has been created, you can synthesize any text written here.
The synthesizer expects to generate outputs that are somewhere between 5 and 12 seconds.
To mark breaks, write a new line. Each line will be treated separately.
Then, they are joined together to make the final spectrogram. Use the vocoder to generate audio.
The vocoder generates almost in constant time, so it will be more time efficient for longer inputs like this one.
On the left you have the embedding projections. Load or record more utterances to see them.
If you have at least 2 or 3 utterances from a same speaker, a cluster should form.
Synthesized utterances are of the same color as the speaker whose voice was used, but they're represented with a cross.”

Assistent.flac je primer razgovora sa assistentom sa gore navedenim sintetisanim glasom. Dobar deo snimka je bez zvuka jer je asistentu potrebno vremena da generiše odgovor.

Razgovor:

Assistent: *Hello, can i know your name?*

Korisnik: *Sofija.*

Assistent: *Hi Sofija, how can i help you?*

Korisnik: *How are you.*

Assistent: *I am fine thank you.*

Korisnik: *What time is it?*

Assistent: *2 jan 14:00 25 2022*

Korisnik: *Tell me a joke.*

Assistent: *Did you hear about the guy who invented the knock knock joke? He won the no bell prize.*

Git sa snimcima: sofija99/snimciAI (github.com)

REFERENCE

Jemine, Corentin, Automatic Multispeaker Voice Cloning, 2019

Sercan Arik, Gregory Diamos, Andrew Gibiansky, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep voice 2: Multi-speaker neural text-to-speech, 2017