DC-TTS

TEXT-TO-SPEECH SYSTEM BASED ON DEEP CONVOLUTIONAL NETWORKS WITH GUIDED ATTENTION

TEHNICI DE COMPRESIE A SEMNALULUI VOCAL

GAL OSCAR

CONTENT

INTRODUCTION

01

Text-to-speech (TTS) is getting more and more common recently, and is getting to be a basic user interface for many systems. To further promote the use of TTS in various systems, it is significant to develop a manageable, maintainable, and extensible TTS component that is accessible to speech non-specialists, enterprising individuals and small teams.

02

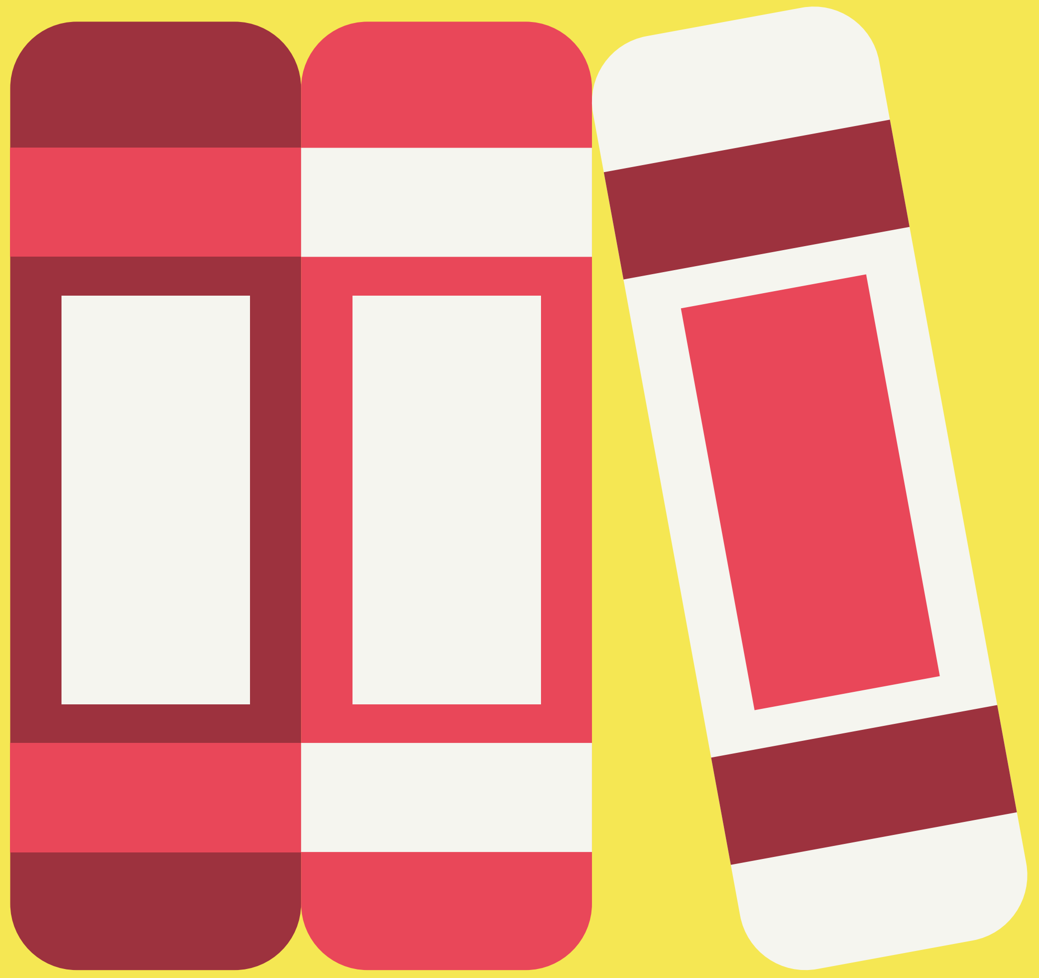
Traditional TTS systems, however, are not necessarily friendly for them, since they are typically composed of many domain-specific modules. For example, a typical parametric TTS system is an elab- orate integration of many modules e.g. a text analyzer, an F0 gen- erator, a spectrum generator, a pause estimator, and a vocoder that synthesize a waveform from these data, etc.

CONCEPTS AND DEFINITION

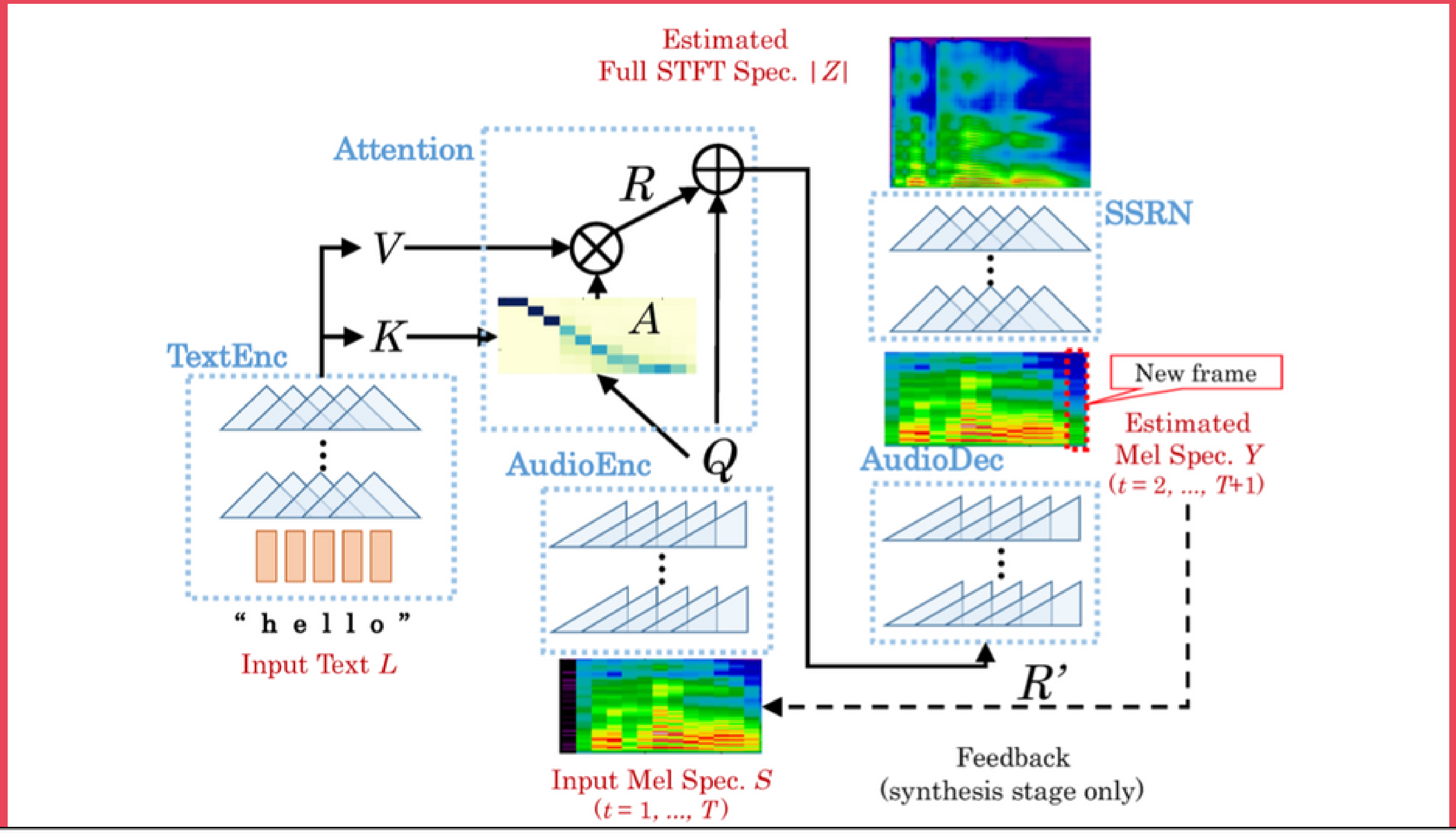
RECENTLY, DEEP LEARNING-BASED TTS SYSTEMS HAVE BEEN INTENSIVELY STUDIED, AND SURPRISINGLY HIGH QUALITY RESULTS ARE OBTAINED IN SOME OF RECENT STUDIES.

SOME OF THEM TRIED TO REDUCE THE DEPENDENCY ON HAND-ENGINEERED INTERNAL MODULES. THE MOST EXTREME TECHNIQUE IN THIS TREND WOULD BE TACOTRON, WHICH DEPENDS ONLY ON MEL AND LINEAR SPECTRO-GRAMS, AND NOT ON ANY OTHER SPEECH FEATURES E.G. F0. OUR METHOD IS CLOSE TO TACOTRON IN A SENSE THAT IT DEPENDS ONLY ON THESE SPECTRAL REPRESENTATIONS OF AUDIO SIGNALS.

MOST OF THE EXISTING METHODS ABOVE USE RNN, A NATURAL TECH-NIQUE OF TIME SERIES PREDICTION. AN EXCEPTION IS WAVENET, WHICH IS FULLY CONVOLUTIONAL. OUR METHOD IS ALSO BASED ONLY ON CNN BUT OUR USAGE OF CNN WOULD BE DIFFERENT FROM WAVENET, AS WAVENET IS A KIND OF A VOCODER, OR A BACK-END, WHICH SYNTHESIZES A WAVEFORM FROM SOME CONDITIONING INFORMATION THAT IS GIVEN BY FRONT-END COMPONENTS. ON THE OTHER HAND, OURS IS RATHER A FRONT-END (AND MOST OF BACK-END PROCESSING). WE USE CNN TO SYNTHESIZE A SPECTROGRAM, FROM WHICH A SIMPLE VOCODER CAN SYNTHESIZE A WAVEFORM.



Since some literature suggest that the staged synthesis from low-to high-resolution has advantages over the direct synthesis of high-resolution data, we synthesize the spectrograms using the following two networks. Text2Mel, which synthesizes a mel spectrogram from an input text, and Spectrogram Super-resolution Network (SSRN), which synthesizes a full STFT spectrogram from a coarse mel spectrogram.



Network architecture

Details for each component

$$\mathsf{TextEnc}(L) := (\mathsf{HC}^{2d\leftarrow 2d}_{1\star 1})^2 \lhd (\mathsf{HC}^{2d\leftarrow 2d}_{3\star 1})^2 \lhd (\mathsf{HC}^{2d\leftarrow 2d}_{3\star 27} \lhd \mathsf{HC}^{2d\leftarrow 2d}_{3\star 9} \lhd \mathsf{HC}^{2d\leftarrow 2d}_{3\star 3} \lhd \mathsf{HC}^{2d\leftarrow 2d}_{3\star 1})^2 \lhd \mathsf{C}^{2d\leftarrow 2d}_{1\star 1} \lhd \mathsf{ReLU} \lhd \mathsf{C}^{2d\leftarrow e}_{1\star 1} \lhd \mathsf{CharEmbed}^{e\text{-dim}}(L).$$

$$\mathsf{AudioEnc}(S) := (\mathsf{HC}^{d\leftarrow d}_{3\star 3})^2 \lhd (\mathsf{HC}^{d\leftarrow d}_{3\star 27} \lhd \mathsf{HC}^{d\leftarrow d}_{3\star 9} \lhd \mathsf{HC}^{d\leftarrow d}_{3\star 3} \lhd \mathsf{HC}^{d\leftarrow d}_{3\star 1})^2 \lhd \mathsf{C}^{d\leftarrow d}_{1\star 1} \lhd \mathsf{ReLU} \lhd \mathsf{C}^{d\leftarrow d}_{1\star 1} \lhd \mathsf{ReLU} \lhd \mathsf{C}^{d\leftarrow F}_{1\star 1}(S).$$

$$\mathsf{AudioDec}(R') := \sigma \lhd \mathsf{C}^{F\leftarrow d}_{1\star 1} \lhd (\mathsf{ReLU} \lhd \mathsf{C}^{d\leftarrow d}_{1\star 1})^3 \lhd (\mathsf{HC}^{d\leftarrow d}_{3\star 1})^2 \lhd (\mathsf{HC}^{d\leftarrow d}_{3\star 27} \lhd \mathsf{HC}^{d\leftarrow d}_{3\star 9} \lhd \mathsf{HC}^{d\leftarrow d}_{3\star 3} \lhd \mathsf{HC}^{d\leftarrow d}_{3\star 1}) \lhd \mathsf{C}^{d\leftarrow 2d}_{1\star 1}(R').$$

$$\mathsf{SSRN}(Y) := \sigma \lhd \mathsf{C}^{F'\leftarrow F'}_{1\star 1} \lhd (\mathsf{ReLU} \lhd \mathsf{C}^{F'\leftarrow F'}_{1\star 1})^2 \lhd \mathsf{C}^{F'\leftarrow 2c}_{1\star 1} \lhd (\mathsf{HC}^{2c\leftarrow 2c}_{3\star 1})^2 \lhd \mathsf{C}^{2c\leftarrow c}_{1\star 1} \lhd (\mathsf{HC}^{c\leftarrow c}_{3\star 3} \lhd \mathsf{HC}^{c\leftarrow c}_{3\star 1} \lhd \mathsf{D}^{c\leftarrow c}_{2\star 1})^2 \lhd (\mathsf{HC}^{c\leftarrow c}_{3\star 3} \lhd \mathsf{HC}^{c\leftarrow c}_{3\star 1}) \lhd \mathsf{C}^{c\leftarrow F}_{1\star 1}(Y).$$

For simplicity, we trained Text2Mel and SSRN independently and asynchronously using different GPUs. All network parameters were initialized using He’s Gaussian initializer. Both networks were trained by the ADAM optimizer. When training SSRN, we randomly extracted short sequences of length T = 64 for each iteration to save memory usage. To reduce the disk access, we reduced the frequency of creating the snapshot of parameters to only once per 5K iterations. Other parameters are shown in Table 1 (next slide).

As it is not easy for us to reproduce the original results of Tacotron, we instead used a ready-to-use model for comparison, which seemed to produce the most reasonable sounds in the open implementations. It is reported that this model was trained using LJ Dataset for 12 days (877K iterations) on a GTX 1080 Ti, newer GPU than ours. Note, this iteration is still much less than the original Tacotron, which was trained for more than 2M iterations.

We evaluated mean opinion scores (MOS) of both methods by crowdsourcing on Amazon Mechanical Turk using crowdMOS toolkit. We used 20 sentences from *Harvard Sentences* List 1&2. The audio data were synthesized using five methods shown in Table 2 (next slide). The crowdworkers rated these 100 clips from 1 (Bad) to 5 (Excellent). Each worker is supposed to rate at least 10 clips. To obtain more responses of higher quality, we set a few incentives shown in the literature. The results were statistically processed using the method shown in the literature.

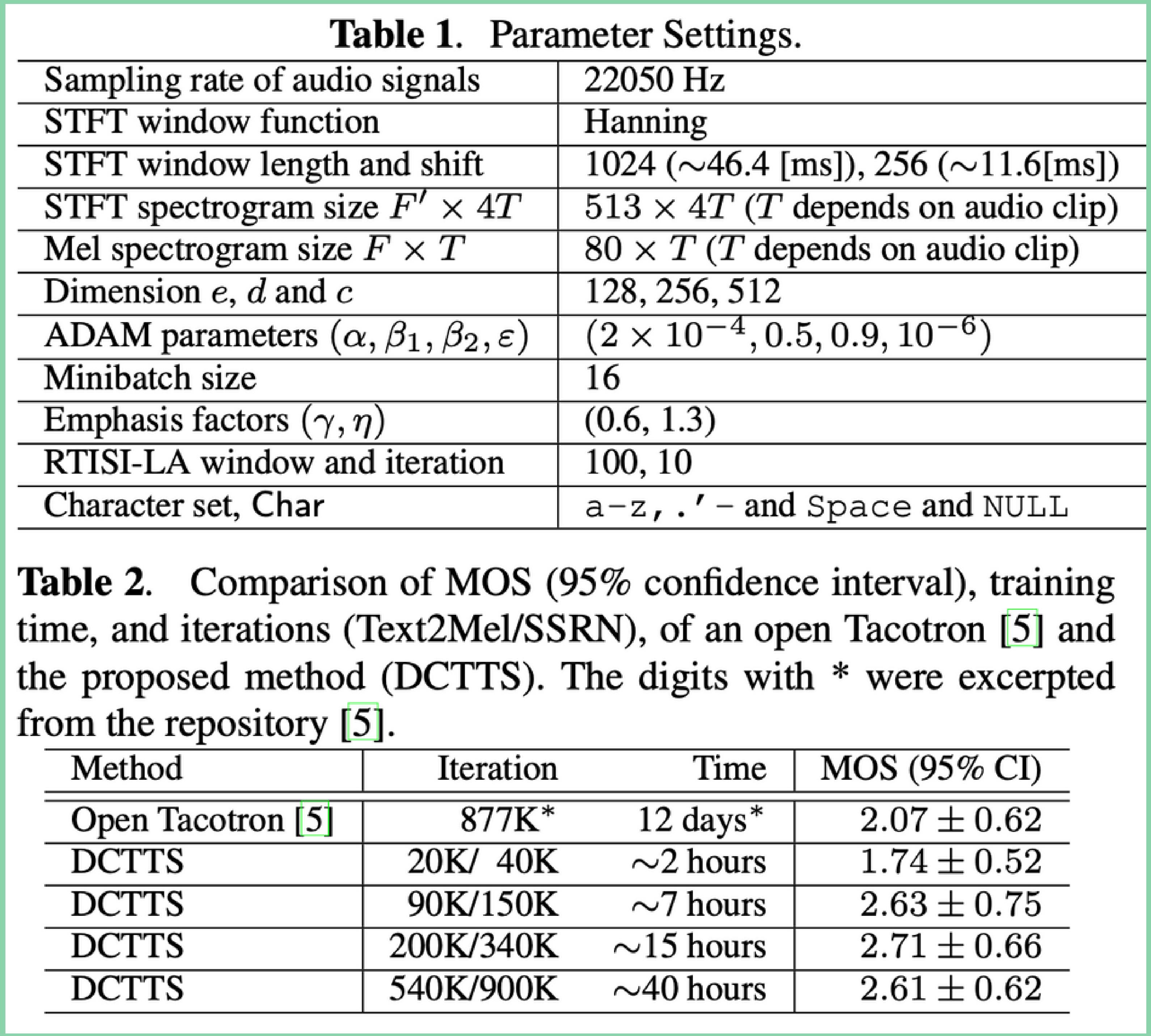


Table 1. Parameter Settings.

Sampling rate of audio signals	22050 Hz
STFT window function	Hanning
STFT window length and shift	1024 (∼46.4 [ms]), 256 (∼11.6[ms])
STFT spectrogram size $F' \times 4T$	$513 \times 4T$ (T depends on audio clip)
Mel spectrogram size $F \times T$	$80 \times T$ (T depends on audio clip)
Dimension e, d and c	128, 256, 512
ADAM parameters $(\alpha, \beta_1, \beta_2, \varepsilon)$	$(2 \times 10^{-4}, 0.5, 0.9, 10^{-6})$
Minibatch size	16
Emphasis factors (γ, η)	(0.6, 1.3)
RTISI-LA window and iteration	100, 10
Character set, Char	`a-z,.'-` and `Space` and `NULL`

Table 2. Comparison of MOS (95% confidence interval), training time, and iterations (Text2Mel/SSRN), of an open Tacotron [5] and the proposed method (DCTTS). The digits with * were excerpted from the repository [5].

Method	Iteration	Time	MOS (95% CI)
Open Tacotron [5]	877K*	12 days*	2.07 ± 0.62
DCTTS	20K/ 40K	∼2 hours	1.74 ± 0.52
DCTTS	90K/150K	∼7 hours	2.63 ± 0.75
DCTTS	200K/340K	∼15 hours	2.71 ± 0.66
DCTTS	540K/900K	∼40 hours	2.61 ± 0.62

SOURCE CODE 1

```
import sys
sys.path.append(project_name)

import warnings
warnings.filterwarnings("ignore") # ignore warnings in this notebook

import numpy as np
import torch

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

from hparams import HParams as hp
from audio import save_to_wav
from models import Text2Mel, SSRN
from datasets.lj_speech import vocab, idx2char, get_test_data
```

Step 1: Imports

```
torch.set_grad_enabled(False)
text2mel = Text2Mel(vocab)
text2mel.load_state_dict(torch.load("ljspeech-
text2mel.pth").state_dict())
text2mel = text2mel.eval()
ssrn = SSRN()
ssrn.load_state_dict(torch.load("ljspeech-ssrn.pth").state_dict())
ssrn = ssrn.eval()
```

Step 2: Train

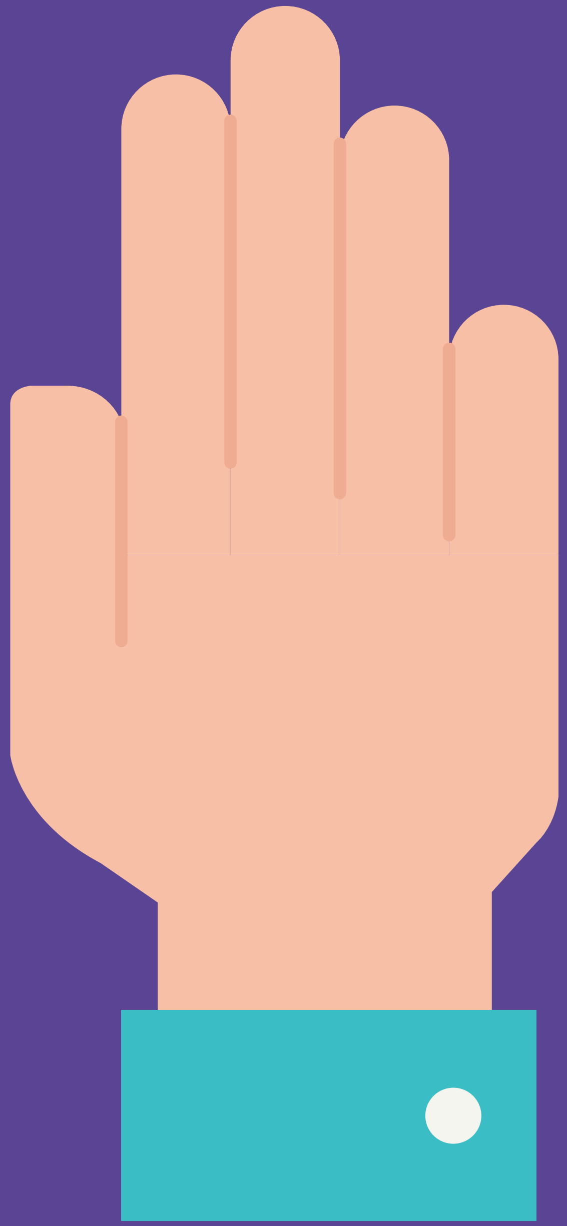
```
SENTENCES = [
 "Glue the sheet to the dark blue background.",
 "It's easy to tell the depth of a well.",
 "These days a chicken leg is a rare dish.",
 "Rice is often served in round bowls.",
 "The juice of lemons makes fine punch.",
 "The box was thrown beside the parked truck.",
 "The hogs were fed chopped corn and garbage.",
 "Four hours of steady work faced us.",
 "Large size in stockings is hard to sell.",
 "The boy was there when the sun rose.",
 "A rod is used to catch pink salmon.",
 "The source of the huge river is the clear spring.",
 "Kick the ball straight and follow through.",
 "Help the woman get back to her feet.",
 "A pot of tea helps to pass the evening.",
 "Smoky fires lack flame and heat.",
 "The soft cushion broke the man's fall.",
 "The salt breeze came across from the sea.",
 "The girl at the booth sold fifty bonds."
]
```

Step 3: Establish the sentences

```
for i in range(len(SENTENCES)):
    sentence = SENTENCES[i]
    normalized_sentence = "".join([c if c.lower() in vocab else '' for c in sentence])
    print(normalized_sentence)

    sentences = [normalized_sentence]
    max_N = len(normalized_sentence)
    L = torch.from_numpy(get_test_data(sentences, max_N))
    zeros = torch.from_numpy(np.zeros((1, hp.n_mels, 1), np.float32))
    Y = zeros
    A = None

    for t in range(hp.max_T):
    _, Y_t, A = text2mel(L, Y, monotonic_attention=True)
    Y = torch.cat((zeros, Y_t), -1)
    _, attention = torch.max(A[0, :, -1], 0)
    attention = attention.item()
    if L[0, attention] == vocab.index('E'): # EOS
    break

    _, Z = ssrn(Y)

    Z = Z.cpu().detach().numpy()
    save_to_wav(Z[0, :, :].T, '%d.wav' % (i + 1))
    IPython.display.display(Audio('%d.wav' % (i + 1), rate=hp.sr))
```

Step 4: Generate Audio

EXAMPLE

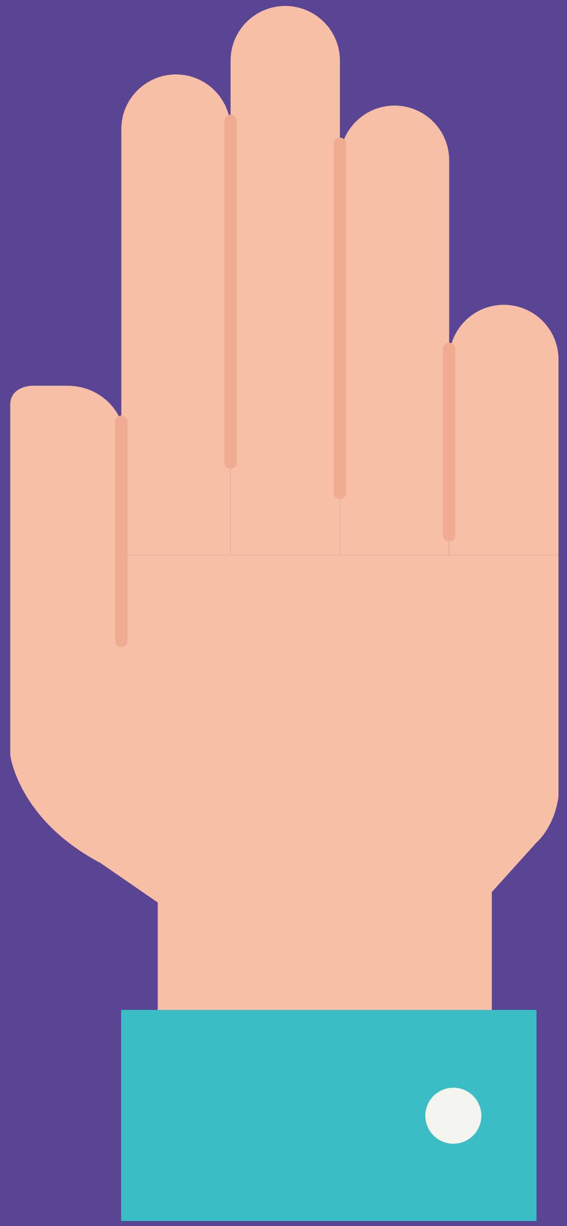


QUESTIONS

HOW MORE EFFICIENT IS THE PRESENTED METHOD IN COMPARISON TO TECOTRON ?

FOR WHAT WE ARE USING CNN?

WHAT TEXT2MEL AND SSRN DOES ?



QUESTIONS

HOW MORE EFFICIENT IS THE PRESENTED METHOD IN COMPARISON TO TECOTRON ?

Tacotron training for a dataset of 877k takes up to 12 days and the presented method on the same dataset takes up to 40h

FOR WHAT WE ARE USING CNN?

We use CNN to synthesize a spectrogram, from which a simple vocoder can synthesize a waveform.

WHAT TEXT2MEL AND SSRN DOES ?

Text2Mel, which synthesizes a mel spectrogram from an input text, and Spectrogram Super-resolution Network (SSRN), which synthesizes a full STFT spectrogram from a coarse mel spectrogram.