Speaker Identification using audio

Agenda

1. Prerequisites
2. Problem definition
3. Audio → Image
4. Transfer learning
5. Few-shot learning
6. Siamese networks

Prerequisites:

1. CNNs
2. Transfer Learning
3. Fourier Transform
4. Keras

Problem - Definition

Application : Smart-Speaker

Input : few short audio snippets

output : recognize the speaker

Speakers : ≤ 6

Datasets & Code

1. Transfer - Learning

LibriVox (scrape data)

https://github.com/hamzag95/voice-classification

https://towardsdatascience.com/automatic-speaker-recognition-using-transfer-learning-6fab63e34e74

Youtube

Christopher Gill, Hamza Ghani, Yousef Abdelrazzaq, Minkoo Park [UT-Austin]

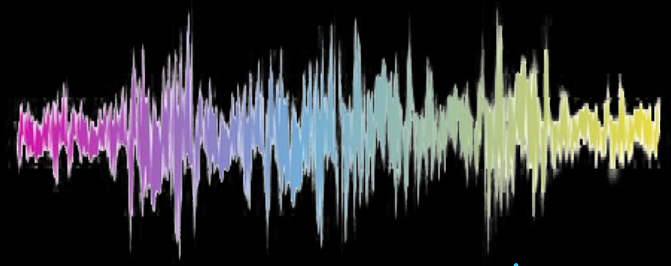
2. Siamese - network based few-shot learning

https://github.com/oscarknagg/voicemap

LibriSpeech

By Oscar Knagg

Audio - data



- Sequence data ⇒ RNN / Attention / Transformer models

- Short - Sequence (~5 sec long)

- few - samples of data

⇓

Transfer Learning

not much scope for Transfer learning

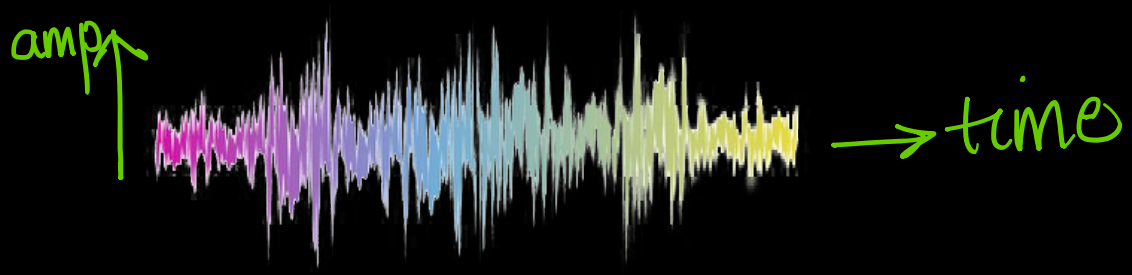
Audio ⇒ Image

- Image-Kala ⇒ CNNs ⇒ Pretrained models (AlexNet, VGG, ...)

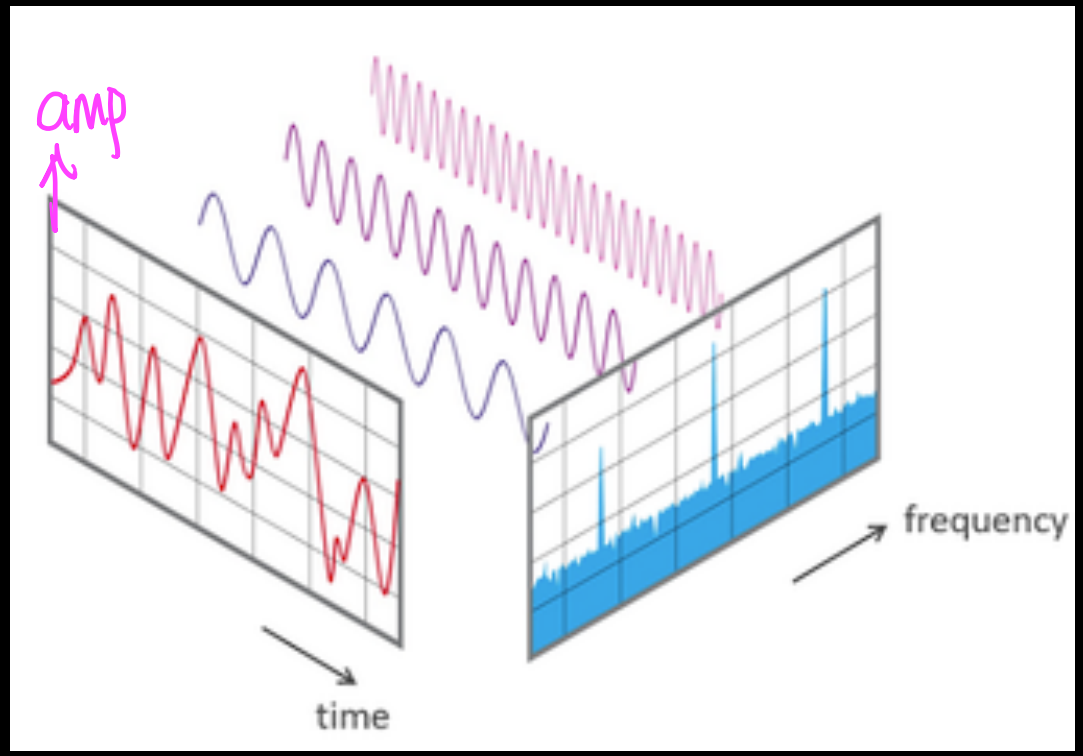
⇓

Transfer Learning is possible

Audio - Image:

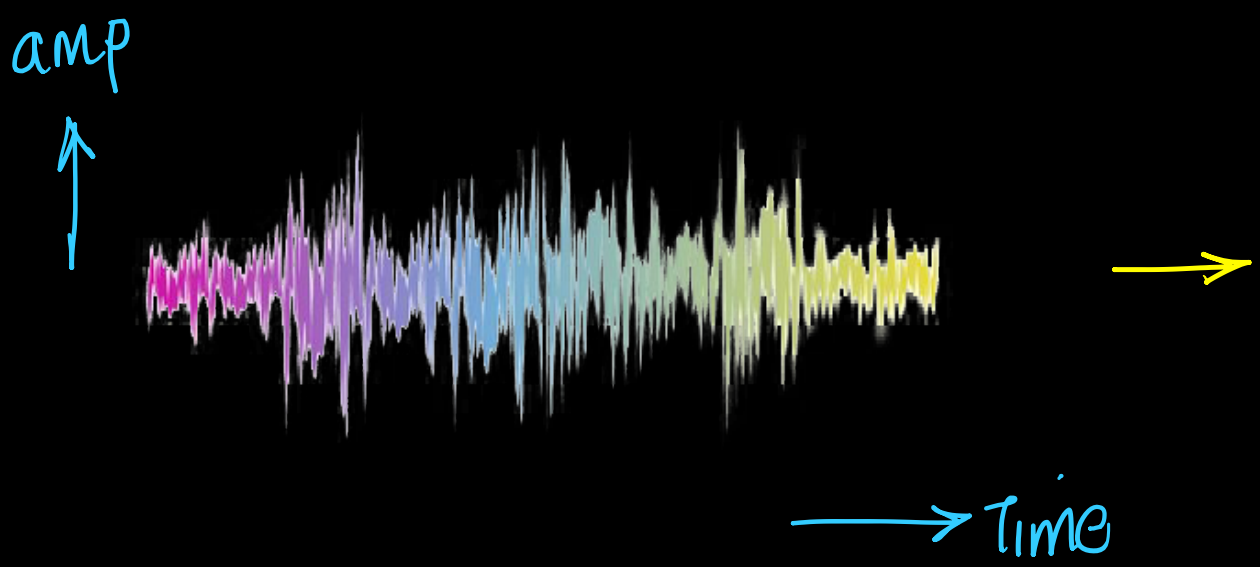
1. 

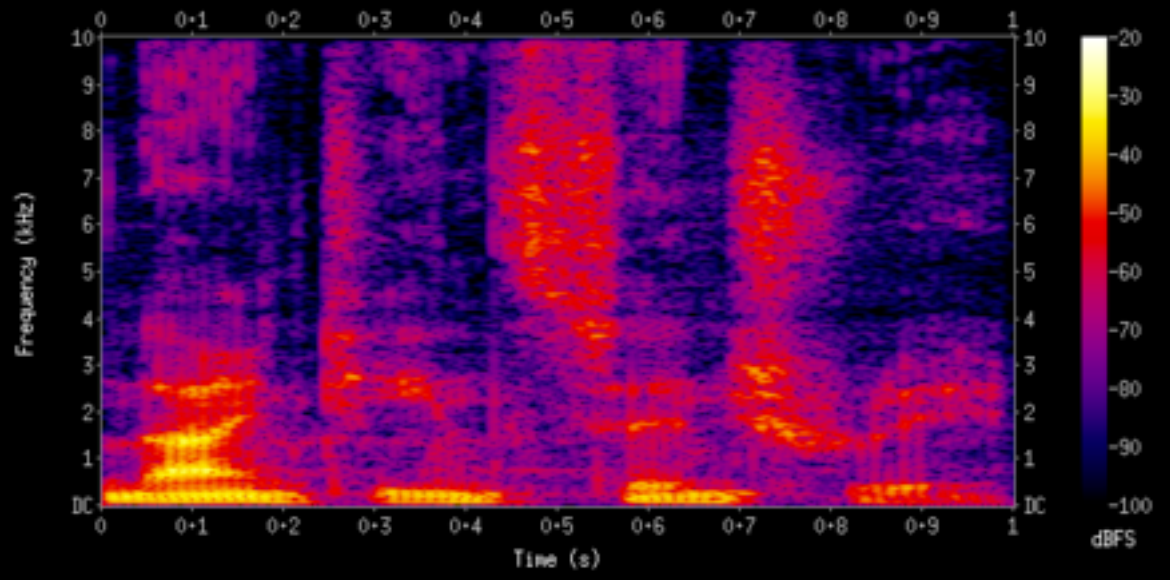
2. Fourier-decomposition (recap)



Audio - Image:

(3) Spectrogram





APPLIED COURSE

Audio - image (code)

$ sox abc.wav -n spectrogram -o abc.png

http://sox.sourceforge.net/

Branch: master ▾ **voice-classification** / scripts / **convert_wav2spect.sh**

Find file | Copy path

hamzag95 wac2spect — 7c71af0 on 7 Dec 2017

1 contributor

10 lines (9 sloc) | 240 Bytes — Raw | Blame | History

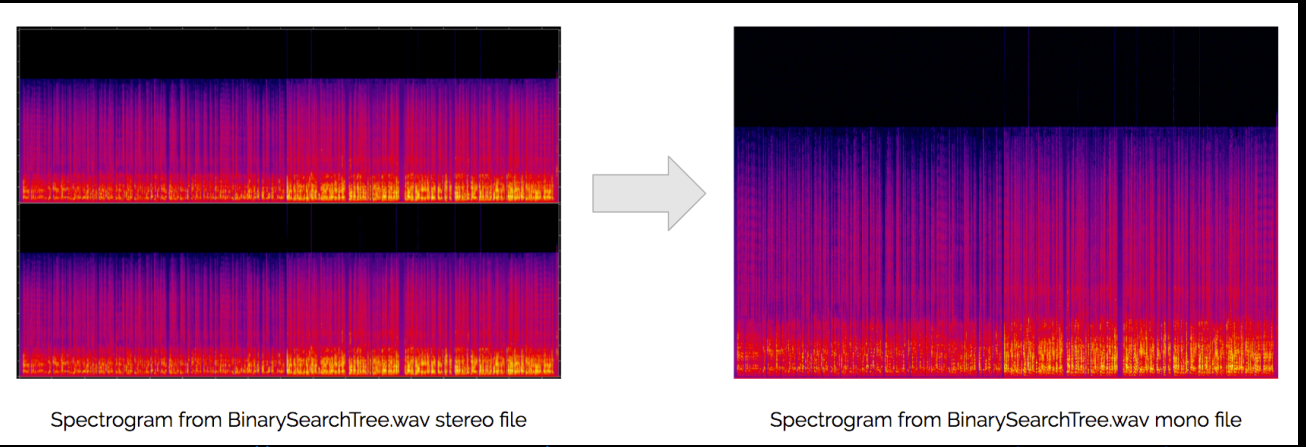
```
1  #!/bin/bash
2
3  for file in *.wav
4  do
5      outfile=${file%.*}
6          sox "$file" ${outfile}.l.wav remix 1
7          sox "$file" ${outfile}.r.wav remix 2
8      sox -m ${outfile}.l.wav ${outfile}.r.wav "$file"
9          sox "$file" -n spectrogram -r -o ${outfile}.png
10 done
```

} → dual-mono (discussed next)

spectrogram

Implementation - Specifics

→ dual-channel to mono-audio



Spectrogram from BinarySearchTree.wav stereo file

Spectrogram from BinarySearchTree.wav mono file

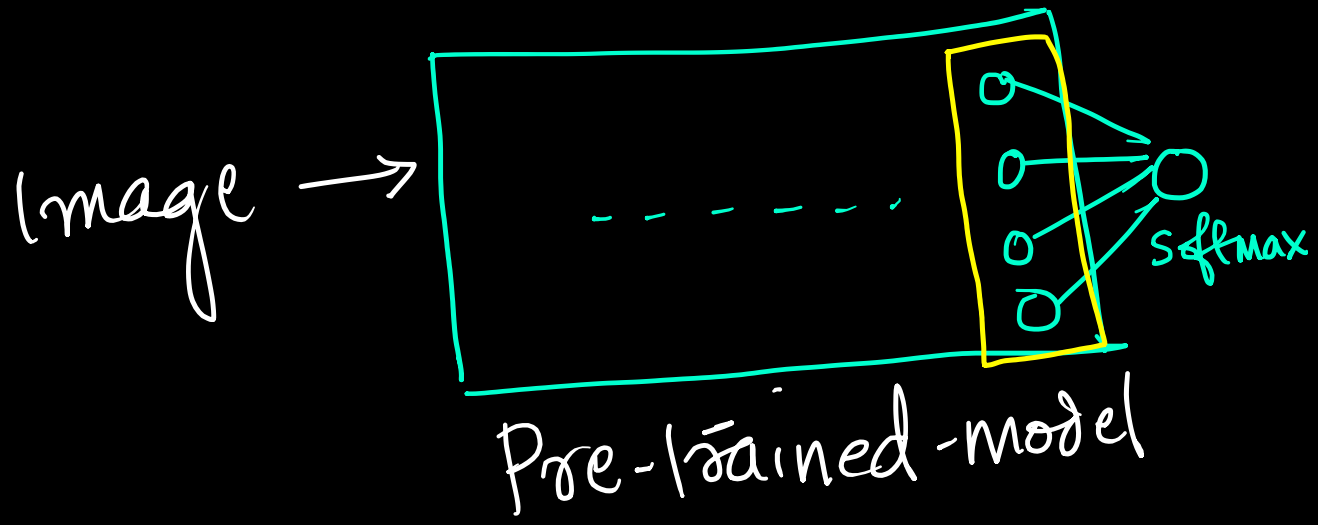
https://towardsdatascience.com/automatic-speaker-recognition-using-transfer-learning-6fab63e34e74

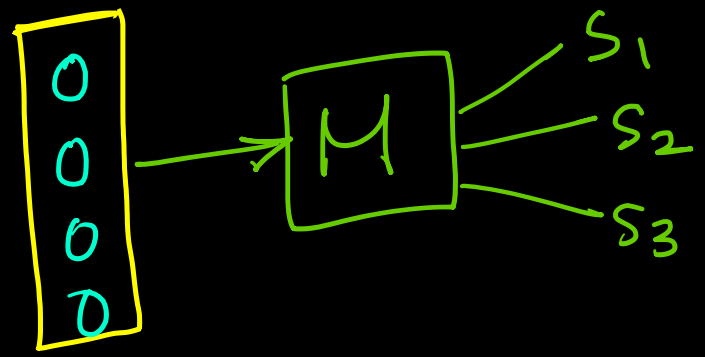
→ cut/trim the input audio into smaller chunks.

$ sox file_in.mp3 file_out.mp3 trim START LENGTH

Transfer Learning:

(recap)





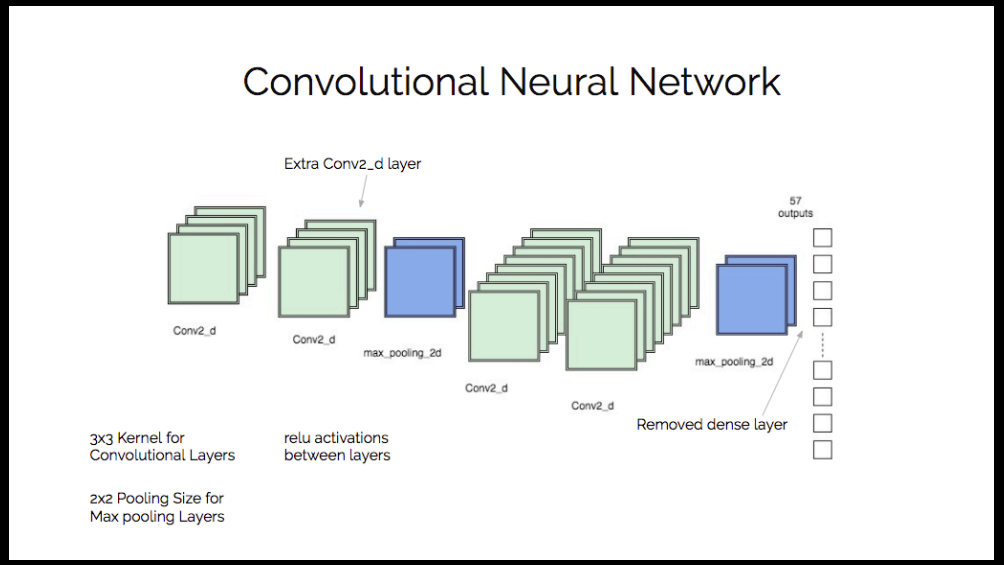
Issue: ImageNet-pretrained models

https://github.com/hamzag95/voice-classification

- Uses CIFAR-10 arch
- removed dense layer
- 40,000 dim-vector

↓

SVM + RBF kernel



retrained the model with 57 speakers' spectrograms for transfer learning

Performance:

① 35 sec per Speaker : Train (5sec x 7)

Speakers = 3

Test-perf on 35sec: 95% accuracy

② 15 Sec per Speaker: 83% accuracy

Performance:

③ language - neutral

④ gender - sensitive

⑤ # speakers increases ⇒ performance drops

Can we do better with few-samples?

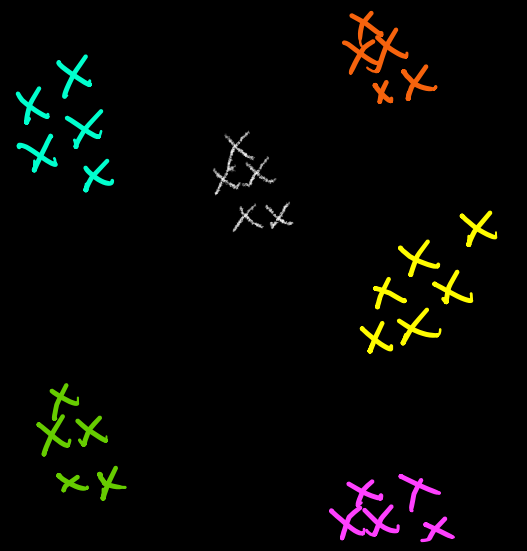
few-shot learning:

s_1 : $a_{11}, a_{12}, a_{13}, a_{14}, a_{15}$

s_2 : $a_{21}, a_{22}, a_{23}, a_{24}, a_{25}$

$\vdots$ $\qquad\qquad$ $\vdots$

s_6 : $a_{61}, a_{62}, \cdots a_{65}$

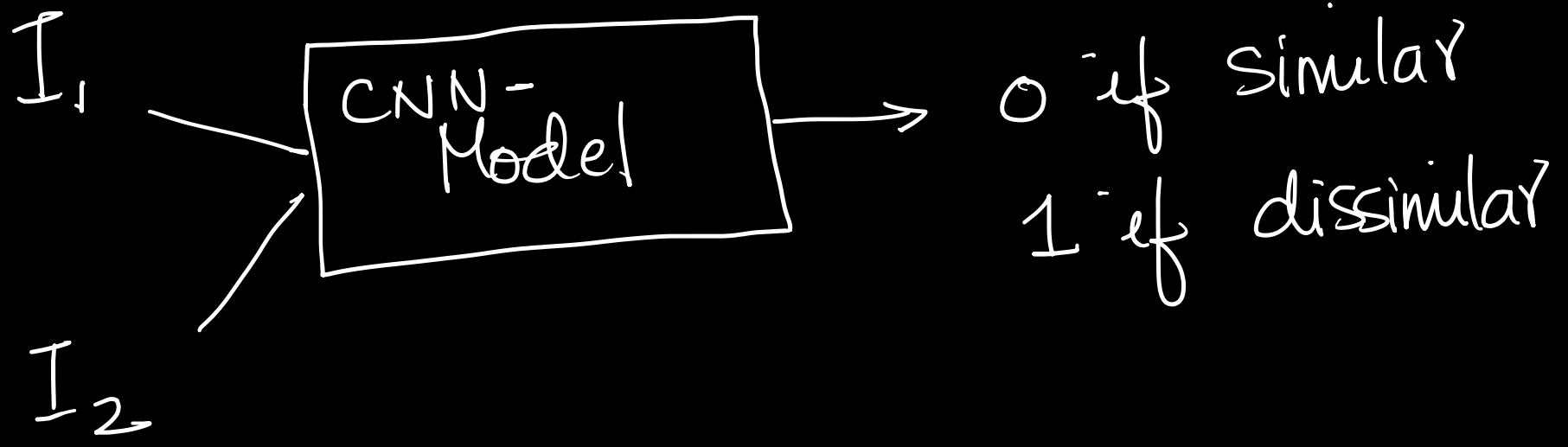


6-Speakers & 5 samples/speaker

similar - pairs: ——→ $^5C_2 \times 6 = 60$

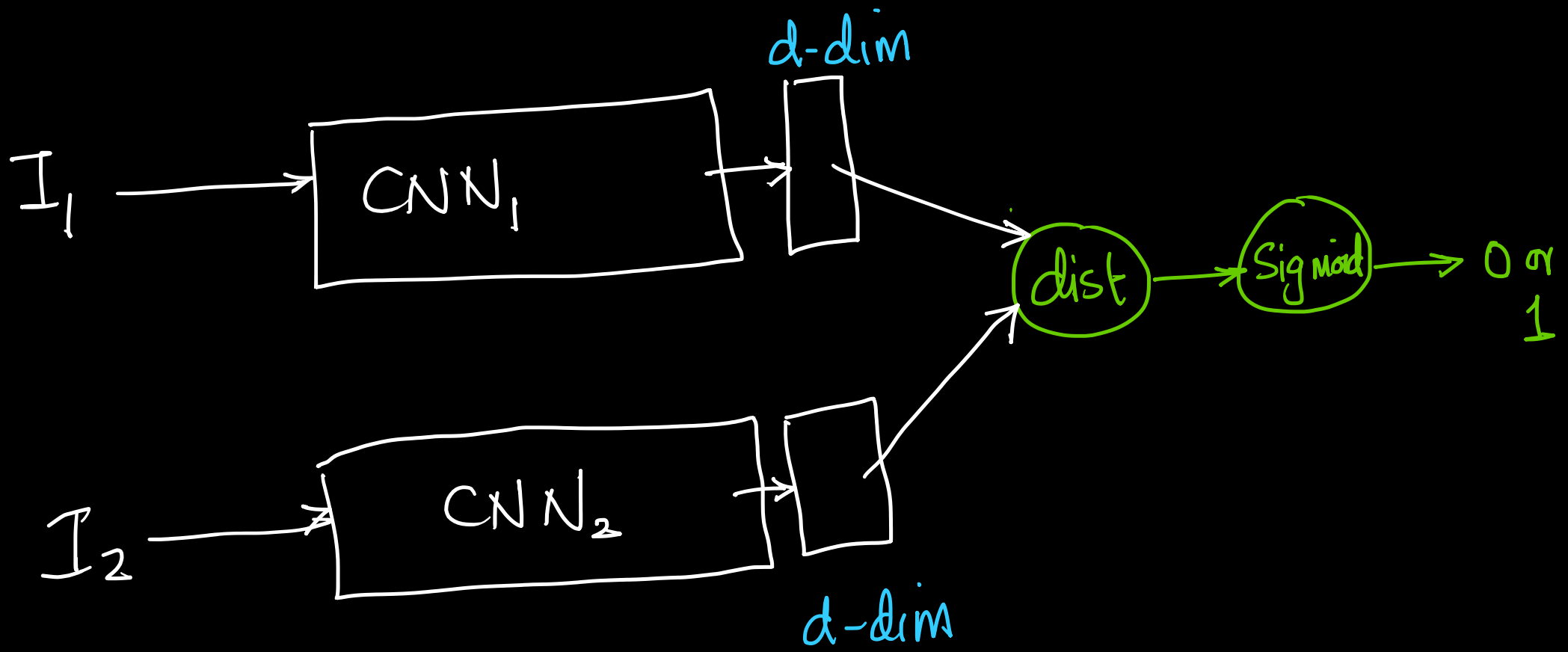
dissimilar pairs: ——→ $^6C_2 \times 5 \times 5 = 375$

Idea:



{ similar ⇒ closer in the d-dim space

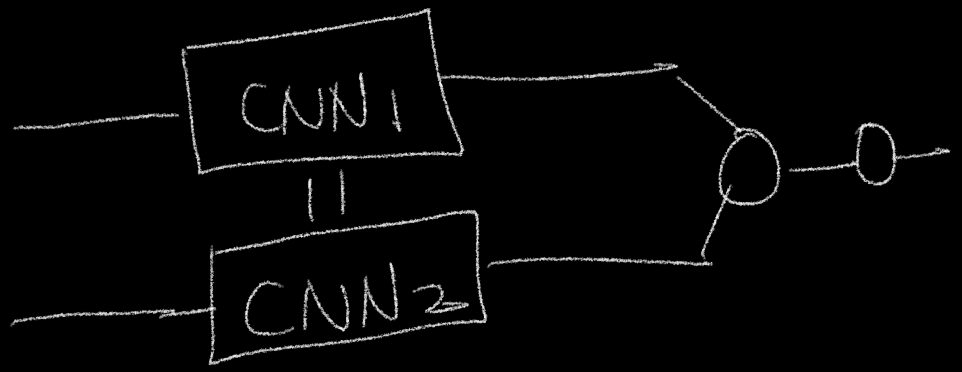
dissimilar ⇒ farther

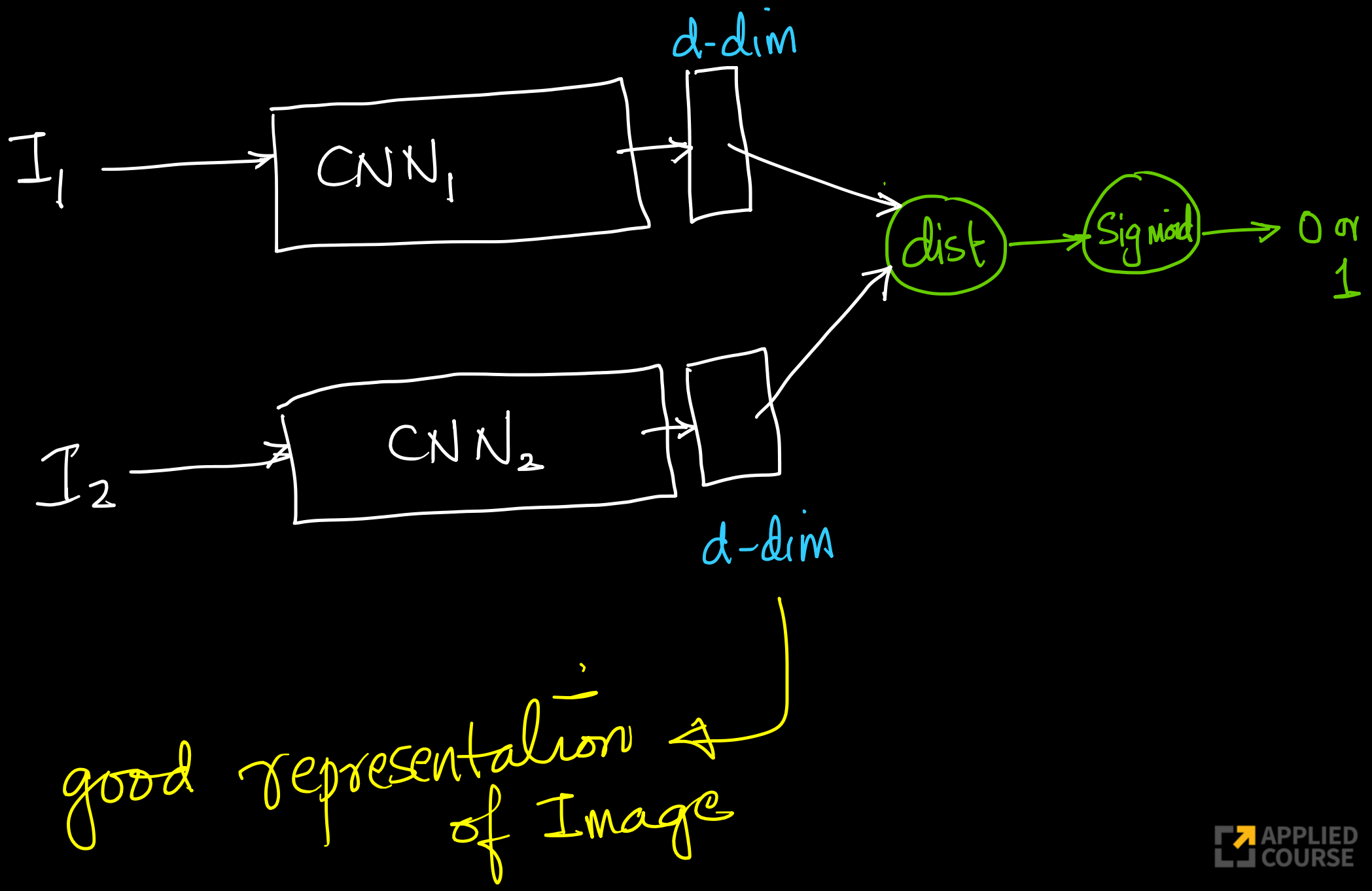


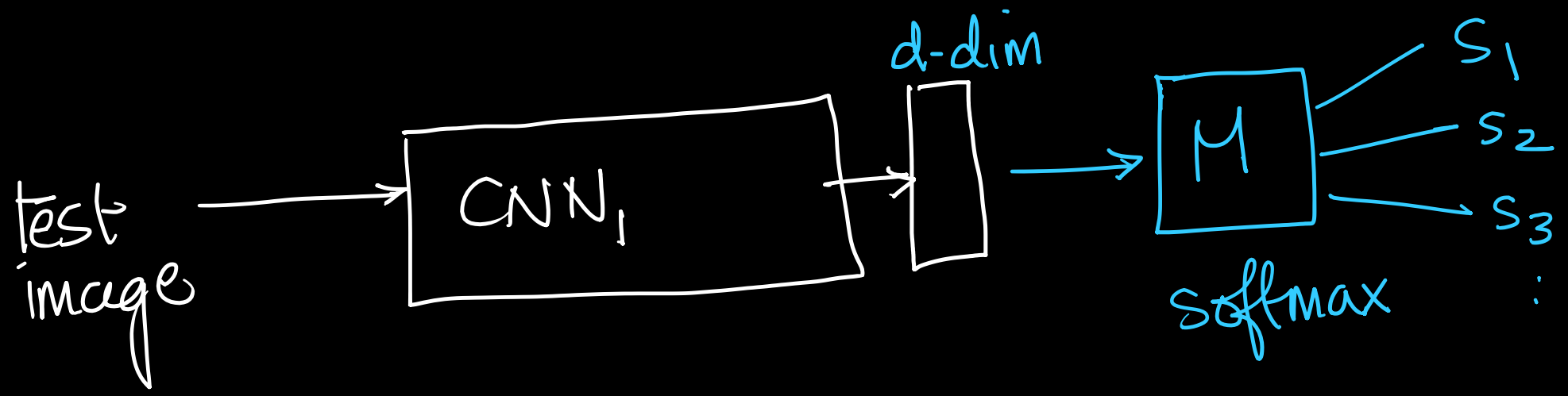
$CNN_1 = CNN_2$ (same weights & arch)

Siamese - twins

(a.k.a conjoined twins)




d-dim
I_1
CNN_1
dist
Sigmoid
0 or 1
I_2
CNN_2
d-dim
good representation of Image



Code! https://github.com/oscarknagg/voicemap

1. https://github.com/oscarknagg/voicemap/blob/master/voicemap/models.py

   def get_baseline_convolutional_encoder ……

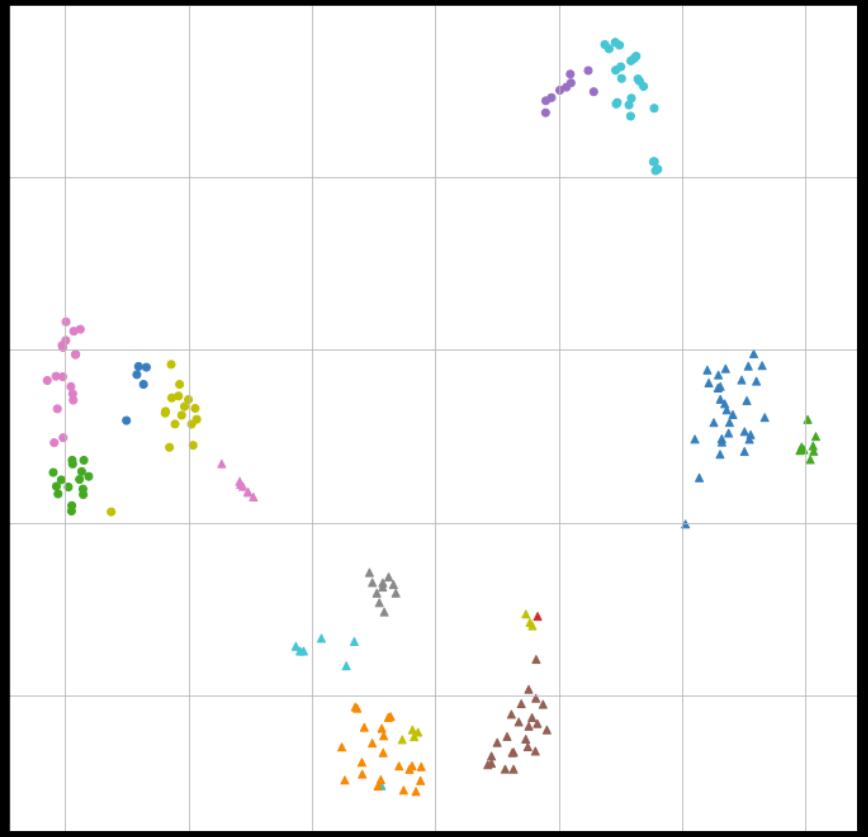
   def build_siamese_net ……..

2. https://github.com/oscarknagg/voicemap/blob/master/experiments/train_siamese.py

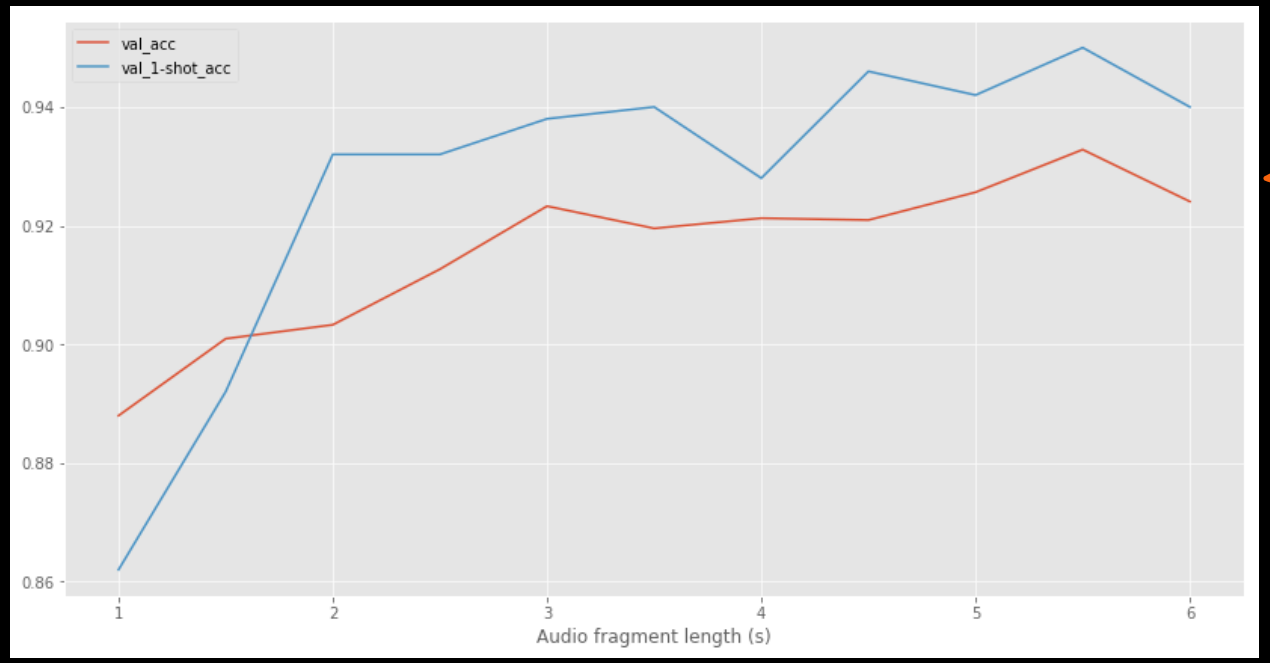
Performance:

https://github.com/oscarknagg/voicemap/blob/master/notebooks/Embedding_Space_Visualisation.ipynb

t-SNE →



● : male
▲ : female



← Audio-length vs ACC

As #Speakers increase, performance drops.

2 → 96% acc

5 → 92% acc

7 → 87%

10 → 85%

NOTE: Transfer learning & few-shot learning

can be used for many similar tasks

e.g: Face-recognition for attendance/ login