Speaking Avatar

HiWi Project – Frank Fundel

Inhalt

1. The Idea

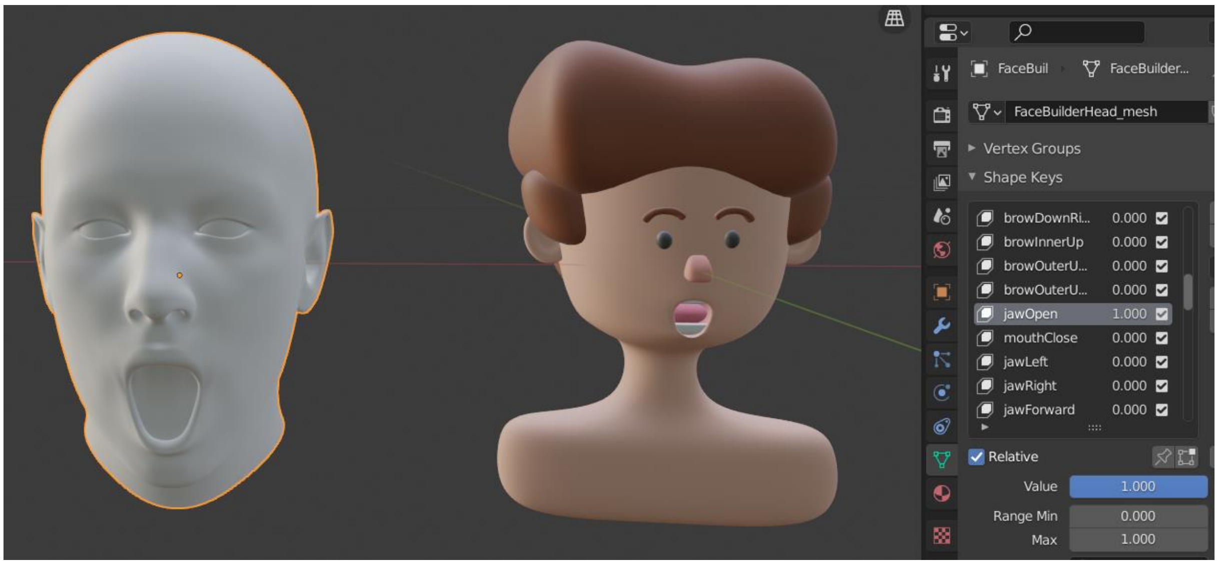
Use facial tracking data from an iPhone X in combination with speech recordings to generate face movement, especially mouth movement of a 3D avatar from pure audio.

2. The Model

I created some 3D models in blender that can be used as avatar. The first one I tried was using images of myself and recreating me in 3D (which looked horrifying).

After that I watched several tutorials on how to create a friendly looking avatar and built the following one which was certainly much better.

The next step was to create the ShapeKeys for the BlendShapes. The BlendShapes are the attributes you get from recording facial expressions using Apples Face Tracking API (ARKit). So I had to animate all 61 BlendShapes by hand into the ShapeKeys which are the shapes the model moves according to certain variables i.g. the BlendShape attributes.



browDownRi...	0.000	✓
browInnerUp	0.000	✓
browOuterU...	0.000	✓
browOuterU...	0.000	✓
jawOpen	1.000	✓
mouthClose	0.000	✓
jawLeft	0.000	✓
jawRight	0.000	✓
jawForward	0.000	✓

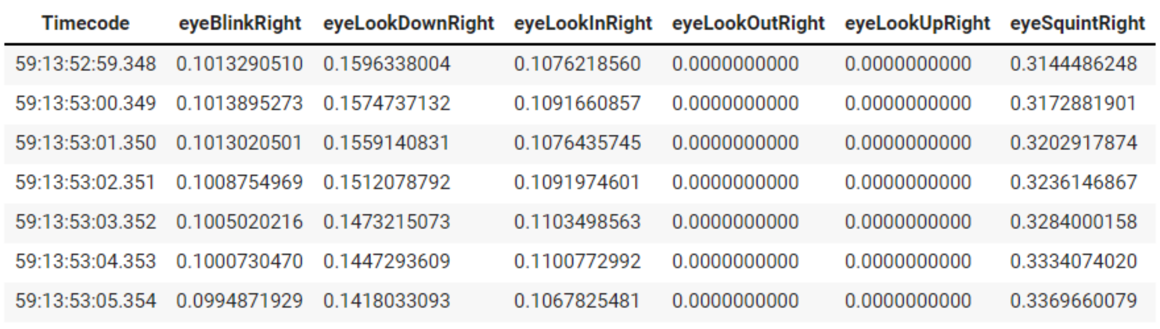
✓ Relative

Value 1.000

Range Min 0.000

Max 1.000

The attributes are in between the range of 0 and 1 and look like this after recording (stored in a CSV):



Timecode	eyeBlinkRight	eyeLookDownRight	eyeLookInRight	eyeLookOutRight	eyeLookUpRight	eyeSquintRight
59:13:52:59.348	0.1013290510	0.1596338004	0.1076218560	0.0000000000	0.0000000000	0.3144486248
59:13:53:00.349	0.1013895273	0.1574737132	0.1091660857	0.0000000000	0.0000000000	0.3172881901
59:13:53:01.350	0.1013020501	0.1559140831	0.1076435745	0.0000000000	0.0000000000	0.3202917874
59:13:53:02.351	0.1008754969	0.1512078792	0.1091974601	0.0000000000	0.0000000000	0.3236146867
59:13:53:03.352	0.1005020216	0.1473215073	0.1103498563	0.0000000000	0.0000000000	0.3284000158
59:13:53:04.353	0.1000730470	0.1447293609	0.1100772992	0.0000000000	0.0000000000	0.3334074020
59:13:53:05.354	0.0994871929	0.1418033093	0.1067825481	0.0000000000	0.0000000000	0.3369660079

I decided that we actually don't need all 61 attributes but just 23:

blendshapes = ["jawOpen", "mouthClose", "mouthFunnel", "mouthPucker", "mouthRight", "mouthLeft", "mouthSmileRight", "mouthSmileLeft", "mouthFrownRight", "mouthFrownLeft", "mouthDimpleRight", "mouthDimpleLeft", "mouthStretchRight", "mouthStretchLeft", "mouthRollLower", "mouthRollUpper", "mouthShrugLower", "mouthShrugUpper", "mouthPressRight", "mouthPressLeft", "mouthLowerDownRight", "mouthLowerDownLeft", "mouthUpperUpRight"]

First we need to prepare the raw data. Because the frame rate is not consistent when recording facial data, we split up the data into 1 second blocks of varying sizes using the timecodes. Then we lower the FPS to 24 FPS for all blocks e.g. take 24 evenly spaced items out of each block. Then we do some more magic so the data can be read the right way and done.

Now we want to apply the facial data to our model. There was no existing script oder plugin for Blender that could do that so I created one.

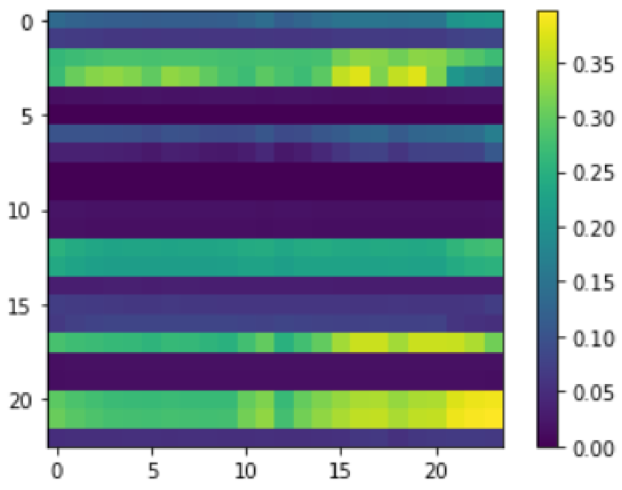
Perfect. So now we can gather some training data.

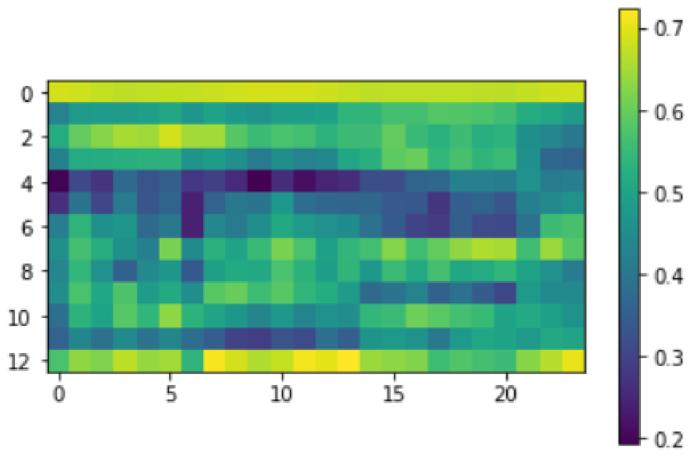
3. Training Data

I was reading hours stories and wikipedia articles out loud into the front camera of my phone using the Live Link Face App from Unreal Engine which records sound and facial data. After preparing (24 FPS and magic) I loaded each CSV into Blender on my model and adjusted the sound to the animation because most of the time the alignment was off. After that the data was almost ready to use, I now chopped the face and audio data into pieces. Because speech sounds do not occur in isolation and do not have a one-to-one mapping to characters, we can capture the effects of coarticulation (the articulation of one sound influencing the articulation of another). By training the network on overlapping windows of audio data that captures sound from before and after the current time index. So let's concatenate all data and slide a window over it to generate 6 frames (0.25 sec) long samples.

"Depending on the data sampling rate, we recommend 26 cepstral features for 16,000 Hz and 13 cepstral features for 8,000 hz.“ - https://zhuanlan.zhihu.com/p/28274740

So for each audio peace we extract the 13 cepstral features (I manually lowered the sampling rate off the audio to 8000Hz). Then we select the 23 facial features and voilà. Here is a visual representation off the data (facial features on the top, audio features on the bottom):





For each network we try the audio features will be the input and the facial features will be the desired output.

4. The Training

I had three different network models in mind which we will try:

1. First we try a complete encoder-decoder convolutional neural network, we could also try Frequency to Time convolution
2. Then we will try a complete sequence to sequence recurrent neural network, with LSTM or GRU
3. Transformer-Network

Hyperparamters to try:

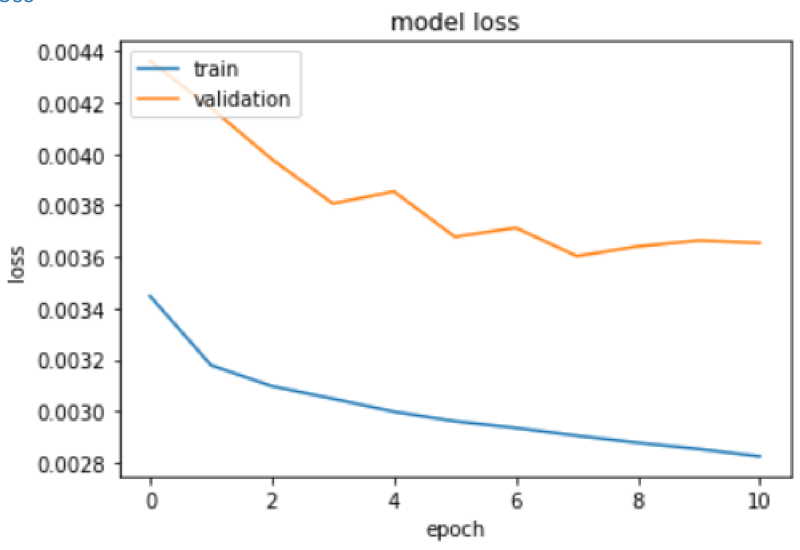
- Different sizes
- Different pooling
- Dropout
- BatchNorm

1. CNN

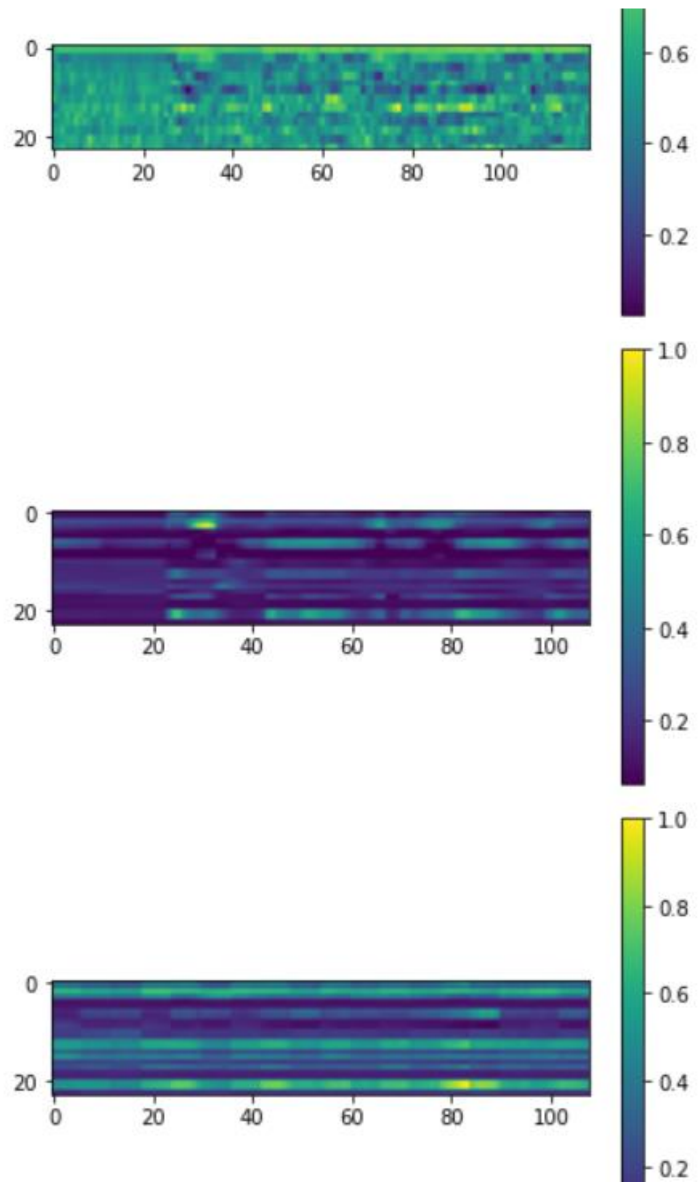
Model summary

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 24, 13, 16)        640
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 12, 13, 16)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 12, 13, 32)        20000
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 6, 13, 32)         0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 6, 13, 64)         79936
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 6, 6, 64)          73792
_________________________________________________________________
dropout (Dropout)            (None, 6, 6, 64)          0
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 6, 3, 64)          0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 6, 3, 64)          36928
_________________________________________________________________
dropout_1 (Dropout)          (None, 6, 3, 64)          0
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 6, 1, 64)          0
_________________________________________________________________
reshape_1 (Reshape)          (None, 6, 64)             0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 6, 23)             8855
=================================================================
```

Loss



Performance (input audio features on top, ground truth facial features in the middle and predicted facial features on the bottom)



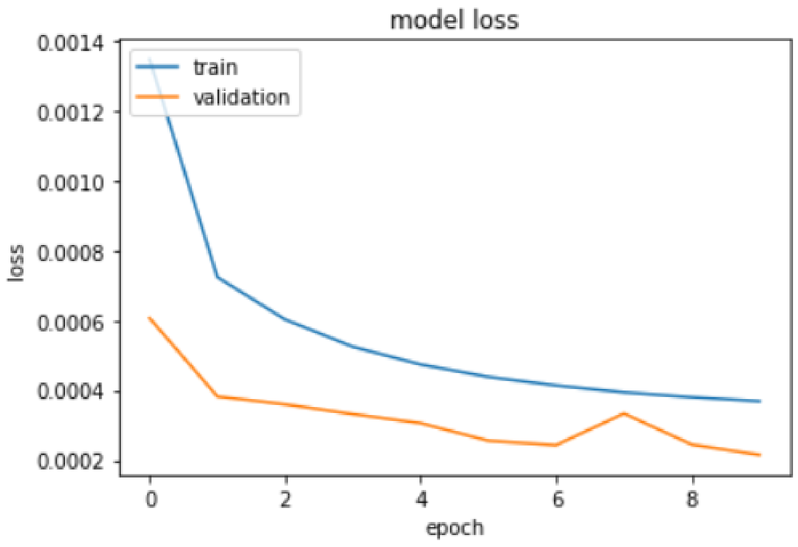
The results are „Okay“, we can see some understanding.

2. RNN

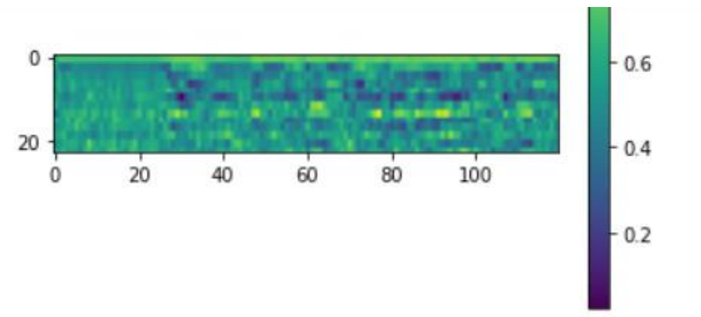
Model summary

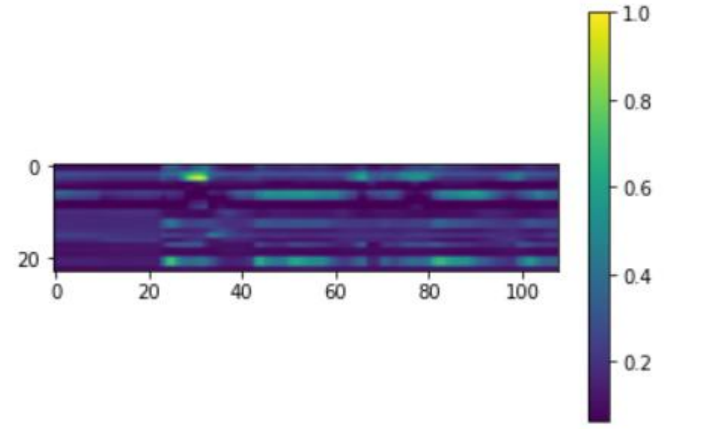
```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_3 (InputLayer)            [(None, None, 13)]   0
__________________________________________________________________________________________________
input_4 (InputLayer)            [(None, None, 23)]   0
__________________________________________________________________________________________________
lstm_2 (LSTM)                   [(None, 256), (None, 276480      input_3[0][0]
__________________________________________________________________________________________________
lstm_3 (LSTM)                   [(None, None, 256),  286720      input_4[0][0]
                                                                 lstm_2[0][1]
                                                                 lstm_2[0][2]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, None, 23)     5911        lstm_3[0][0]
==================================================================================================
```

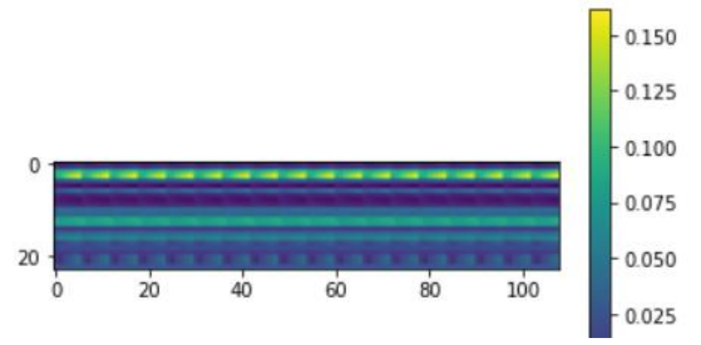
Loss



Performance







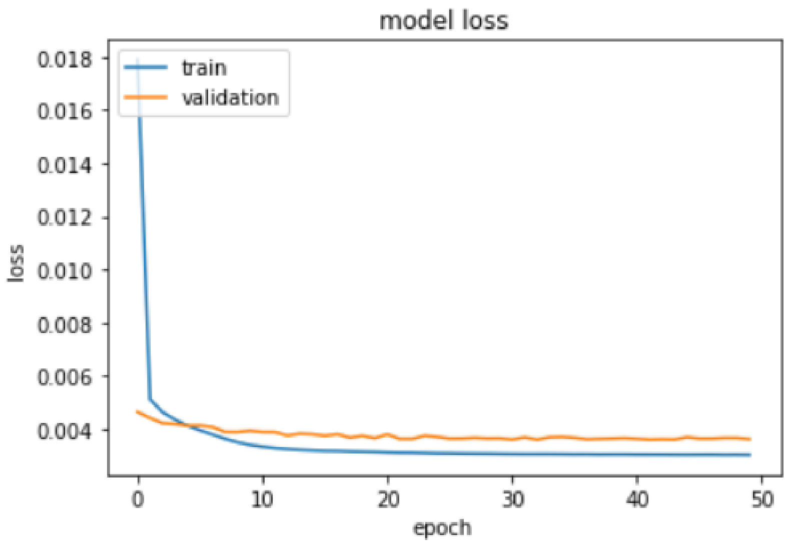
Either some error lies in here or this model is just not made for this kind of problems.

3. Transformer Network

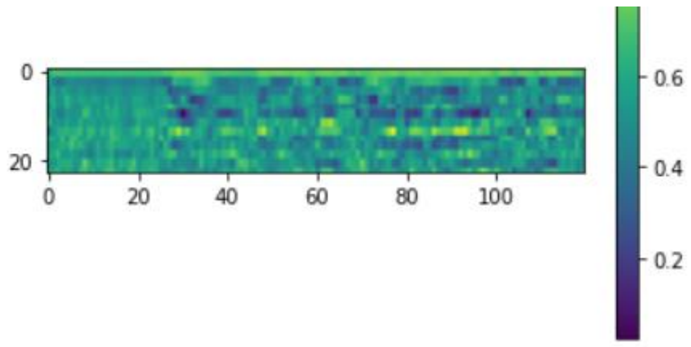
Model summary

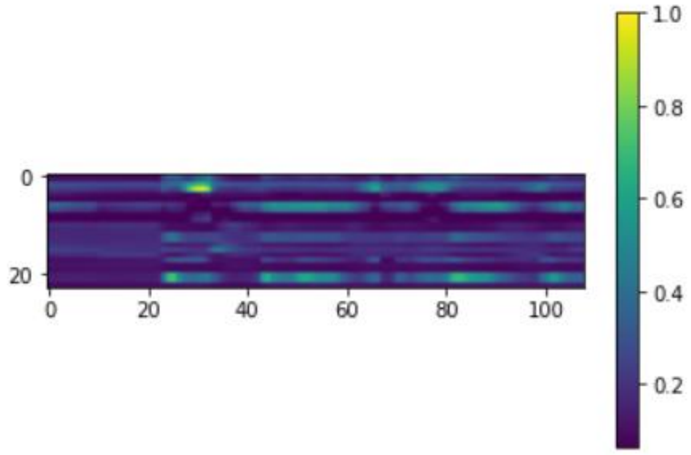
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 24, 13)]          0
_________________________________________________________________
transformer_block (Transform (None, 24, 13)            2372
_________________________________________________________________
global_average_pooling1d (Gl (None, 13)                0
_________________________________________________________________
dropout_2 (Dropout)          (None, 13)                0
_________________________________________________________________
dense_2 (Dense)              (None, 138)               1932
_________________________________________________________________
reshape (Reshape)            (None, 6, 23)             0
=================================================================
```

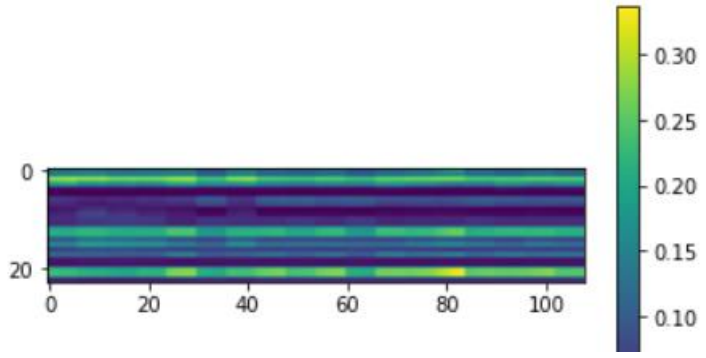
Loss



Performance







Results are comparable to CNN.

4. Audio Transcription

One more method came to my mind: Use audio transcription and forced alignment to get the timestamps of each word, syllable or phoneme and then match them to the corresponding animation. We could either use an existing transcription of an audio or use speech recognition to align the audio.

5. Application

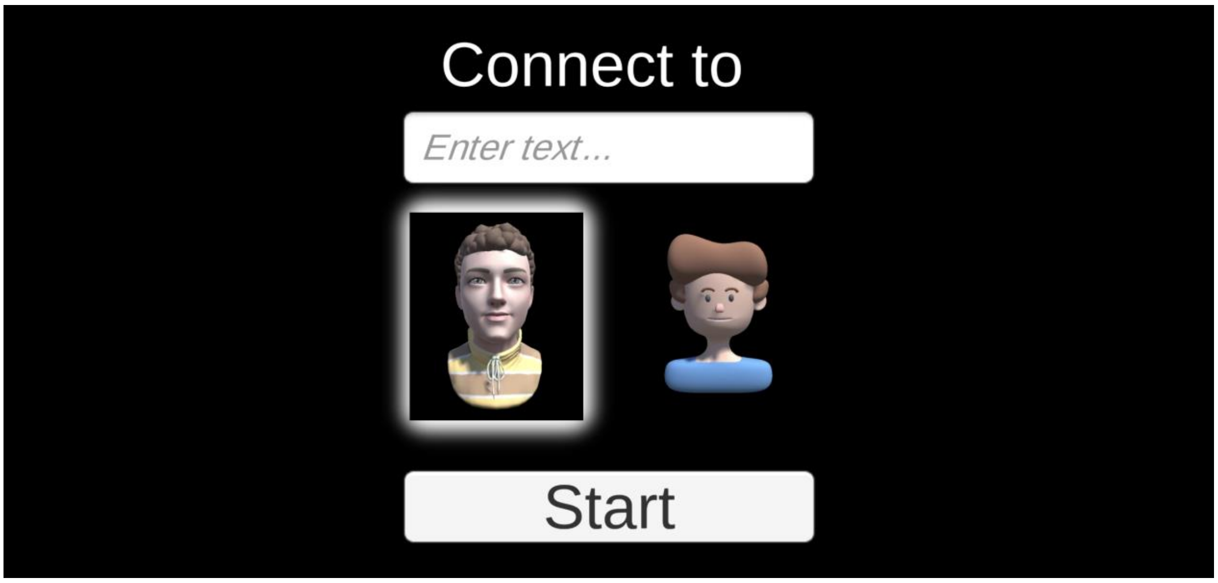
Next I copied the CSV to Animation Script and modified it to read .WAV files and run it through the saved neural network models to apply the predicted animations to the Blender model as before. I chose the CNN one because the results applied to the Blender model looked better than the Transformer ones.

Next was the last problem: display the animation on the Holofil. Luckily there is an App for the Holofil that can display 3D-Files. But when I export them from Blender they don't work, I spent days trying to resolve this problem efficiently but could not find a solution. So next best thing would be: Live stream it from the PC to the Holofil using TeamViewer or using a secondary display via HDMI. Or you could render the animation and play it but that would take too long.

So I figured out we have to make a custom Unity App for Android, because the Holofil App cannot play sound and rendering takes too long and is not movable in 3D space. I could easily set up a Unity App on android and import the Blender model with all its BlendShapes.

Now all that is needed is a Script that converts the neural network output into an animation. Luckily I found exactly that on the internet. Now you can enter text, it gets sent to a server which runs it through text-to-speech, prepares it and then runs it through the neural network to send back the keyframe data and the audio.

We then added eye tracking and remote control support for a better experience.


Connect to
Enter text...
Start