DT2119 - Speech-To-Animation project

Berkan Yapici, Romain Darous
KTH, Royal Institute of Technology
May 20th, 2024

Abstract

The present study investigated automated modeling of 3D Avatar faces using only audio input. Various applications can be envisioned for 3D model animations (cinema, animated movies, video games, virtual reality...) at a lower cost compared to motion capture. In this study, we trained the SAiD model [4] to perform speech-driven face animation on Unreal Engine MetaHumans. While our results did not match the performance of the SAiD model, we were still able to generate usable animations that exhibited some relevant articulation patterns.

1 Introduction

When it comes to 3D face animation, the most realistic technique is motion capture [4]. Sensors are put on an actor, its facial expressions (also called blendshapes) are recorded and plugged into the 3D model face.

Trying to use only speech as input data to perform realistic animation of a 3D face has challenged the research community since the 70's [1]. In 2005 [1] was developped a whole pipeline to animate a 3D model of the face. The system captures detailed 3D face dynamics of speaking actors, which are then represented using Independent Component Analysis (ICA) to output blendshapes among a 'Viseme Space' (i.e. a list of blendshapes). This 3D model allows for realistic speech animation by replicating learned visemes (also called blendshapes) and adding coarticulation effects.

However, further steps have been taken in automating the process of capturing 3D face dynamics, using Deep Learning methods. It is now possible to get those blendshapes only using 2D videos recordings (instead of motion capture, which is more time consuming and expensive [4]). It makes the building of bigger datasets quicker and cheaper. In [2] is built a MLP classifier that maps phoneme labels and corresponding blendshapes. It uses a context window to capture localized context and coarticulation effects.

The FaceFormer model [3], more recent, suggests a Transformer based autoregressive approach. Using mulit-head attention mechanism to encode a long-term audio context, the model could lead to a significant increase of the performance regarding facial animation.

The main asset of those two models is that, compared to the pipeline suggested in the paper [1], the algorithm is performing a regression on the blendshape features. The pipeline from [1] consisted in classifying the speech input between different 'Visemes', which were sequences of pre-determined blendshapes.

However, our attention was drawn to the SAiD model [4], published in 2023. Combining state-of-the-art audio feature extraction and a conditional diffusion model to generate frame-per-frame blendshapes, the results were stunning.

We will use this last paper and the provided code as a baseline to train our speech-to-animation model.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

2 Method

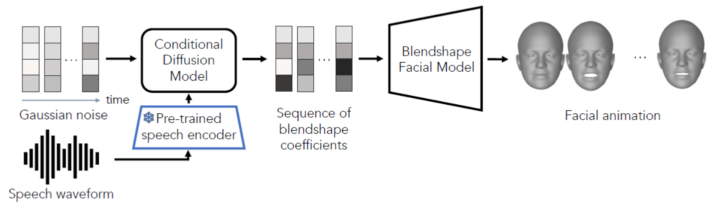

Figure 1: SaiD model pipeline, from the original paper [4].

Using the SAiD model as a baseline, we want to train it to build a pipeline from speech to MetaHuman avatars face animation from Unreal Engine. Those are very high quality avatars that require a high number of blendshape features.

2.1 Recording the data

One goal of the project was to record a personal dataset to train the model. Given the limited amount of time we had to build our training dataset, we decided to restrict it to a single speaker to try to achieve better results.

For this dataset we need needed audio input mapped to blendshape features. Live Link Face is an app that allows recording blendshapes from video, directly formatted as required to perform MetaHuman animation in Unreal Engine. The blendshapes are output as a .csv file (see Figure 2). We decided to use this efficient tool to build a ready-to-train dataset.

In total, we recorded around 2 hours and 30 minutes of videos, capturing blendshapes at a rate of 30 fps. In the original paper, they captured the blendshapes at a frequency of 60 fps but had less data. We decided to halve the frame rate, as we believed it would still capture enough information while reducing storage requirements and providing a broader range of facial animations thanks to the larger dataset.

We also paid attention to recording videos that covered a broad range of English phonetics and emotions. Here is the content used to record the data:

- Roald Dahl, *The Enormous Crocodile*,
- Roald Dahl, *The Landlady*,
- ImprovClub KTH, *Kill Karl* (script from the Spex show),
- Gerard Nolst Trenité, *The Chaos Poem*,
- Blackalicious, *Alphabet Aerobics*.

Some natural speaking samples were also recorded (without reading anything).

Timecode	BlendshapeCount	EyeBlinkLeft	EyeLookDownLeft	EyeLookInLeft	EyeLookOutLeft	EyeLookUpLeft	EyeSquintLeft	EyeWideLeft	EyeBlinkRight
11:15:14:06.938	61	0.08502109	0.1072044	0	0	0	0.008252895	0	0.085056804
11:15:14:07.938	61	0.08610896	0.107748941	0	0	0	0.008345387	0	0.086144663
11:15:14:08.938	61	0.08697998	0.108231425	0	0	0	0.008428166	0	0.08701545
11:15:14:09.938	61	0.08773835	0.10915143	0	0	0	0.008477722	0	0.087773196
11:15:14:10.938	61	0.08847918	0.111107886	0	0	0	0.008451747	0	0.088513576
11:15:14:11.939	61	0.08915828	0.112675503	0	0	0	0.008426413	0	0.089192167
11:15:14:12.939	61	0.08979601	0.114708401	0	0	0	0.00837478	0	0.089829229
11:15:14:13.939	61	0.09053952	0.116726533	0	0	0	0.008362645	0	0.090572536
11:15:14:14.939	61	0.0913085	0.117471613	0	0	0	0.00841231	0	0.091341563
11:15:14:15.939	61	0.09454318	0.118527032	0	0	0	0.008581819	0	0.09457805
11:15:14:16.940	61	0.10106532	0.119813569	0	0	0	0.008849004	0	0.101101331

Figure 2: Example of a blendshape file captured using Live Link Face. Each blendshape has a Timecode and its feature for the given time it has been recorded.

2.2 Sound pre-processing

The audio signal (speech waveform) is processed using a pre-trained Wav2Vec 2.0 model to extract audio features. This model captures the essential characteristics of the audio and converts it into a format suitable for the animation model.

Then, to synchronize the audio features with the blendshape coefficients, linear interpolation is used to ensure that the number of audio frames matches the number of blendshape frames.

2.3 Blendshape pre-processing

The data was recorded in sequences of 10 minutes using Live Link Face. Then, to train our model, we needed to split these recordings to perform small-batch training with a reasonable data size. This part was challenging, as we had to synchronize the split audio with the corresponding blendshape coefficients.

We had to deal with incomplete recordings (some frames were sometimes missing, and we had sequences with 27 blendshapes per second, for instance). This made the synchronization between blendshapes and audio files more difficult. To tackle this problem, we went through the following steps for each recording:

- Splitting the audio file into sequences of 10 seconds each,
- Processing the blendshape file sequentially,
- Using the time code of the file to take the next blendshapes that would correspond to 10 seconds as closely as possible,
- Adding the time difference between the audio duration and the blendshape equivalent duration to a variable that stores accumulation error,
- When the accumulation error exceeds two-thirds of a frame duration (around 20 ms), taking the blendshapes of the next samples by going one frame back in time.

We verified that the approach was working by plugging the samples and their blendshapes into Unreal Engine and checking if the lips were synced to the audio. We achieved satisfying lip synchronization.

Regarding feature processing in the SaiD model [4], an important part of the work involves computing the frame-by-frame blendshape coefficients. In our case, these coefficients were directly provided by the application Live Link Face, which supplied the relevant features to perform face animation in Unreal Engine directly. We decided not to preprocess the blendshapes and to keep all the features recorded by the app (in the SAiD model, they remove some of the features to improve model stability).

Hence, we had blendshapes with 61 features each, which is almost twice as many as in the original paper [4] (31 features per blendshape).

2.4 Speech to Animation model

The SaiD model uses a simplified version of the Conditional UNet architecture from OpenAI, adapted for 1D input. It acts as a denoising autoencoder. This model predicts and removes noise from the noisy blendshape coefficient sequence conditioned on the speech waveform.

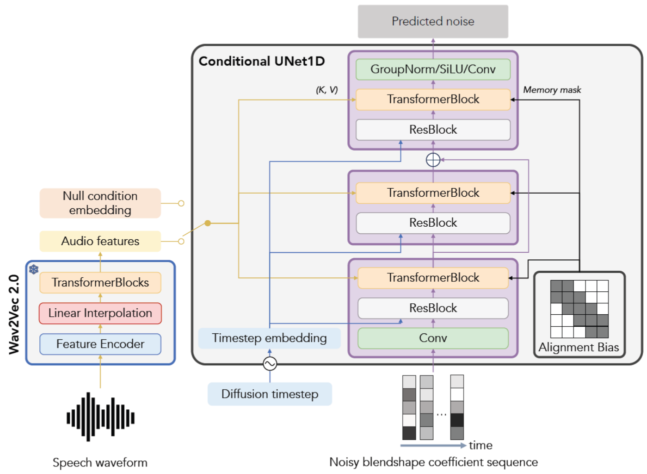

Figure 3: SaiD architecture, from the original paper [4].

The denoiser model is composed of one encoder block, one middle block and one decoder block. The downsampling and upsampling layers are removed (present in the original model from OpenAI). Diffusion timestep is converted into the sinusoidal embedding and then becomes the input of each residual block in the denoiser.

Regarding the learnt features of the speech input become the key and values matrices of the cross-attention layer in the denoiser. The SAiD models also implmements alignment bias as a memory mask for the cross-attention layer to enhance the alignment between the speech and blendshape feature sequence.

2.5 Training

The model is trained using a loss function that minimizes the absolute error between the predicted noise and the actual noise at each diffusion timestep. Additionally, a noise-level velocity loss is introduced to maintain smooth transitions in the animation.

To train the model, given its complexity, we used a virtual machine (VM) on the Google Cloud computing service. We ran our model on an NVIDIA L4 GPU.

3 Experiments

We started by testing the pipeline on a small dataset to make sure that we could perform end-to-end speech-to-animation modelling.

4 Training parameters

Then, we ran our model on the whole dataset with the following hyperparameters :

- Number of epochs : 40 000 (50 000 in the SAiD model),
- Number of warm-up epochs : 5 000,
- Losses : Absolute loss, Velocity loss,
- Weight of the velocity loss : 1.0,
- Weight of the absolute loss : 0.2,
- Learning rate : 1e-5,
- Batch size : 8, as in the original paper [4],
- Validation period : 200,

- Frame per seconds : 30,
- Optimizer : Adam,
- Number of (de)noising steps : 1 000

4.1 Inference parameters

To generate an output given an audio input, we have to go through the denoiser in the other direction :

- Number of denoising steps : 1000,
- Frame par seconds : 30,
- Number of repetitions : 144 (originally 72 in the SAiD model [4]).

4.2 Results

- Running time : $\approx$ 30 hours,
- Time for generating a 10 secs output : $\approx$ 15 mins

We observe that the model is converging. The animation of the face appears quite natural, indicating that the model produces realistic animations. However, the synchronization with speech does not match the results of the original SAiD model or those rendered using motion capture. Although there are moments where the articulation aligns with the input audio, the overall animation lacks consistency. For instance, the mouth position during periods of silence is not at rest, which is problematic.

He is a comparison between motion capture, the SAiD model and our model :

- Motion capture example : Digital Domain Motion Capture
- SAiD model original results : SAiD model results
- Our results : Our model results

5 Discussion

5.1 Speed of generation

Diffusion models rely on sequential Markov processes for output generation. This makes the process inherently slow, even for short audio clips, as demonstrated earlier. Powerful machines are necessary to achieve timely results.

Live avatar animation in VR, for instance, would necessitate a faster architecture for it not to rely on motion capture sensors (nor computer vision).

5.2 Effect of the single speaker

The generated expressions exhibited a recognizable similarity, often characterized by exaggerated movements like raised eyebrows, protruding lips, and wide mouth openings. This highlights the importance of speaker diversity for achieving more neutral 3D face animation.

We initially believed that training the model on a single speaker would yield better results for that specific speaker, surpassing the original SAiD model's performance. However, this wasn't the case. While a single speaker might be a contributing factor, it's crucial to explore other potential error sources as detailed below. However, it doesn't mean that having a single speaker is responsible for poor results (on the specific speaker, as other source of errors can be explored, as seen below.

5.3 Possible sources of errors

Identifying definitive improvement strategies for the algorithm is complex. However, several avenues are worth exploring:

- Insufficient training duration: Utilizing more powerful GPUs and extending training times might yield better results. Longer training allows the model to learn more complex relationships between audio and facial expressions.
- **Biased training data**: The dataset may contain inconsistencies, with speakers adopting non-neutral expressions during pauses. While these expressions might be contextually relevant, the model might struggle to interpret the emotions without explicit conditional features.
- **Limited amount of speakers**: Training on a broader range of speakers could lead to more neutral expressions, as opposed to the single-speaker approach. This diversity could improve the model's ability to generalize.
- **Excessive feature number**: We employed twice the number of features compared to the original paper. Reducing the blendshape features or implementing feature extraction techniques like VAEs could be beneficial in future studies. This could lead to a more efficient model with improved performance.

5.4 Conclusion

We proposed our own implementation of the SAiD model, a diffusion-based approach for the speech-driven 3D facial animation problem. Overall, we set up a pipeline to perform end-to-end speech-to-animation using our single-speaker dataset. By utilizing the SAiD model, we were able to generate animations simply by inputting audio sequences into our model. During testing, some local audio sequences were well animated, indicating that the algorithm was learning relevant blendshapes. However, the results need improvement for practical use.

Regarding the use of such a model in the industry, the time required to compute blendshape features given an audio input is too long to be considered a viable solution for live animation. The most reasonable applications would be for animation movies or videos games, where the animation can be done in post-production.
However, its main advantage is the low amount of resources required to generate the animation, as we do not need sensors, cameras, or actors. This approach opens the door to a more flexible technique for 3D face animation.

Acknowledgments

We would like to thank Jonas Berkow, PROF., DEPUTY HEAD OF DIVISION OF SPEECH, MUSIC AND HEARING at KTH and Anna Deichler, PhD student OF DIVISION OF SPEECH, MUSIC AND HEARING at KTH, our supervisors for this project.

References

[1] Muller, P., Kalberer, G.A., Proesmans, M., & Van Gool, L. (2005). Realistic speech animation based on observed 3D face dynamics. IEE Proceedings - Vision Image and Signal Processing, 152(4), 491-500. doi:10.1049/ip-vis:20045112.

[2] Taylor, S., Kim, T., Yue, Y., Mahler, M., Krahe, J., Garcia Rodriguez, A., Hodgins, J., & Matthews, I. (2017). A Deep Learning Approach for Generalized Speech Animation. University of East Anglia, California Institute of Technology, Disney Research, Carnegie Mellon University.

[3] Fan, Y., Lin, Z., Saito, J., Wang, W., & Komura, T. (2022). FaceFormer: Speech-Driven 3D Facial Animation with Transformers. arXiv preprint arXiv:2112.03334.

[4] Park, I., & Cho, J. (2023). SAiD: Speech-driven Blendshape Facial Animation with Diffusion. arXiv preprint arXiv:2401.08655.