MerkelNet: Training a Self-Supervised Lip-to-Speech Model in German

Gian Saß
Technische Hochschule Mittelhessen
Gießen
gian.sass@mni.thm.de

Abstract—This project report presents a lip-to-speech synthesizer trained on the Merkel Podcast Corpus [1]. The model consists of an encoder-decoder architecture that leverages a self-supervised training method, performing a sequence-to-sequence mapping of face frames to a mel spectrogram.

1 Introduction

The lip-to-speech task involves taking in a soundless video of a talking person and reconstructing the original speech. This may be useful, for instance, where a noisy sound recording renders a person's speech unintelligible, or when recovering speech from silent CCTV recordings. Notably, this involves many challenges. First, the lip movements and speech have to be in synchronization with each other, and lip movements may contain ambiguities, as several phonemes can be recognized from the same sequence of lip movements. It may not be entirely possible to reconstruct speech from only lip or face movements, since a significant portion of speech generation is done internally, for instance, in the vocal cords and pharynx.

Generating speech from lip movements can be modelled as sequence-to-sequence task. Given a sequence of face frames $F = \{F_1, F_2, ..., F_T\}$ we wish to output the speech $S = \{S_1, S_2, ..., S_T\}$. As output dimensionality it is affordable to use a low-level representation like mel spectrograms instead of raw waveforms. This greatly reduces the complexity of the target information that needs to be predicted. The resulting mel spectrogram can be converted back to a waveform representation using the Griffin-Lim reconstruction algorithm [2] or using a vocoder like WaveNet [3].

The model presented in this project report tries to leverage the natural co-occurrence of lip movement and audio without requiring any manual annotations. MerkelNet employs an encoder-decoder architecture which is trained end-to-end. The encoder consists of a stack of 3D convolution blocks to learn an implicit lip feature representation followed by a bidirectional LSTM to model long term dependencies in the temporal dimension. The decoder consists of an location-aware attention mechanism coupled with a single LSTM cell, and a projection layer. The mel spectrogram is decoded auto-regressively by conditioning it on the previous output (during training the ground-truth is used). A pre-net is used to bottleneck the information from the previous step, as this is crucial for attention alignment [4]. Finally, a post-net adds a residual to the resulting mel spectrogram to help smooth the transitions between individual frames.

2 Related Work

Inspired by the Tacotron 2 [5] text-to-speech synthesizer mapping text input to a low-level mel spectrogram representation, Qu et al. [6] adopted the architecture and proposed to directly use video inputs instead of text for lip-to-speech reconstruction. Similarly, Prajwal et al. [4] used 3-D CNNs and skip connections and improved the model performance. In [7] Qu et al. revisited their original approach and modified their encoder architecture to also use 3-D CNNs.

3 Model

MerkelNet uses a encoder-decoder architecture which is trained end-to-end. See figure 1 for an architecture overview.

3.1 Encoder

MerkelNet adopts a spatio-temporal face encoder similar to [7]. Input is of shape $B \times 3 \times T \times H \times W$ where 3 is the channel size, H and W are the spatial dimensions and T is the temporal dimension. A sliding window technique similar to [4] is employed where only small subclips of the same duration are trained on. A length of $T = 75$ is chosen which implies a three second video at 25 frames-per-second. Frames are fed into a stack of 3D CNN blocks which consist of a 3D convolution, batch normalization, ReLU activation, max pooling, and dropout. The encoder downsamples the spatial dimension of the input while increasing the channel size to get a single D-dimensional feature vector per input frame. Then, two bidirectional LSTM layers capture the long-term relationships from both the left and right direction. In this way, the encoder is forced to learn an implicit embedding that contains information about future and past lip movements and hence helps in the subsequent speech generation.

3.2 Decoder

3.2.1 *Attention*

The decoder uses an location-aware attention mechanism [8] to attend to the encoder output $h = (h1, ..., h_T)$ during decoding, extending the classical additive attention [9]. The mechanism is implemented according to [7].

The normalized attention weights a_t at decoding time t are obtained as follows:

$$a_t = \text{softmax}(W \cdot \tanh(M \cdot h + Q \cdot x + L \cdot y)) \quad (1)$$

$$x = \text{LSTM}(h \cdot a_{t-1}, p_{prenet}) \quad (2)$$

$$y = \text{Conv}(a_{t-1}, \sum_{0 \leq i \leq t-1} a_i) \quad (3)$$

where W, M, Q, and L are learned weight matrices of the attention layer. Through y the cumulative sum of previous attention weights is integrated, which allows the current step to be aware of the global location. Now, the attention content vector is given by

$$v_t = a_t \cdot h.$$

3.2.2 *Decoding Step*

The decoder LSTM consumes the attention content vector and the prenet output to generate the current frame. This output is then concatenated with the current attention content vector and passed through a projection layer to map it to the mel dimension. During training the ground-truth mel spectrogram is fed into the prenet, while during inference the previous mel vector is used. After decoding, a postnet consisting of five Conv1D layers generates a residual mel spectrogram which is added on top of the decoded mel spectrogram to help smoothen the transition between frames using future information that the decoder does not have access to.

3.3 Loss Function

Similar to [7] the loss is calculated as the MSE of the decoder output and target mel spectrogram which is further added to the MSE of the postnet output and the target mel spectrogram.

$$\mathcal{L} = \text{MSE}(O_{dec}, M_{target}) + \text{MSE}(O_{post}, M_{target}) \quad (4)$$

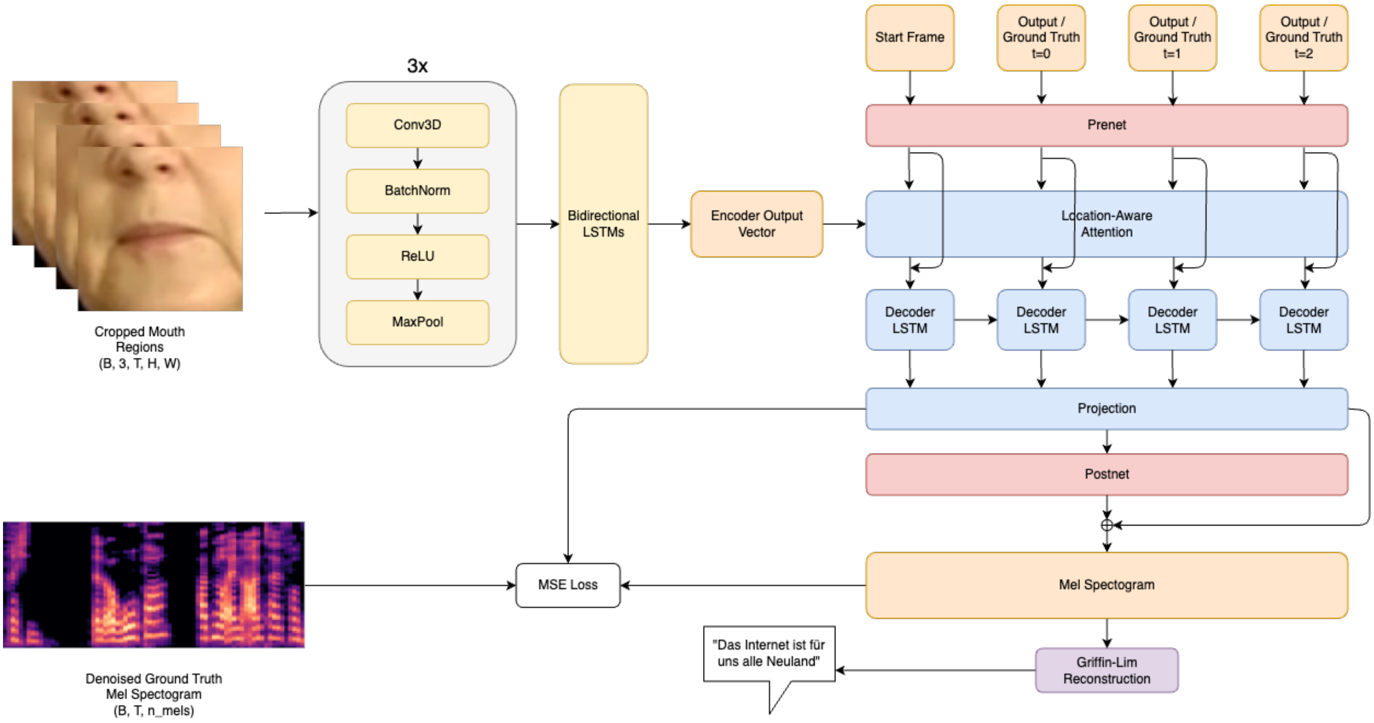
3x
Conv3D
BatchNorm
ReLU
MaxPool
Cropped Mouth Regions (B, 3, T, H, W)
Bidirectional LSTMs
Encoder Output Vector
Start Frame
Output / Ground Truth t=0
Output / Ground Truth t=1
Output / Ground Truth t=2
Prenet
Location-Aware Attention
Decoder LSTM
Decoder LSTM
Decoder LSTM
Decoder LSTM
Projection
Postnet
Mel Spectogram
MSE Loss
Griffin-Lim Reconstruction
"Das Internet ist für uns alle Neuland"
Denoised Ground Truth Mel Spectogram (B, T, n_mels)

Fig. 1. The architecture of MerkelNet. During training an input video is split into a visual stream of images, whose frames are fed into the encoder. The mel spectrogram representation of the video is designated to be the ground truth. The output mel spectrogram can be converted back to the time-domain waveform using the Griffin-Lim reconstruction algorithm.

3.4 Hyperparameters

See table A for an elaborate description of the hyper-parameters used during training.

4 Dataset Preprocessing & Training Parameters

4.1 Dataset

The model is trained on the Merkel Podcast Corpus [1], an audio-visual text corpus collected from 16 years of weekly internet podcasts. The full processed dataset contains about 17 hours worth of footage. The moviepy [10] library is used to extract frames and audio from the videos. Then, the Google MediaPipe [11] face landmark detector is used to detect lip movements in the source videos. After performing landmark smoothing the frames are centered around the center mouth position and cropped to a size of 96×96, and pixel values are normalized to a $[0, 1]$ range. Scaling is performed so that the mouth region occupies roughly the same size in the video regardless of distance to face. During training, on-the-fly image augmentation is used to increase the data available for training, which uses horizontal flipping, saturation modifications, and contrast changes.

The librosa [12] library is used to generate mel spectrograms from the source videos at a sampling rate of 16000 Hz, using 80 mel bands, maximum frequency of 8000 Hz, and an FFT window length of 2048. A hop length of $H = \frac{16{,}000}{25} = 640$ is chosen so that the face frames and mel spectrogram have the same temporal dimension. While this simplifies the model architecture, it also severely degrades the sound quality of the reconstructed speech. The noisereduce [13] library is used to remove noise from the original audio file. Then the mel spectrograms are converted to the decibel scale and normalized into a $[-4, +4]$ interval.

After preprocessing the dataset consists of short three second long clips extracted from the podcast corpus at random intervals.

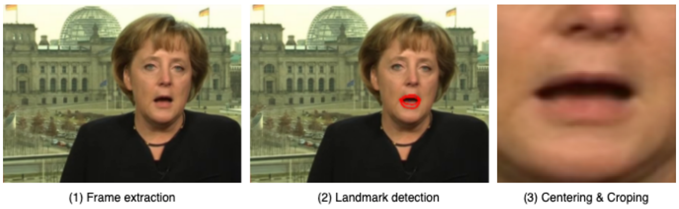

Fig. 2. Visualization of the preprocessing pipeline. Video frames (image no. 1) are first extracted using the moviepy library. Then MediaPipe is used to detect lip landmarks (image no. 2, marked in red) from the video frames. The mean position of lip landmark points is calculated and smoothed along the time dimension. Finally, the frames are cropped (image no. 3) into a standard resolution and centered around the mean position.

The dataset is divided into 90% training data and 10% test data. See figure 2 for a visualization of the data processing pipeline.

4.2 Training

For training, random clips are selected from the training dataset. The Adam [14] optimizer is used with a learning rate of 10^{-3} and a batch size of 32. Training was done on the THM Slurm Cluster on a NVIDIA V100 and on the RunPod service running on a NVIDIA A100. The platform Weights & Biases is used to track training metrics. The model is trained until the average test loss plateaus for at least 50k steps, for a total of about 150k steps or 300 epochs.

During the validation phase that occurs after every training epoch, the attention weights per decoder timestep are collected and plotted as a heatmap (figure 3). This is to keep insight into the model's alignment. Linear alignment between encoder and decoder starts to emerge around 15k steps and seems to firmly manifest at around 30k steps.

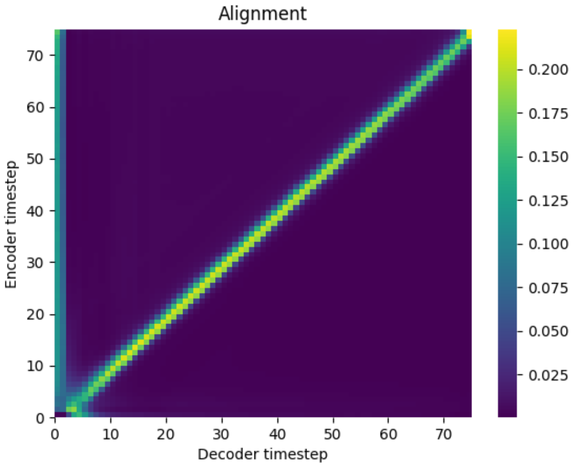

Fig. 3. Heatmap of attention alignment of the validation dataset at around 30k steps, demonstrating that generated speech is strongly conditioned on the sequence of lip movements.

Dataset	STOI	ESTOI
Merkel Podcast Corpus	0.540	0.318

TABLE 1
Evaluation metric results.

5 Results

Evaluation metrics (table 1) are determined using the latest model checkpoint on a random validation dataset. The average Short-Time Objective Intelligibility [15] (STOI) and Extended Short-Time Objective Intelligibility [16] (ESTOI) measures are calculated between the ground truth waveform and the inferenced waveform using the pystoi [17] library, after converting both mel spectrogram representations back to their waveform representations using the Griffin-Lim reconstruction. While a STOI measure of 0.54 is okay, the ESTOI measure of 0.318 is rather poor, hinging on the verge of unintelligibility.

On selected random data samples the model performs rather poorly, however the model clearly demonstrates solid alignment between lip movements and speech, e.g. there is no speech during moments of pause. Also, some frequent words like "Deutschland and "Bundestag" are clearly discernible.

5.1 Demo

A Gradio [18] demo interface is provided in the model repository [19]. This UI allows to easily inference a sample video and compare the mel spectrogram representations of the actual and predicted speech signals.

6 Discussion

6.1 Amount of data crucial

While the model shows promising results, there's clearly a lack of data, as the Merkel Podcast Corpus only compiles to about 17 hours worth of usable footage (as returned by the dataset preprocessing script). Meanwhile projects like Lip2Wav [4] are trained on about 10x the same amount of data. It is assumed that a greater volume of data would yield better results.

6.2 Prenet crucial for learning attention

Training experiments have shown that the density of the prenet layer is important for the model to learn attention. If the prenet does not sufficiently "compress" the previous mel vector the model starts to rely too much on the previous decoder output and not sufficiently enough on the encoder output. This can be clearly seen in the attention alignment heatmap, as the model only pays attention to the very beginning or very end of the encoder output. Notably the amount of dropout in the prenet layer is important. The model uses a prenet dropout of 0.5.

7 Conclusion

MerkelNet uses an encoder-decoder architecture to learn to map lip movements into speech signals. It does so through a self-supervised learning method. The challenges of the lip-to-speech task were mentioned, and a sample sequence-to-sequence model architecture was proposed.

References

[1] D. Saha, S. Nayak, and T. Baumann, "Merkel Podcast Corpus: A Multimodal Dataset Compiled from 16 Years of Angela Merkel's Weekly Video Podcasts,"
[2] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, pp. 236–243, Apr. 1984. Conference Name: IEEE Transactions on Acoustics, Speech, and Signal Processing.
[3] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," Sept. 2016. arXiv:1609.03499 [cs].
[4] K. R. Prajwal, R. Mukhopadhyay, V. Namboodiri, and C. V. Jawahar, "Learning Individual Speaking Styles for Accurate Lip to Speech Synthesis," May 2020. arXiv:2005.08209 [cs, eess].
[5] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," Feb. 2018. arXiv:1712.05884 [cs].
[6] L. Qu, C. Weber, and S. Wermter, "LipSound: Neural Mel-Spectrogram Reconstruction for Lip Reading," pp. 2768–2772, Sept. 2019.
[7] L. Qu, C. Weber, and S. Wermter, "LipSound2: Self-Supervised Pre-Training for Lip-to-Speech Reconstruction and Lip Reading," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2022.
[8] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-Based Models for Speech Recognition," June 2015. arXiv:1506.07503 [cs, stat].
[9] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," May 2016. arXiv:1409.0473 [cs, stat].
[10] "moviepy: Video editing with Python."
[11] "google/mediapipe," Mar. 2024. original-date: 2019-06-13T19:16:41Z.
[12] "librosa.feature.melspectrogram — librosa 0.10.1 documentation."
[13] "noisereduce: Noise reduction using Spectral Gating in Python."
[14] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017. arXiv:1412.6980 [cs].
[15] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4214–4217, Mar. 2010. ISSN: 2379-190X.
[16] J. Jensen and C. H. Taal, "An Algorithm for Predicting the Intelligibility of Speech Masked by Modulated Noise Maskers," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, pp. 2009–2022, Nov. 2016. Conference Name: IEEE/ACM Transactions on Audio, Speech, and Language Processing.
[17] P. Manuel, "mpariente/pystoi," Mar. 2024. original-date: 2018-04-18T12:01:22Z.
[18] G. Team, "Gradio."
[19] "dl4cvl2w / lip2wav · GitLab," Mar. 2024.

APPENDIX

Parameter	Purpose	Valu
temporal_dim	Temporal dimension of input	75
fps	Frames per second	25
sr	Sample rate of audio	1600
n_mels	Number of Mel bands to generate	80
n_fft	Length of the FFT window	1280
hop_length	Number of samples between successive frames	640
f_max	Highest frequency (Hz)	8000
w	Width of input frames	96
h	Height of input frames	96
batch_size	Training batch size	32
epochs	Number of training epochs	1000
learning_rate	Initial learning rate	10^{-3}
train_test_ratio	Ratio of training to testing data	0.9
codec	Audio codec used	"pcm
prenet_dim	Dimensionality of prenet layers	256
prenet_dropout	Dropout rate in prenet	0.5
postnet_dim	Dimensionality of postnet layers	512
postnet_kernel_size	Kernel size in postnet convolutions	5
postnet_n_convs	Number of convolutional layers in postnet	5
postnet_dropout	Dropout rate in postnet	0.5
attn_hidden_size	Hidden size of the attention LSTM	1024
attn_dim	Dimensionality of attention layers	128
attn_n_filters	Number of filters in attention convolutions	32
attn_kernel_size	Kernel size in attention convolutions	31
encoder_layers	Number of layers in the encoder	2
encoder_hidden_size	Hidden size of encoder LSTM	256
decoder_hidden_size	Hidden size of decoder LSTM	1024
min_level_db	Minimum decibel level for spectrograms	-10
ref_level_db	Reference decibel level for normalization	20
max_abs_value	Maximum absolute value for spectrograms	4
dropout	Dropout rate in various network parts	0.1

TABLE 2
Hyperparameters