Using Semantic Embeddings for Initiating and Planning Articulatory Speech Synthesis

Paul Schmidt-Barbo, Sebastian Otte, Martin V. Butz, R. Harald Baayen and Konstantin Sering

Eberhard Karls Universität Tübingen
paul.schmidt-barbo@student.uni-tuebingen.de; konstantin.sering@uni-tuebingen.de

Abstract: We present a method to both resynthesize and produce speech with the articulatory speech sythesizer VocalTractLab. We extend the recurrent gradient-based motor inference model for speech resynthesis with two generative adversarial networks (GANs). As a result, we are able to synthesize articulatory speech starting from the semantic level with distributed word embedding vector representations from fastText.

1 Introduction

Articulatory speech synthesis refers to the challenge to generate speech via a physical simulator of the vocal tract, including the pressure changes and resonances in the vocal and nasal cavities as well as the properties of the larynx. The parameters of the simulator control the position of individual articulators, such as jaw, lips, or tongue aiming to simulate how humans use their speech organs to produce speech. The main challenge in the process of articulatory speech synthesis is to find a physically plausible set of control parameter trajectories (cp-trajectories) to produce intelligible speech. This approach stands in contrast to approaches that generate speech by averaging similarly sounding speech segments [1] or by learning autoregressive, generative models from natural language data [2].

To generate a target acoustic via a cp-trajectory fully or semi-automatically, typically, prior assumptions about phonemic and syllable structures including feedback control are made. The neuro-inspired DIVA model [3], for example, learns to generate cp-trajectories via auditory and somatosensory feedback loops. It is able to resynthesize or copy-synthesize target wave files. DIVA and related models, however, do not take semantic information, such as word meanings, into account. From a psychological perspective, though, meaning is considered the beginning and primary purpose of language production and processing [4]. Accordingly, we explore in which way word meanings may improve speech synthesis [5].

One of the most promising approaches for capturing semantic information of words (or other linguistic constructs) is the usage of distributional semantics models. These models derive semantic information from distributional properties found in large text corpi, such that word meanings are expressed in the form of points in an embedding space—so-called word embeddings or semantic vectors. A suitably-structured embedding space allows a distance-based computation of word meaning dissimilarities and associates linear shifts with specific semantic changes in meaning. Because meaning is thus encoded by a fixed-size vector, it is not necessary to deal with an infinite set of different symbolic meanings. Furthermore, in contrast to symbolic approaches, word embeddings derived from real language data can be directly connected to experience and human learning.

At ESSV 2021 [5] we introduced a first model that is able to infer cp-trajectories from semantic embeddings. We extended the recurrent gradient-based motor inference model for speech resynthesis [6] with an embedding model that is trained to learn a mapping from an acoustic representation to a semantic word embedding. The resulting model enabled semantic discrimination, instead of mimicking acoustics, by adding an additional loss term capturing the discrepancy between the predicted and target semantic embedding.

In this contribution, we aim at starting the language production process fully on the semantic level by extending our previous model [6, 5] to the Predictive Articulatory speech synthesis Utilizing Lexical Embeddings (PAULE) model[1]. PAULE follows a behavior-based approach relying on prediction errors of a forward model and an internal planning process, which uses gradient information to infer cp-trajectories. Based on temporal gradient-based inference [7, 8], we introduce a sequence-to-sequence and a fixed-vector-to-sequence approach to infer an appropriate cp-trajectory. Besides our usage of long short-term memory recurrent artificial neural networks (LSTMs) [9], we use two generative adversarial networks (GAN) for building an ensemble of models that is capable of a) resynthesizing given audio and b) generating speech from semantic embedding vectors. Given a word embedding, the GANs thus allow us to sample new motoric and acoustic word representations in the form of cp-trajectories (Cp-GAN) and log-mel spectrograms (Mel-GAN), thus initiating the speech generation process from the semantic level.

All models used within our framework are completely agnostic to any motor gestures or phonemic transcriptions. Moreover, they are fully differentiable. Feedback from errors between target and produced speech on the acoustic and semantic level is, in contrast to the DIVA model, not directly learned but provided by backpropagated gradient information. The whole framework is hard-constrained only by the geometrical and physical constraints of the articulatory synthesis model, by the chosen semantic embedding and the log-mel spectrogram encoding, and by the temporal resolution used in the control parameters and in the acoustic domains. Soft constraints on the flexibility of articulators, namely jerk and velocity constraints, are applied during planning.

2 Methods

As depicted in **Figure 6** the framework involves five different models or, more specifically, two generative models (shaded in green) and three predictive models (shaded in red), as well as the VocalTractLab (henceforth, VTL) speech synthesizer (shaded in yellow). Blue shaded boxes indicate data objects, namely, 1) the cp-trajectories, which govern the vocal tract simulator, 2) the log-mel spectrograms, which encode the acoustics, and 3) the semantic vector space embeddings, which operationalize the meaning of the utterances. The three predictive models are LSTM-based and include 1) an inverse model, which provides initial cp-trajectories predicted from a log-mel spectrogram, 2) a forward model, which predicts a VTL synthesized log-mel spectrogram from a set of cp-trajectories, and 3) an embedding model, which maps a variable-sized acoustic representation in form of a log-mel spectrogram onto a fixed length semantic vector. Because the embedding and forward model are differentiable, the loss in semantic and/or acoustic space can be backpropagated towards the encoding of the cp-trajectories, striving to infer the best error-minimizing trajectory. The inverse model, on the other hand, is used to initialize the cp-trajectories.

Besides the predictive models, our framework incorporates two generative models shaded in green. More precisely, we used two conditional Wasserstein GANs trained with Gradient Penalty. The first model, henceforth Cp-GAN, was trained to find a mapping conditioned on a semantic vector indicating a certain word meaning from a random noise vector z to a set of cp-trajectories. The second one, henceforth Mel-GAN, was trained similarly to find a mapping to a log-mel spectrogram, again with the log-mel spectrograms from VTL synthesized audios as physical ground truth. Details of the models are provided below.

[1] `https://github.com/quantling/paule`

2.1 Scenarios and model evaluations

To explore the inference of motor commands for speech resynthesis, we distinguish three starting scenarios with three different planning objectives. In the first starting scenario, the *mimicking* condition, we provide a real-world recording as the target acoustic. Initial cp-trajectories are predicted by the inverse model. In the second scenario, called *supervised generation*, we also provide the real-world target acoustic but generate an initial set of cp-trajectories using the Cp-GAN. In the third scenario, called *full generation*, we provide a semantic target and use both GANs: the Cp-GAN to initialize the cp-trajectories; the Mel-GAN to generate a target acoustic. After initialisation, the inference process unfolds similarly in all conditions. Starting from the initial cp-trajectories as input, we use the forward model to predict an imagined acoustic representation. The predicted acoustic representation is mapped onto the semantic space by the embedder, imagining the matching semantic embedding. Depending on the chosen objective (see Section 2.5), we calculate an error between target and predictions. Since all predictive models are fully differentiable we can derive the gradients given this discrepancy and use them to iteratively improve the initial cp-trajectories. This process is called planning. We further add some soft constraints to the motor effort of articulators, namely jerk and velocity loss. We repeat the planning process until we reach a final set of optimized cp-trajectories, which are then executed by the VTL producing the final resulting acoustics. Finally, these acoustics are mapped onto the semantic space by the embedder, predicting the final semantic representation.

By using the gradients, we are able to learn corrections to some cp-trajectories, creating a new sample of motor-acoustics relations for the VTL. This new sample can be incorporated in further training of the forward model in order to keep it in sync with the VTL synthesizer. Depending on the desired performance, planning and additional learning steps can be interleaved to improve the planned cp-trajectories. In our experiments we plan each cp-trajectory for 120 iterations, while continuing learning every 24 iterations. This gives a good compromise between model accuracy and computation time.

To test and quantify the performance of the whole framework, we compare the final synthesized acoustics with real-world recordings. The comparisons include calculating the root mean square error (RMSE) between produced and target log-mel spectrograms as a measure of acoustic similarity and the RMSE between embedded semantic vectors as a measure of intelligibility. As a control condition, we include a comparison with a segment-based approach (see Section 2.2) [10].

2.2 VocalTractLab (VTL)

For the articulatory speech synthesis we used the *VocalTractLab* synthesiser (VTL) version 2.3 developed by Peter Birkholz [11] as described in [5]. VTL takes a sequence of 30 control parameters (cps) per time step as input and emits a 44,100 Hz mono audio signal. Every parameter is defined every 110 audio samples (2.5 ms) with transitions interpolated between. Furthermore, we use the VTL API method to derive cp-trajectories from phone sequences and predefined gestural scores (segment-based approach; [10]).

2.3 Forward, Inverse, & Embedding Model

The predictive *forward model* shortcuts the VTL simulator. It learns to predict a log-mel spectrogram from a set of cp-trajectories approximating the acoustic representation resulting from the forward synthesis by the VTL. It plays the main role during the planning process since its gradients are used as a local approximation of the gradients of the VTL. Besides being differentiable the forward models possesses an execution time advantage compared to the vocal tract lab. A single execution of the forward model plus the backpropagation of the error is around 100 times faster than an actual simulation with the VTL simulator. The model is implemented as a recurrent sequence to sequence LSTM model. The input cp-trajectories serve as input for a 720 cell one-layer LSTM-network. The output of the LSTM-layer is linearly mapped to 60 dimensions. Since we calculated log-mel spectrograms with a sampling resolution of 5 ms in contrast to cps being sampled every 2.5 ms the sequence length is halved with an average pooling layer.

The *inverse model* is only used in the mimicking scenario. It is trained to learn a mapping from a log-mel spectrogram to a cp-trajectory. Since this mapping is a one-to-many problem (many cp-trajectories resulting in very similar or identical acoustic manifestations), the inverse model is only used to create a first guess of suitable cp-trajectories, thus initializing the planning process ideally from an eligible trajectory. The model is implemented as a recurrent sequence to sequence LSTM model with 720 hidden units and an additional upstream convolutional block as described in [5] . The output of the last layer is linearly mapped to 30 dimensions, representing the 30 control parameters of the VTL. We double the sequence length by linearly interpolating between time steps before smoothing trajectories over time by a set of one dimensional convolutions.

Both models are trained independently using the ADAM optimizer [12] with default parameters and an initial learning rate of 0.001 which was dropped to 0.0001 after 50 epochs.They strive to minimize the RMSE loss with a batch size of 8 for 100 epochs. To keep the influence of padding small, batches are grouped to samples with similar length (same size batching).

The predictive embedding model (*embedder*) maps an acoustic representation in the form of a log-mel spectrogram to a 300 dimensional word embedding vector. We calculate dissimilarities between meanings in terms of Euclidean distances between two word embedding vectors. Although it is debatable whether the embedding space is a valid vector space and Euclidean distance globally meaningful, the Euclidean distance is the most direct way to determine a loss signal. We use the minimal Euclidean distance to a target fastText semantic vector to predict labels for an embedded acoustic. Therefore, the distance in semantic space serves as a kind of measurement for intelligibility. We built the sequence-to-fixed-vector embedder using a LSTM-network containing 2 layers with a hidden size of 720. The output is linearly mapped to 300 output units corresponding to the prediction of a word embedding.

We trained the embedder using the RMSE loss and an ADAM optimizer [12] with default parameters and an initial learning rate of 0.0001 for 200 epochs. We used a batchsize of 8 applying same size batching. During training we used dropout of 0.7 in the last LSTM-layer forcing the model to predict a target vector with only a subset of hidden units. Additionally, we trained the model on target vectors with additive noise. For each sample in each batch, we drew a noise vector from an independent multivariate Gaussian with $\mu = 0$ and $\sigma = 6^{-5}$. The standard deviation σ was derived as one third of the averaged minimal Euclidean distance in each dimension between all target vectors.

2.4 Cp- and Mel-GAN

The Cp-GAN and Mel-GAN belong to the category of implicit generative models, the so-called Generative Adverserial Networks (GAN) [13].Each GAN consists of two individual models trained simultaneously with a technique from game theory, the so called minimax two-player game. One model is called the generator. It is trained to find a mapping from a predefined latent space to a data distribution $P_z(z)$ with the same statistics as the training data P_{data}. The second model is called the discriminator or critic. It is trained to distinguish true and generated samples, which corresponds to minimizing a similarity or maximizing a distance between true and generated data distributions. The generator tries exactly the opposite by using gradient information of the critic. A common choice is to train the critic with the JS-divergence. This however, can cause the common problem of vanishing gradients. In cases of little to no overlap between real and generated distribution, the JS-divergence leads to a constant value of $log(2)$ giving no gradient information. That is why Gulrajani et al. [14] introduced the so called Wasserstein GANs with a critic approximating the Wasserstein distance and leading to a continuous differentiable value function. Besides the vanishing gradient problem, GANs may suffer from so-called mode collapse, in which the model generates samples for only one specific class. Mirza and Osindero [15] could show that providing additional information to both generator and critic can direct the data generation process and control the modes of the data being generated.

For the Cp-GAN and Mel-GAN we used both improvements. We trained a critic to approximate the Wasserstein distance between true and generated distribution taking variable length-sequences and a 300 dimensional semantic embedding vector as additional information. For the generator we sampled a 100 dimensional latent noise vector from a normal distribution, which was concatenated with a provided semantic embedding vector leading to 400 input dimensions. We implemented both models as deep convolutional networks with an adapted architecture inspired by Kendrick et al. [16]. The generator model takes the 400 dimensional input vector together with a desired sequence length. The input vector is mapped to 1024 dimensions by a fully connected layer. The output is reshaped to a two dimensional tensor with an initial sequence length of four and 256 initially extracted features before fed into five blocks of 1-dimensional convolutional layers. We ensured that no size editing takes place within these blocks. With respect to Kendrick et al. [16] and in order to create a sequence equal to the provided input length resizing is done progressively throughout the network by linearly upsampling the output between blocks. Arriving at a sequence of exact length we apply a linear layer to map the 256 hidden features to either 30 output dimensions representing the 30 cp-trajectories (Cp-GAN) or 60 output dimensions representing the mel-channel of a log-mel spectrogram (Mel-GAN).

The critic receives a sample either from the generator or from the training data. We also provide the corresponding 300 dimensional semantic vector. The semantic vector is concatenated with the input dimensions leading to an input of size $length \times 330$ (Cp-GAN) or $length \times 360$ (Mel-GAN). This input is fed into an initial linear layer with 180 hidden units and is passed on to five blocks of 1-dimensional convolutional layers. In the last layer we take the average over all channels, generating an estimate of the Wasserstein distance (Global Average Pooling [17]).

Although W-GAN are stated to have a better training stability [14] their performance is dependent on their training procedure, especially on their estimate of the Wasserstein distance. We therefore started the training by giving the critic 5 more update iterations before updating the generator. Updates were performed using a ADAM optimizer with a learning rate of 0.0001 and parameters $\beta_1 = 0.5$, $\beta_2 = 0.9$, $\varepsilon = 10^{-8}$. We used a batch size of 64 applying same size batching. We calculated the Wasserstein distance by taking the average of differences between the output of the critic for a batch of true samples and a batch of corresponding generated ones. For the critic loss we minimized the negative (reformulation of maximizing the the positive) of the result together with an added gradient penalty loss weighted by $\lambda = 10$ [14]. For the generator loss we minimized the mean of differences between the outputs of the critic for real and generated trajectories. Over training we monitored the loss and steadily increased the critic iterations until we reached a computational maximum of 100 critic iterations. We trained the Cp-GAN for 415 epochs and the Mel-GAN for 400 epochs in total.

Calculations for all models were performed using an Intel Xeon CPU and a Tesla P100-PCIE-16GB GPU. Training the inverse and forward model finished after around 12h while training the embedder took around 22h. For both GANs we stopped the training after around 550h. The whole training process consumed approximately 400 kWh.

2.5 The planning loss

The planning loss is implemented as an additive loss depending on the objective. The RMSE loss between target and predicted log-mel spectrogram focuses on acoustic patterns only (acoustic objective). The RMSE loss between target and predicted semantic vector focuses on semantics (semvec objective). Combining both losses (acoustic semvec objective) enforces a semantically-stable acoustic imitation. All objectives add a velocity and jerk loss component per motor trajectory dimension, implementing localization and smoothness constraints that favor physical plausible trajectories that do not move or move with a constant force.

2.6 Initial training data

The initial training data set consists of a subset of the German Common Voice corpus [18], which is a crowd-sourcing project consisting of mostly read out sentences recorded and reviewed by volunteers in many different languages. After aligning sentences using the Montreal forced aligner [19] on the segment, syllable and word level, a subset of 26271 word tokens (training: 21175, validation: 5096) is randomly selected. For this subset, resynthesized versions are generated with the segment-based approach from VTL. The predictive and inverse model are trained on the derived cp-trajectories (sequence length between 246 and 924 time steps) together with resulting log-mel spectrograms (sequence length between 123 and 462 time steps) computed on the resynthesized speech. The log-mel spectrogram for each word is computed using 60 mel banks within a frequency range from 10 Hz to 12,000 Hz, a window length of 1024 samples (23.2 ms), and a time delta of 220 samples (5 ms). The mel banks are calculated on a magnitude spectrum using `librosa` (version 0.8.0). Wave forms are sampled at 44,100 Hz. Finally, the log-mel spectrogram is linearly transformed into the 0 to ∞ range where 0 corresponds to silence and 1 is a loud and clear tone. As silence is not mapped on a fixed log-mel spectrogram value, the minimum value is subtracted individually in each log-mel spectrogram. Data per time slice was reduced from 220 samples in the wave form to 60 samples in the log-mel spectrum. The subsample contains 4311 unique word types. For the embedder log-mel spectrograms calculated on the real-world recorded audio (sequence length between 7 and 487 time steps) are added to the data set. For the target semantic, vectors consist of 300 dimensional pre-trained word embedding vectors from fastText [20]. These come from a pre-trained continuous bag of words model (CBOW) trained on the German version of Common Crawl and Wikipedia. The Cp-GAN and Mel-GAN are trained on a subset of resynthesized GECO corpus words [21] (29521 word tokens). Besides the training and validation set, a test set consisted of 225 unseen utterances of 13 words from the Common Voice corpus. Words were chosen with the aim to create a diverse set of examples capturing different speech aspects. The word frequency effect is minimized by between 10 and 20 samples per word.[2]

3 Experiments and Results

The results of our experiments indicate the potential of PAULE. We first analyze the predictive abilities of the forward model and the GANs and then report the inference-based planning performance of the model.

[2]Words included 'Beispiel', 'Freunde', 'Lehrer', 'Studium', 'aber', 'eigentlich', 'naemlich', 'natuerlich', 'praktisch', 'schwierig', 'tatsaechlich','trotzdem', and 'zurueck'.

3.1 Performance of Predictive and Generative Models

We trained the forward model and the inverse model for 100 Epochs on the Common Voice training set. After training, the forward model achieved an average RMSE loss of 1.88×10^{-2} (first epoch: 4.73×10^{-2}) between predicted and produced log-mel spectrogram. The inverse model yielded an average RMSE loss of 4.77×10^{-3} (first epoch: 5.93×10^{-2}) between predicted and target cp-trajectories, given the segment-based approach as ground truth in the validation set.

Training the Embedder for 200 Epochs resulted in an average RMSE loss of 1.13×10^{-2} (first epoch: 3.31×10^{-2}) between predicted and target semantic embeddings for the log-mel spectrograms of the validation set. Using the minimal Euclidean distance, we assigned labels to the predicted embeddings before calculating the top 1 accuracy. In order to account for the imbalance of token within types we first calculated the mean accuracy within a word type before averaging over types. We achieved a type accuracy of 57.6%. Splitting up the results into recordings and segment-based syntheses revealed a large gap: While the embedder achieves an accuracy of 33.6% for recordings, it achieves an accuracy of 81.6% for the segment-based syntheses. This behavior might be due to the large variation in sound quality of the recordings, potentially misaligned recordings, and artificial regularities in the segment-based log-mel spectrograms. For the samples of our small selected test set we achieved a overall accuracy of 80.9% with 62.7% for recordings and 99.1% for segment-based syntheses.

Due to time and energy considerations, we stopped GAN training before reaching an equilibrium. Therefore, we may have not tapped into the maximum potential of both models. Nonetheless, the inspected results (examples in **Figure 3,4**) already indicate that the generated samples capture most of the relevant properties of their underlying distribution.

3.2 Inference-based planning

Having evaluated all models of our PAULE framework individually, we moved on to the planning procedure of the 225 word tokens in the test subset. We first fixed our target acoustic and target semantic. Depending on the condition, this consisted of a log-mel spectrogram either derived from a real-world recording or generated by our Mel-GAN. For the target semantics we used the ground truth fastText vector. We started planning from an initial cp-trajectory, which was either predicted by the inverse model or generated by the Cp-GAN. Given this starting point, we utilized the forward model to predict a log-mel spectrogram and the embedder to further map the spectrogram onto a semantic vector (black arrows in **Figure 6**). We calculated the RMSE for predicted and target log-mel spectrograms as well as for the predicted and target semantic vectors (red dotted lines in **Figure 6**). Depending on the objective one or both of the errors were backpropagated along the gradients of the models. Reaching the initial cp-trajectories, we combined the backpropagated with the velocity and jerk loss.

Figure 1 shows the improvements of the acoustic RMSE loss for the final produced acoustics in all conditions. The inference procedure is reducing the loss in all conditions, confirming that the planning generally works. Both the inverse model and the Cp-GAN are able to predict/generate reasonable cp-trajectories, which result in acoustically similar or even better syntheses compared to the segment-based approach. As expected, planning solely on acoustics achieves best acoustic results followed by the combined loss of acoustics and semantic loss (acoustic semvec). Optimizing solely on the semantic loss does not yet improve the acoustics.

Figure 2 shows the best resulting semantic RMSE losses for recorded and segment-based embeddings. This is due to the fact that the embedder is trained on log-mel spectrograms that were derived from both recordings and segment-based syntheses. The semvec RMSE loss for embeddings from generated and initially produced log-mel spectrograms shows that both GANs

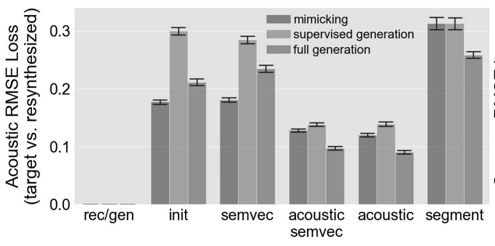

Figure 1 – Mean RMSE loss between final produced and target acoustic for the semvec, acoustic semvec, and acoustic objective. Init refers to the initial cp-trajectory.

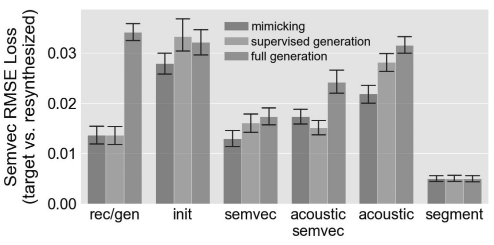

Figure 2 – Mean RMSE loss between embedded final acoustic and target semantic vector for the semvec, acoustic semvec, and acoustic objective. Init refers to the initial cp-trajectory.

seem to miss out on details in the log-mel spectrograms, respectively in the cp-trajectories producing log-mel spectrograms, that are important for the embedder. Nevertheless, planning shows clear improvements in the semantic RMSE loss while optimizing solely on the semantic loss and on the acoustic and semantic loss.[3]

Finally, **Figure 5** shows the top 1 accuracies of the resulting semantic vector embeddings. While planning on one objective alone only slightly improves accuracy, trying to match the acoustic target while simultaneously aiming for the right semantic embedding vector performs comparable to real world recordings.

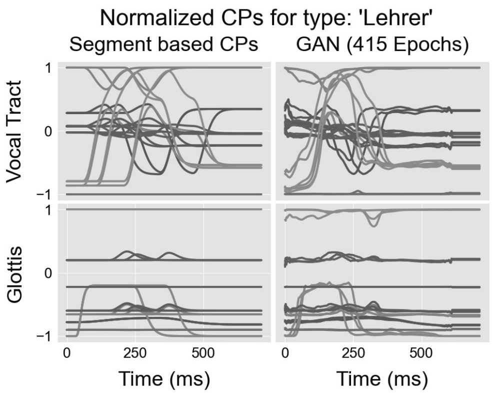

Figure 3 – Exemplar Cp-trajectories generated by the Cp-GAN for the word “Lehrer”. Colors encode the first eight control parameters.

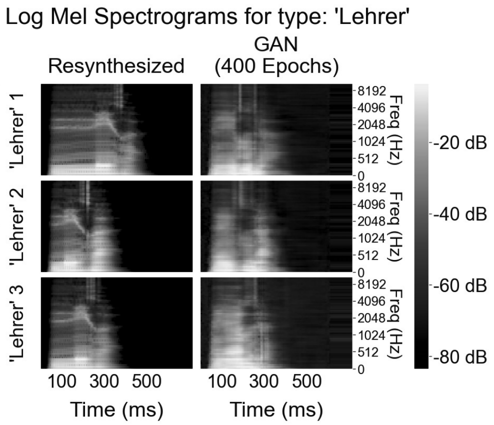

Figure 4 – Log-mel spectrograms generated by the Mel-GAN for the word “Lehre” in comparison to segment-based syntheses.

4 Discussion, Conclusion, & Future work

In conclusion we can say that the PAULE model is able to find and improve cp-trajectories in a reasonable time (300 ms of speech planned in 3-4 minutes) using gradient based planning starting from a target acoustic solely using a semantic vector. Combining an acoustic RMSE loss between a predicted and target log-mel spectrogram, with a semantic loss in form of the RMSE between embeddings in semantic space, leads to cp-trajectories that produce an acoustic, which is as ”*intelligible*” for our embedder as a real world recording.

[3]The iterative nature of improving the semantic distance to a target semantic embedding is visualized by the paths in the U-Map embedding [22] shown in the framework overview (**Figure 6**).

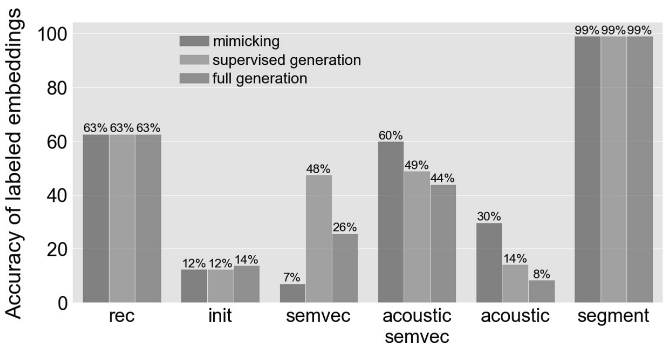

Figure 5 – Word classification accuracy of produced acoustics by embedding the produced acoustic into a 300 dimensional embedding space and selecting the closest fastText vector of a word type for the different objectives.

We are aware that this cannot be equated with human performance. That is why the actual objective should be the perceived opinion of a human listener comparing synthesized to recorded words. However, with respect to Arnold et al. [23] and the overall low recognition accuracy of human listeners for single spliced-out words, we think a reasonable first step should be to establish an iterative process for planning whole sentences before drawing this comparison. Besides improving the computations by applying bigger vocabulary and even switching the language to English, future plans include informing the model with sensomotoric feedback, similar to the DIVA model [3]. An obvious approach would be to extract contact points from the VTL, train another forward model to predict these contact points from cp-trajectories, and use the forward model inversely to improve cp-trajectory inference even further.

Although certainly further improvements are possible, we conclude that the good performance of PAULE implies that the generation of semantically well-comprehensible words can be significantly improved by combining direct acoustic and cp-trajectory initializations with inference-based planing that optimizes both acoustic quality and semantic discriminability.[4]

References

[1] Yoshimura, T., K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura: *Mixed excitation for hmm-based speech synthesis*. In *Seventh European Conference on Speech Communication and Technology*. 2001.

[2] van den Oord, A., S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu: *Wavenet: A generative model for raw audio*. 2016. 1609.03499.

[3] Tourville, J. A. and F. H. Guenther: *The diva model: A neural theory of speech acquisition and production. Language and cognitive processes*, 26(7), pp. 952–981, 2011.

[4] Harley, T. A.: *The psychology of language: From data to theory*. Psychology press, 2013.

[5] Schmidt-Barbo, P., E. Shafaei-Bajestan, and K. Sering: *Predictive articulatory speech synthesis with semantic discrimination*. In S. Hillmann, B. Weiss, T. Michael, and S. Möller (eds.), *Studientexte zur Sprachkommunikation: Elektronische Sprachsignalverarbeitung 2021*, pp. 177–184. TUDpress, Dresden, 2021.

[6] Sering, K., P. Schmidt-Barbo, S. Otte, M. V. Butz, and H. Baayen: *Recurrent gradient-*

[4]**Acknowledgments** This research was supported by an ERC Advanced Grant (no. 742545) awarded to R. Harald Baayen. All authors are members of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645.

based motor inference for speech resynthesis with a vocal tract simulator. In *12th International Seminar on Speech Production*. 2020.

[7] OTTE, S., T. SCHMITT, K. FRISTON, and M. V. BUTZ: *Inferring adaptive goal-directed behavior within recurrent neural networks*. In *International Conference on Artificial Neural Networks*, pp. 227–235. Springer, 2017.

[8] BUTZ, M. V., D. BILKEY, D. HUMAIDAN, A. KNOTT, and S. OTTE: *Learning, planning, and control in a monolithic neural event inference architecture*. *Neural Networks*, 117, pp. 135–144, 2019.

[9] HOCHREITER, S. and J. SCHMIDHUBER: *Long short-term memory*. *Neural computation*, 9(8), pp. 1735–1780, 1997.

[10] GAO, Y., P. STEINER, and P. BIRKHOLZ: *Articulatory copy synthesis using long-short term memory networks*. In R. BÖCK, I. SIEGERT, and A. WENDEMUTH (eds.), *Studientexte zur Sprachkommunikation: Elektronische Sprachsignalverarbeitung 2020*, pp. 52–59. TUDpress, Dresden, 2020.

[11] BIRKHOLZ, P.: *Modeling consonant-vowel coarticulation for articulatory speech synthesis*. *PloS one*, 8(4), p. e60603, 2013.

[12] KINGMA, D. P. and J. L. BA: *Adam: A method for stochastic optimization*. *3rd International Conference for Learning Representations*, abs/1412.6980, 2015.

[13] GOODFELLOW, I., J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, and Y. BENGIO: *Generative adversarial nets*. *Advances in neural information processing systems*, 27, 2014.

[14] GULRAJANI, I., F. AHMED, M. ARJOVSKY, V. DUMOULIN, and A. C. COURVILLE: *Improved training of wasserstein gans*. *CoRR*, abs/1704.00028, 2017. `1704.00028`.

[15] MIRZA, M. and S. OSINDERO: *Conditional generative adversarial nets*. 2014. `1411.1784`.

[16] KENDRICK, C., D. GILLESPIE, and M. H. YAP: *Anysize gan: A solution to the image-warping problem*. 2020. `2003.03233`.

[17] CUI, Y., F. ZHOU, J. WANG, X. LIU, Y. LIN, and S. J. BELONGIE: *Kernel pooling for convolutional neural networks*. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3049–3058, 2017.

[18] ARDILA, R., M. BRANSON, K. DAVIS, M. HENRETTY, M. KOHLER, J. MEYER, R. MORAIS, L. SAUNDERS, F. M. TYERS, and G. WEBER: *Common voice: A massively-multilingual speech corpus*. *CoRR*, abs/1912.06670, 2019. `1912.06670`.

[19] MCAULIFFE, M., M. SOCOLOF, S. MIHUC, M. WAGNER, and M. SONDEREGGER: *Montreal forced aligner: Trainable text-speech alignment using kaldi*. In *INTERSPEECH*. 2017.

[20] GRAVE, E., P. BOJANOWSKI, P. GUPTA, A. JOULIN, and T. MIKOLOV: *Learning word vectors for 157 languages*. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*. 2018.

[21] SCHWEITZER, A. and N. LEWANDOWSKI: *Convergence of articulation rate in spontaneous speech*. In *INTERSPEECH*, pp. 525–529. 2013.

[22] MCINNES, L., J. HEALY, and J. MELVILLE: *Umap: Uniform manifold approximation and projection for dimension reduction*. 2020. `1802.03426`.

[23] ARNOLD, D., F. TOMASCHEK, K. SERING, F. LOPEZ, and R. H. BAAYEN: *Words from spontaneous conversational speech can be recognized with human-like accuracy by an error-driven learning algorithm that discriminates between meanings straight from smart acoustic features, bypassing the phoneme as recognition unit*. *PLOS ONE*, 12(4), pp. 1–16, 2017. doi:10.1371/journal.pone.0174623.

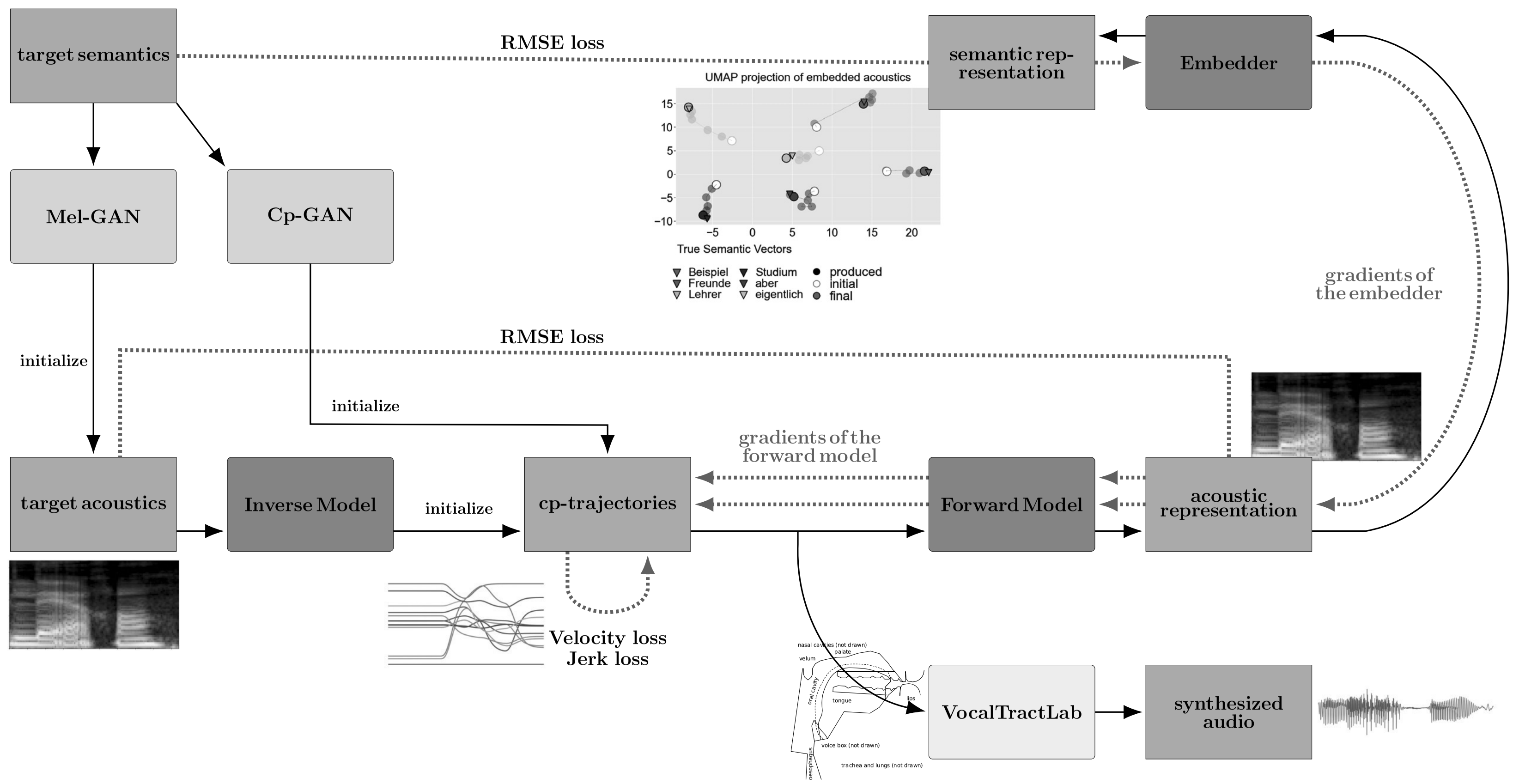

Figure 6 – The PAULE model. Data structures (shaded in blue) are connected by different models. The models used can be divided into generative (shaded in green) and predictive models (shaded in red). All forward passes are depicted by black arrows while backward passes containing gradient information are shown as dotted red arrows. Losses are calculated between target and predictions connected by dotted red lines. The VTL synthesizer receiving the final planned cp-trajectories is displayed in yellow.