# PREDICTIVE ARTICULATORY SPEECH SYNTHESIS WITH SEMANTIC DISCRIMINATION

*Paul Schmidt-Barbo, Elnaz Shafaei-Bajestan, Konstantin Sering*

*Eberhard Karls Universität Tübingen*
*paul.schmidt-barbo@student.uni-tuebingen.de*

**Abstract:** An articulatory speech sythesiser is presented that extends the Recurrent Gradient-based Motor Inference Model for Speech Resynthesis [1] with a classifier optimizing for semantic discrimination. The new design features show promising results in improving model performance.

## 1 Introduction

Articulatory speech synthesis uses a physical simulation of the pressure changes and resonances in the vocal and nasal cavities as well as the properties of the larynx. In contrast to statistical or deep neural networks based speech synthesis, this approach gives insights into how humans produce speech with their speech organs. One challenge in doing articulatory speech synthesis is, given a target acoustics, finding the right set of control parameter (cp-)trajectories that drive the speech synthesiser to produce intelligible speech with a similar meaning and a similar acoustic pattern.

There are semi-automatic and fully automatic approaches to infer the cp-trajectories for a given target acoustic [2, 3]. All of these models allow for a resynthesis or a copy-synthesis of a target wave file but differ in acoustic quality, cp-trajectory quality, and movement plausibility. While these approaches do focus on the acoustics and motor planning, they do not take into account any general intelligibility or discriminability measure. Using a measure that discriminates between different possible meanings could not only further optimize the produced audio and the planned cp-trajectories, but is also highly motivated from a psychological perspective: Harley argues that meaning is that start of the language production process and the primary purpose of language processing [4].

In the last twenty years models widely known as the distributional semantics models have seen a rising wave of interest as they appear to elegantly model the degree of semantic similarity between words (or other linguistic constructs). In this framework, word meanings are no longer atomic units or a composite of logical symbolic elements. Rather, they are points in an embedded space; hence the name *word embeddings*. Linear shifts in the space can be associated with specific common changes in meaning and dissimilarities between meanings can be computed in terms of distances.

In this study, we show how the linguistic information captured by the semantic embeddings can be employed to inform a fully automated resynthesiser for articulatory synthesis. The presented framework takes a sequence-to-sequence approach and maps the positions of the articulators defined at every 2.5 ms to the log-mel spectrogram of the imagined sound defined at every 23.2 ms with a 5 ms time resolution. The model has access to distances between the actual produced meaning, the desired meaning, and all other meaning competitors. What's more, the model is completely agnostic to any motor gestures or phoneme transcriptions. The framework is only hard-constrained by the geometrical and physical constraints of the articulatory synthesis model, and by the time resolution used in the cp-trajectory and the acoustic domains. Soft

constraints on the flexibility of cp-trajectories, namely jerk and velocity constraints, are applied during planning. The training data implicitly brings in some gesture and phone segments due to the way human speech and its models seem to be structured.

# 2 Methods

## 2.1 Framework description

The framework is depicted in Figure 4 and consists of three main data representations, shaded in pale blue, and four models, shaded in pale red, that connect these data structures. The three data structures are: 1) the cp-trajectories which govern the vocal tract simulator; 2) the log-mel spectrograms that represent the acoustics; and 3) the semantic vector space embeddings that operationalise the meaning of the utterances. The four models include: 1) the classifier that maps an acoustic representation to a semantic representation; 2) the VocalTractLab (henceforth, VTL) simulator which consumes the cp-trajectories and synthesises a monophonic audio – this forward synthesised audio determines the physical ground truth of the acoustic representation; 3) the forward predictive model that shortcuts the VTL path and only emulates the VTL synthesis using experience accumulated over a learning history; and 4) the inverse model which provides the initial planning of the trajectories.

To resynthesise a given target audio, first, the cp-trajectories are initialized by the inverse model. At the same time, a target semantic embedding is constructed for the target word. Next, the predictive model maps the initial trajectories to an imagined predicted acoustic representation. This acoustical representation is subsequently mapped to the semantic space by the classifier yielding an imagined predicted semantics.

After minimizing a joint loss (see section 2.5) the planned cp-trajectory is executed through the VTL synthesiser. This creates a new training data sample for the forward and the inverse model. To keep the forward model in sync with the VTL synthesiser the learning for the forward model is continued with this new training sample together with reiterating over ten old samples to prevent total forgetting in the forward model. Depending on the desired performance the planning and learning steps can be repeated several times to improve the planned cp-trajectories. In our experiments we do this 40 times, which gives a good compromise between model accuracy and computation time.

In order to quantify the increase in intelligibility of the produced acoustics due to the semantic classifier, we set up two experimental conditions. In the control condition, we use the original female recordings from our test set and start the planning process. However, we cut the connection to the classifier and optimize solely for the target acoustics by minimizing the spectrogram MSE, the jerk and the velocity loss. After finishing the planning process we use the final produced acoustic and map it to the semantic space. In the classifier condition, we make use of the classifier during planning. Like described above, we optimize for the acoustics and for distance in semantic space. To see whether and how much the classifier adds to intelligibility we compare both conditions by looking at the loss of the final produced acoustics and their predicted semantic vectors. All models are implemented in PyTorch 1.8 and trained with double precision.

## 2.2 Simulator / Articulatory speech synthesis

For the articulatory speech synthesis we use the VocalTractLab synthesiser version 2.3. The VTL models a three dimensional vocal tract shape, i.e. the tongue, the jaw and the lips, together with some properties for the glottis and the subglottal pressure. From the geometrical shape changes the VTL derives pressure differences and the emitted audio as a 44,100 Hz mono signal.

For the acoustic simulation some branching cavities like the nasal cavities are coupled to the cavity of the mouth.

In the configuration used in this study the VTL has 30 input cps that are defined every 110 audio samples (2.5 ms) and interpolated in between. The cp-trajectories define only the moving parts of the simulated human speech organ. The size of the oral cavity, the jaw and the teeth are defined in a speaker file. We used the JD2 speaker that comes with the software.

Furthermore, in version 2.3, VTL introduced an API method to create cp-trajectories from phone segment data. Therefore it is possible to give input phone sequences with their corresponding durations to create cp-trajectories and synthesise audio from them. This interface relies on the gestural scores defined for the JD2 speaker and uses a blending method to move the articulators from one gestural target position to another one.

### 2.3 Forward and inverse model

The *forward model* shortcuts the VTL simulator, but has the same inputs and predicts the same acoustic representation. Due to its nature of being a learned sequence-to-sequence model, it will not learn all associations possible with the VTL simulator, but to approximate a cp-trajectory to log-mel spectrogram mapping. The encoding of learning history allows the model to locally approximate and generalize the VTL simulator. Here the second important property of the forward model comes into play. As the forward model is implemented in fully differentiable form, an error in the acoustical output domain can be backpropagated along the gradients of the local approximation and cp-trajectories can be adjusted according to the local backpropagated error. Furthermore, a single execution of the forward model together with the backpropagation of the error is around 100 times faster than a forward simulation with the VTL simulator.

The forward model is implemented as a recurrent sequence to sequence LSTM model. After an onset dimension is added to help the weight initialisation of the recurrent neural network, in the first layer velocities and accelerations of the input cp-trajectories are deterministically calculated. This brings the 30 input dimensions to 91 dimensions that serve as input for a 512 cell two-layer LSTM-network. The output of the LSTM-layers are then linearly mapped to 60 dimensions and the sequence length is halved with an average pooling layer. Three consecutive 5 (time) by 3 (mel frequency) convolutions are applied in a skip layer fashion. Finally, a ReLU activation function collapses all negative values to zero which serves as the prediction of silence at this mel frequency and time coordinate.

The *inverse model* is only used once per resynthesis but still has the important function of generating an initial cp-trajectory or enabling the model to create a first guess for a given target acoustics. This is especially important as the forward model only locally approximates the gradients of the VTL simulator and therefore the initialisation need to bring the cp-trajectories already near an eligible initial cp-trajectory.

The inverse model is implemented as a recurrent sequence to sequence LSTM model. The model is implemented similar to the forward model with the difference, that the velocities and accelerations of the input log-mel spectrogram lead to 181 input dimensions to the two-layer LSTM. The skip connection is convolved with seven stacked 1D convolution layers each with a kernel size of 5. No ReLU activation function is applied in the end, as the cp-trajectories can have negative values.

Both models are trained independently using the ADAM optimizer [5] with an initial learning rate of 0.001 and default parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$). In both cases we minimized the MSE loss with a batch size of 8 for 150 epochs.

### 2.4 Classifier

The *classifier* maps a 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. When we talk about an word embedding we refer to real valued array containing the activation of a hidden layer in a embedding model. Similar words with similar meanings are assumed to occur in similar contexts which will motivate the model to have similar activation in its hidden layers. The assumption is that these activations capture rich information about the meaning and relationship of words and give a low-dimensional embedding. Each word embedding is therefore considered to be a semantic representation in some form of semantic space. This embedding space is often referred to as a vector space calling the embeddings semantic vectors. We are aware that using this term is problematic because it is not clear whether the embedding space satisfies all axioms of a vector space e.g. the existence of an additive inverse. We also know that using Euclidean distance might therefore be an erroneous assumption. We believe employing the Euclidean distance as a metric is justified since the similarity of words in terms of co-occurrences seems to be consistent with an Euclidean semantic space [6].

The classifier is also an LSTM model with an input layer of 61 units, corresponding to the 60 mel banks of the log-mel spectrogram and one onset dimension. The input is fed into two stacked LSTM layers with 800 hidden units each. The output of the last LSTM layer is further processed by a hidden layer with 8192 cells and a leaky ReLU activation function. In the end the output is linearly mapped to 300 output units corresponding to the prediction of word embedding. The model was trained using ADAM [5] with an initial learning rate of $10^{-5}$ and the default parameters. We minimized the MSE loss between output and the target embedding vectors with a batch size of 8. We ran the learning process for 200 epochs.

### 2.5 The planning Loss

The planning loss is an additive loss that minimizes aspects of the cp-trajectory as the input, the discrepancy between the predicted and the target acoustics and the discrepancy between the predicted and the target semantics. This ensures jointly targeting the plausibility of the shape of the cp-trajectory and the acoustical imitation while keeping the semantics stable. The weights between the different components is adaptively calculated based on the initial cp-trajectories estimated by the inverse model.

The loss on the input cp-trajectory keeps the cp-trajectory as stationary as possible (velocity loss) and as low force as possible (jerk loss). This leads to trajectories that either do not move or move with a relatively constant force.

The MSE loss between the target and predicted log-mel spectrogram enables the framework to imitate the acoustical pattern of the target acoustics. The MSE loss between target semantics and predicted semantics should keep the meaning intact. Minimizing all these losses at once is possible due to the access of the gradients in all these models.

### 2.6 Initial training data

For all model we made use of the GECO corpus [7] in which each sample corresponds to one single German word cut out of a conversation between two native female German speakers. For the predictive and inverse model we derived cp-trajectories from phone segment data of the GECO corpus for the first 5000 words (sequence length between 246 and 1976 time steps). Both models were trained with log-mel spectrograms computed on speech synthesised by the VTL with the derived cp-trajectories. Synthesising speech from segment data was introduced in

VTL 2.3 and is accomplished approximately as described in [8]. 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 were calculated on a magnitude spectrum. Calculations were conducted with `librosa` (version 0.8.0), wave forms were sampled at 44,100 Hz. The log-mel spectrogram was linearly transformed into the 0 to $\infty$ range where 0 corresponds to silence and 1 is a loud and clear tone. In addition, as a simple form of noise reduction we subtracted the minimum value of each log-mel spectrogram from all values. Data per time slice was reduced from 220 samples to 60 samples. We ended up with 1805 unique words from 5000 word utterances from which 4500 were randomly assigned to the training set, and 500 samples to the validation set. For the classifier we extracted the audio of the first 30000 words (sequence length between 33 and 1777 time steps). As a first try, we took the three most frequent words, namely *aber*, *also* and *oder*. In order to ensure the classifier to be in sync with the acoustics the simulator produces we enriched the dataset by a segment based resynthesised version of each word. We ended up with 6634 word utterances for which we calculated the log-mel spectrogram. We randomly assigned 3460 samples to the training set and 1174 samples to the validation set. For the test set we extracted 30 new utterances for each word from unseen GECO data and matched them with the segment based synthesis giving us 180 test utterances in total. For the target output of the classifier we used the pre-trained word embedding vectors from fastText [9] using a continuous bag of words model (CBOW) trained to predict words in a local context window. The vectors used were extracted from a model trained on the German version of Common Crawl and Wikipedia using position-weights, with 300 hidden dimensions, character 5-grams, a window of size 5 and 10 negatives samples per word.

# 3 Experiments and Results

## 3.1 Forward Model & Inverse Model

We tested the forward model after 150 epochs of training on the 90 synthesised samples of the test data. We achieved an average MSE loss of $4.95 \times 10^{-4}$ between predicted and produced log-mel spectrogram given the cp-trajectories as input. We obtained similar results for the Inverse model with an average MSE loss of $5.21 \times 10^{-4}$ between predicted and target cp-trajectories given the log-mel spectrogram of the synthesised audio in the test set.

## 3.2 Classifier

After training for 200 epochs the classifier achieved an average MSE loss of $8.57 \times 10^{-5}$ on the test set. With respect to the pairwise Euclidean distances (figure 1) and an UMAP embedding [10] (figure 2) we see clear clustering and closeness of the predictions to the target vectors. We manually checked seven cases in which the predicted vector are far off their target embedding. All cases were recorded utterances indistinguishable by the human ear. We are inclined to dismiss them as misalignments.

## 3.3 Recurrent gradient based planning

Due to long computation time we only focused on the 30 recorded *also* examples of the test dataset for which we calculated the final MSE loss between produced and target acoustics as well as the MSE loss between predicted and target semantic vector. We also compared the results involving planning with the segment based synthesis. 6 files with inferior audio quality were excluded from further analyses. Figure 3 shows the results. Looking at the MSE loss for

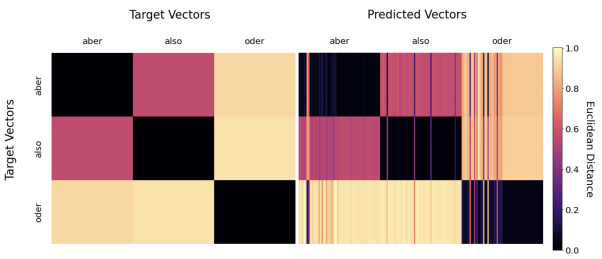


**Figure 1** – Pairwise Euclidean distance between true and predicted embedding vectors for all 180 test utterances sorted by words. Each row and column corresponds to one single utterance.

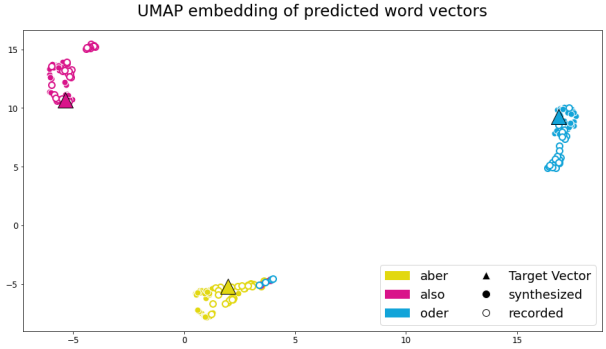


**Figure 2** – UMAP embedding of predicted semantic vectors of the test set utterances.

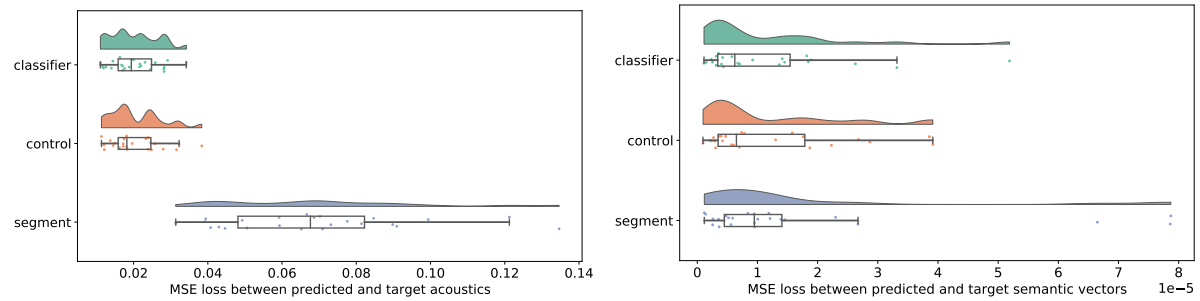


**Figure 3** – Final MSE loss distributions for produced acoustics, on the right, and semantic embeddings on the left [11]. Both plots show the results for the original test utterances of *aber*. After inspecting the data we removed 18 data points for six utterances because of indistinguishable audio.

the produced acoustics (left panel) we see an improve in MSE loss using the planning framework in both conditions. Planning and optimizing solely for the acoustics like in the control condition achieves the lowest MSE loss which is what we expected. Still, we see comparable results in the classifier condition. Looking at the MSE loss in the semantic space (right panel) for the produced acoustics we see low MSE losses in all cases. The low MSE loss for the segment based synthesis can be explained by the classifier being trained on synthesised examples. Comparing the control and classifier condition we do not see a clear improvement by using the classifier during the planning process with respect to intelligibility of the produced acoustics measured as distance in the semantic space. However, with respect to the results of the classifier on the test set (figures 1 and 2) we see an overall low initial MSE loss. Using this MSE loss in an additive fashion might have had little effect on the planning process. However, looking at the quantiles for the classifier condition and with respect to the small test size we are still confident that our framework works and see these results just as the first small steps in the direction of optimizing for general intelligibility.

## 4 Conclusion and future work

This is the first work using the dissimilarity in a vector space embedding to inform the planning process of the control parameter of an articulatory speech synthesis simulator. Explicitly using the semantic dissimilarity seem to help improving or at least does not harm the semantic similarity of an acoustic resynthesis. Unfortunately, the computational time needed for the resynthesis using the semantic loss is very high. For 300 ms of speech it needs one to three hours to plan a trajectory. And even after this time the resulting trajectory could be improved with the same method substantially by simply planning it longer.

Future plans include making the computations faster, applying the method to a bigger vo-

cabulary and implementing a discrimination loss. To scale the framework to a bigger vocabulary the classifier will be swapped with one that is trained on a whole lexicon. The idea here is to preserve the language statistics in the training data of the classifier, i. e. having a lot of low frequency words and a few high frequency words. The hope here is to get an informative semantic planning for high frequency or known words that help the model to articulate these words in its own way, in contrast to unknown words where it has to rely only on the acoustic imitation.

The other next step is to change the semantic loss from optimizing for a target semantics to not confusing the target with a similar but unintended word. With the discrimination loss the model plans cp-trajectories that try to get close to a target acoustics while at the same time try to maximize the distance to unwanted semantics. One final goal would be to enable the model to produce some speech for any given semantic vector.

**Acknowledgments:** This research was supported by an ERC Advanced Grant (no. 742545).

## References

[1] SERING, K., P. SCHMIDT-BARBO, S. OTTE, M. V. BUTZ, and H. BAAYEN: *Recurrent gradient-based motor inference for speech resynthesis with a vocal tract simulator*. In *12th International Seminar on Speech Production*. 2020.

[2] 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.

[3] PARRELL, B., V. RAMANARAYANAN, S. NAGARAJAN, and J. HOUDE: *The facts model of speech motor control: Fusing state estimation and task-based control. PLoS computational biology*, 15(9), p. e1007321, 2019.

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

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

[6] HASHIMOTO, T. B., D. ALVAREZ-MELIS, and T. S. JAAKKOLA: *Word embeddings as metric recovery in semantic spaces. Transactions of the Association for Computational Linguistics*, 4, pp. 273–286, 2016.

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

[8] SERING, K., N. STEHWIEN, Y. GAO, M. V. BUTZ, and H. BAAYEN: *Resynthesizing the geco speech corpus with vocaltractlab. Studientexte zur Sprachkommunikation: Elektronische Sprachsignalverarbeitung 2019*, pp. 95–102, 2019.

[9] 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.

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

[11] ALLEN, M., D. POGGIALI, K. WHITAKER, T. R. MARSHALL, and R. A. KIEVIT: *Raincloud plots: a multi-platform tool for robust data visualization. Wellcome open research*, 4, 2019.

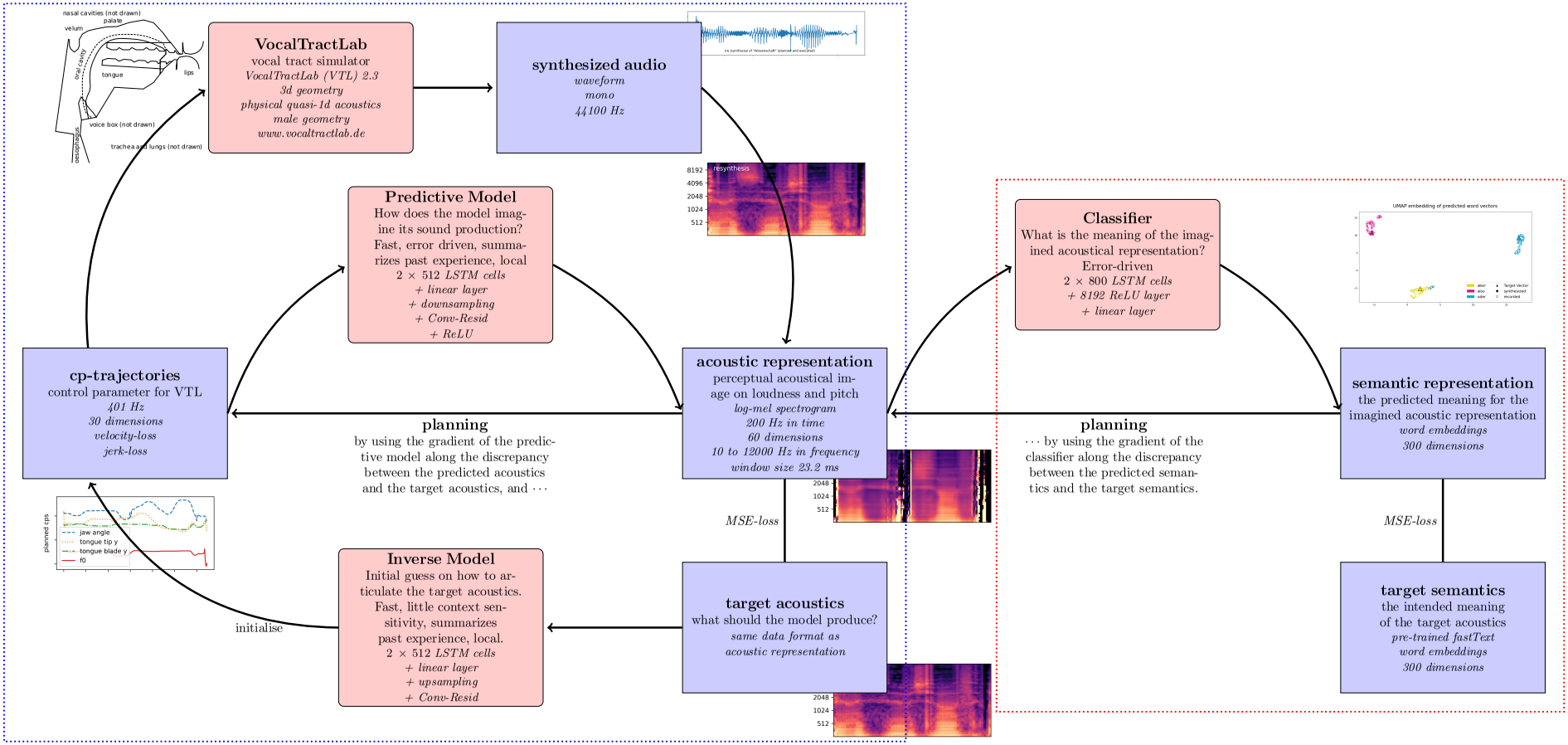


**Figure 4** – The Recurrent Gradient-based Motor Inference Model for Speech Resynthesis with semantic discrimination using the VocalTractLab synthesiser. The blue dotted box represents the network architecture used in [1], and the region bound in the dotted red box represents the contribution of the present study to the whole network. The main idea is using the discrepancy between the predicted semantic vector of an imagined acoustical representation and a target semantic vector generated from a target acoustics. The discrepancy is first back-propagated to the imagined acoustic representation and then further back to the cp-trajectories in order to adjust the initial control parameters for the VTL synthesiser.