Report

A replication report for the original Earshot paper results

Linkai Peng[1]

[a]*https://github.com/vocaliodmiku*

1. Introduction

This report is designed especially for reproducing the results presented in the Earshot paper [1]. We hope to outline the complete workflow for training and testing the model to align current implementations with original paper's finding from scratch.

2. Code

Our implementation is based on the official Github repository. Thanks for the author's contributions to open-source community by maeking both the dataset and code publicly available. A seperate branch will be created after the completion of this reproducibility study. Note that the original code did not set a random seed explicitly. To enhance reproducibility, we will introduce a parameterized random seed mechanism in our modified version.

3. Hardware and Software

The training framework requires NVIDIA GPUs with CUDA support. As the original research was conducted in 2020 using tensorflow 1.X, users must make sure have the compatible python, cudatoolkit, and tensorflow installed (e.g. cudatoolkit == 10.0.130; tensorflow == 1.15; Python == 3.6.13).

4. Methods

4.1. Loss function

According to the statement in [2]

> The semantic output activation O was derived using the following equation.
>
> $$O_t = \sigma(H_t W_{HO} + b_0) \quad (1)$$
>
> where σ is sigmoid function.

The loss function should be:

```
tf.nn.sigmoid_cross_entropy_with_logits
```

Code 1. Loss function.

It is not recommended to modify the loss function directly. A preferable approach is to identify the specific commit where the change occurred and revert to that commit. Then move the current Git HEAD to *Commit 0c5c1c9*, which is one commit ahead of *Commit 1d6abe6*.

4.2. Data

The Github repository provides a pre-genearated input pattern Pattern_Dict.IM_Spectrogram.OM_SRV.AN_10.Size_10000.WL_10.pickle. This contains all necessary materials required by the main training script 'Earshot.py'. Training can stiill proceed without the corresponding wavfiles.

4.2.1. Genrate Pattern for audio files

When introducing a new dataset, patterns must still be generated using the provided script, Pattern_Generator.py. Note that the pattern files produced by this script do not adhere to the Pattern_Dict.IM_*.OM_*.AN_*.Size_*.WL_*.pickle name format. However, after careful inspection, this discrepancy does not affect functionality.

4.3. Determinism

Currently, reproducible results are not firmly guaranteed. We hope that the randomness existing in the entire training process can be controlled by a random seed number. To achieve this, we must make sure (1) the dataset have a deterministic order before the shuffle operation; (2) model parameters are initialized with given random seeds, and the operation performed on parameters must also be deterministic.

Following these priciples, *commit 8673cf8* implements deterministic GPU training successfully. Note that tensorflow 1.15 itself is not determinisitc, but fortunately, a 3rd-party patch named tf_determinism comes from NVIDIA enables it. As a result, an additional parameter named 'seed' is introduced to the training script 'Earshot.py'. Once a number is set as the random seed, the model always yields the same result, regardless of how many time it runs.

4.4. Check List

Table 1 compares the hyperparameters used in the code implementation with the values reported in the paper.

Table 1. Astronomical Object Data

Parameter	Code	Paper
Total # of Words Segment	8100	8100
Mini batch size	1000	2000
Baseline learning rate	0.002	0.002
LR scheduler	Noam	Noam*
Optimizer	Adam	Adam*
# of LSTM hidden nodes	512	512
Length of SRV	300	300
SRV assign number	10	10
Spectrogram dimension	256	256

*: Their hype-parameters are identical.

5. Model Training

The readme file provides a detailed explanation of the parameters. To reproduce the paper's results, we need to train 10 models, each responsible for training with data which completely excludes one speaker. Code 2 demonstrates training and testing with the speaker 'Vicki' excluded.

```
spk="vicki"
id=0
seed=39
res=./Results/IDX_$id/HM_LSTM.H_512.EM_M.ET_${spk^^}.IDX_$id
# Epoch 0 - 8000
python EARShot.py \
    -ht LSTM \ # LSTM cell
    -hu 512 \  # Number of LSTM hidden nodes
    -tt 1000 \ # The frequency of checkpoint saving during learning
    -se 0 \    # Set the model's start epoch. 0 means train
    -me 8000 \ # Set the ending epoch of the model
    -em M \    #
    -id $id \
    -et $spk \  # Set which talker pattern is excluded from the learning.
    -seed $seed

# Epoch 8000 - 10000
python EARShot.py \
```

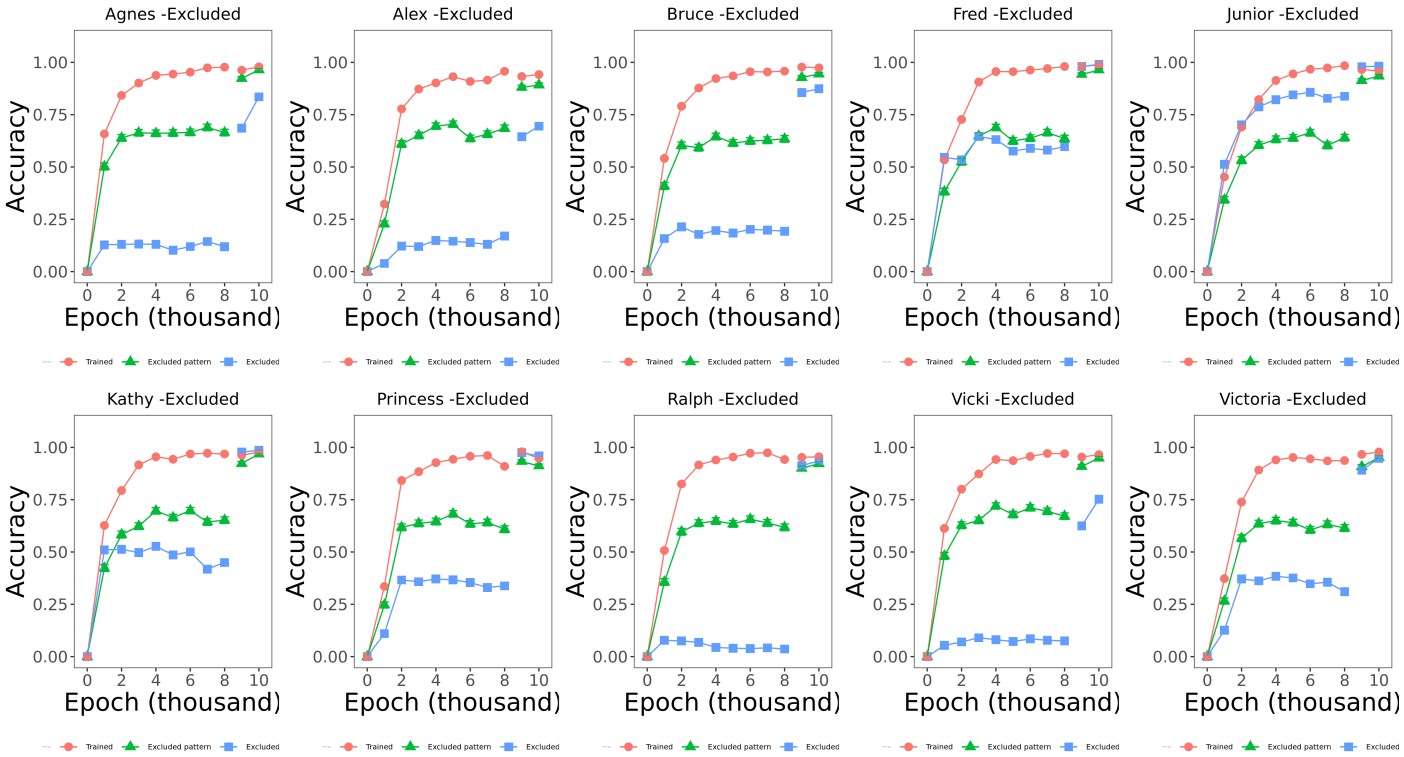

Figure 1. Performance of 10 reproduced models. Even though the 2020 papar didn't show this plot, it is still very useful to help diagnosing models.

```
    -ht LSTM \
    -hu 512 \
    -tt 1000 \
    -se 8000 \ # This parameter and the 'mf' parameter
     ↪ must be set when loading a previously learned
     ↪ model.
    -me 10000 \
    -em M \
    -id $id  \
    -et $spk \
    -mf $res/Metadata.pickle \
    -seed $seed \
    -ei         # All exclusion settings above will be
     ↪ ignored, i.e

# Calculate Accuracy and analyze result
python Result_Analysis.py \
    -d $res
```

Code 2. The training script 'vicki.sh' for the model of speaker vicki.

After completing the training of 10 models, an R script can be used to generate plots of models' individual and overall performance. Figure 1 shows the individual performance of each model. Figure 2 shows the performance of the models organized by talkers. Figure 3 is the paper version of Figure 2. They are very similar to each other.

5.1. Resume training from a specific checkpoint

Sometime, the model's training does not go smoothly. The training accuracy might crashes at certain time steps. In such case, you can restore trianing from the last checkpoint before the crash. Code 3 provides an example of how to do this.

```
# Load checkpoint at specific Epoch and retrain
    ↪ subsequent epochs
python EARShot.py \
    -ht LSTM \
    -hu 512 \
    -tt 1000 \
    -se 4000 \ # Set the model's start epoch, e.g. 4000
    -me 8000 \
    -em M \
    -id $id \
    -et $spk \
    -mf $res/Metadata.pickle \
    -seed $seed
```

Code 3. The training script 'vicki.sh' for the model of speaker vicki. Restore the model to the previous checkpoint when accuracy crashes

6. Results

Evaluation is easy in the project. R scripts located in the repository are clear. Figure 4 shows the reproduced models' performance averaged over ten models. Figure 5 and Figure 6 show the reproduced PSI and FSI result. Data come from the 'Junior-Excluded' model at epoch 10000 with a criterion value of 0.14. The subsequent RSA results are presented in Figure 7, Figure 8, Figure 10, and Figure **??**

7. Random seeds used for this report

Agnes: E.0-3000:1025 3000-7000:1025 7000:326 8000:24584 9000-10000:25269

Alex: E.0-4000:26 4000:98 5000:19436 6000-7000:28647 8000:1015 9000-10000:2926

Bruce: E.0-3000:32 3000-9000:56 9000-10000:9852

Fred: E.0-1000:38 1000-6000:56 6000:98 7000-10000:108

Junior: E.0-1000:85 1000-5000:56 5000-8000:38 8000-10000:29961

Princess: E.0-3000:98 4000:19562 5000:2658 7000-10000:32559

Ralph: E.0-6000:1024 7000-10000:10241

Kathy: E.0-10000:?

Vicki: E.0-1000:35 1000-8000:295 8000-9000:6523

Victoria: E.0-5000:1025 5000-8000:248 8000:98 8000-10000:25297

8. MSIC

Results are not completely the same with the original one, e.g., the maximum value of PSI and FSI are different from the one reported in paper. A new phoneme feature file 'Phoneme_Feature.Paper.re.txt' has been uploaded to address it but still not guranttes identical results.

 Creative Commons CC BY 4.0

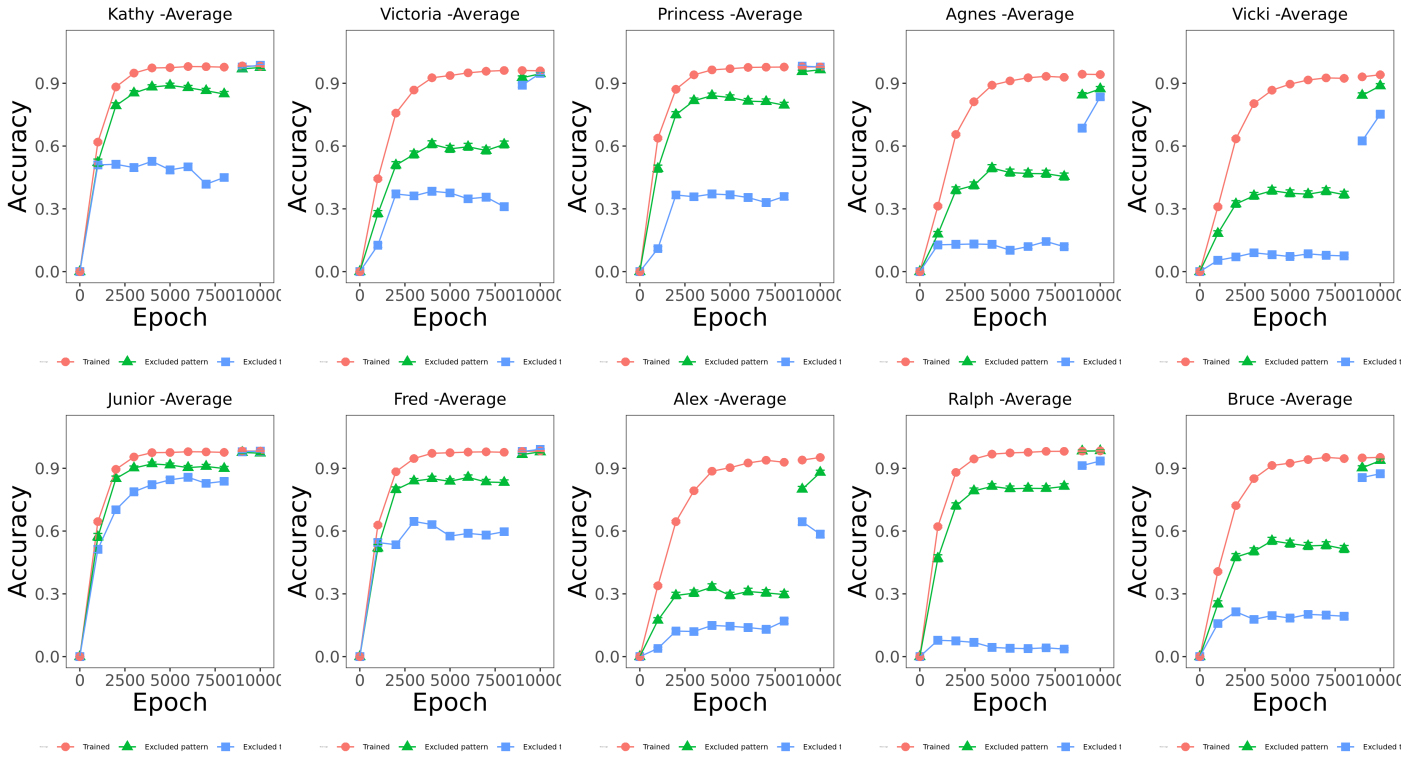

Figure 2. Performance of 10 models organized by talkers. In each panel, "Trained" indicates the performance on the listed talker (e.g., "KATHY") in the 9 simulations where it was included. "Excluded words" indicates performance on the listed talker's 100 excluded words in the 9 models that included that talker in training. The "Excluded talker" lines track performance on the listed talker when it was excluded.

The results differ slightly from the original findings, for instance, the maximum value of PSI and FSI vary from those reported in the paper. A new phoneme feature file, 'Phoneme_Feature.Paper.re.txt', has been uploaded to address this, though it does not guarantee identical results.

References

[1] J. S. Magnuson, H. You, S. Luthra, *et al.*, "Earshot: A minimal neural network model of incremental human speech recognition", *Cognitive science*, vol. 44, no. 4, e12823, 2020.

[2] *Supplementary materials for paper - earshot: A minimal neural network model of incremental human speech recognition. cognitive science, 44, e12823.* [Online]. Available: https://magnuson.psy.uconn.edu/wp-content/uploads/sites/1140/2020/04/earshot_CS_brief_report_REV_2020.04.10_supp.pdf.

Creative Commons CC BY 4.0

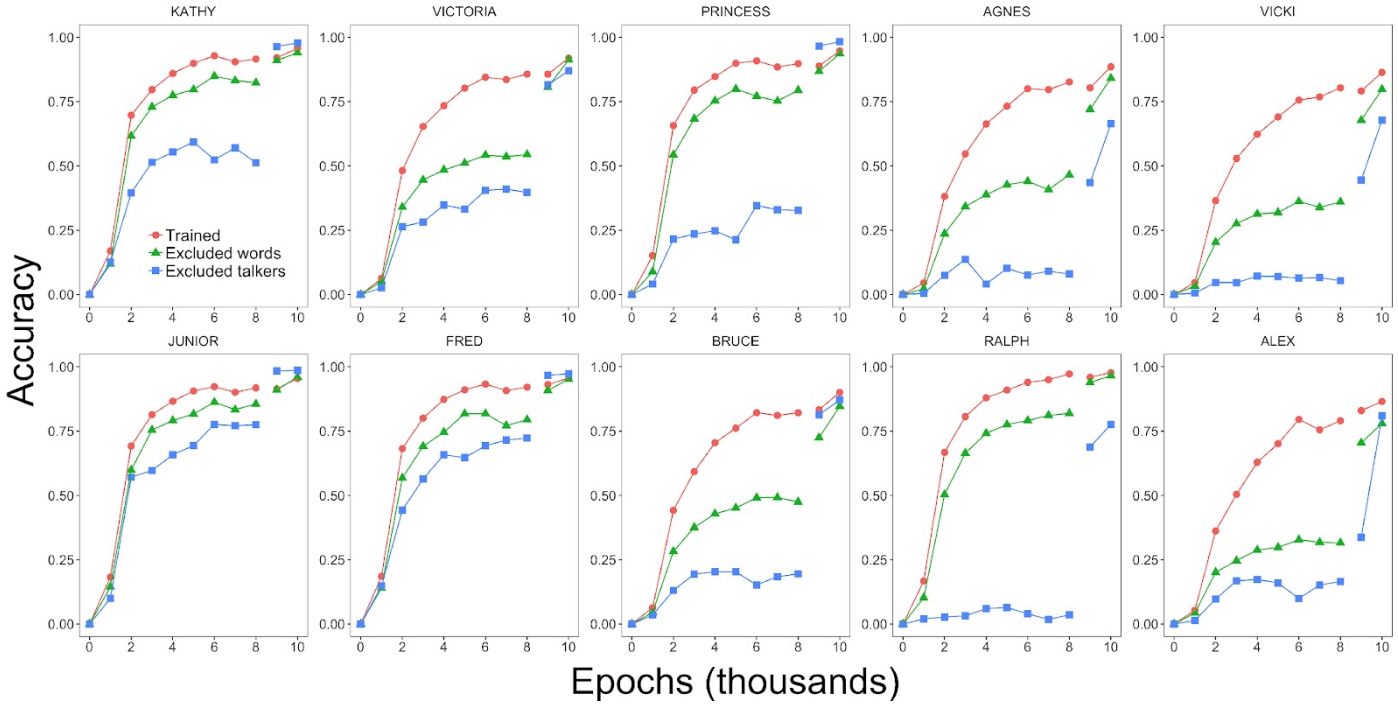

Figure 3. Performance of 10 models organized by talkers. Comes from Paper.

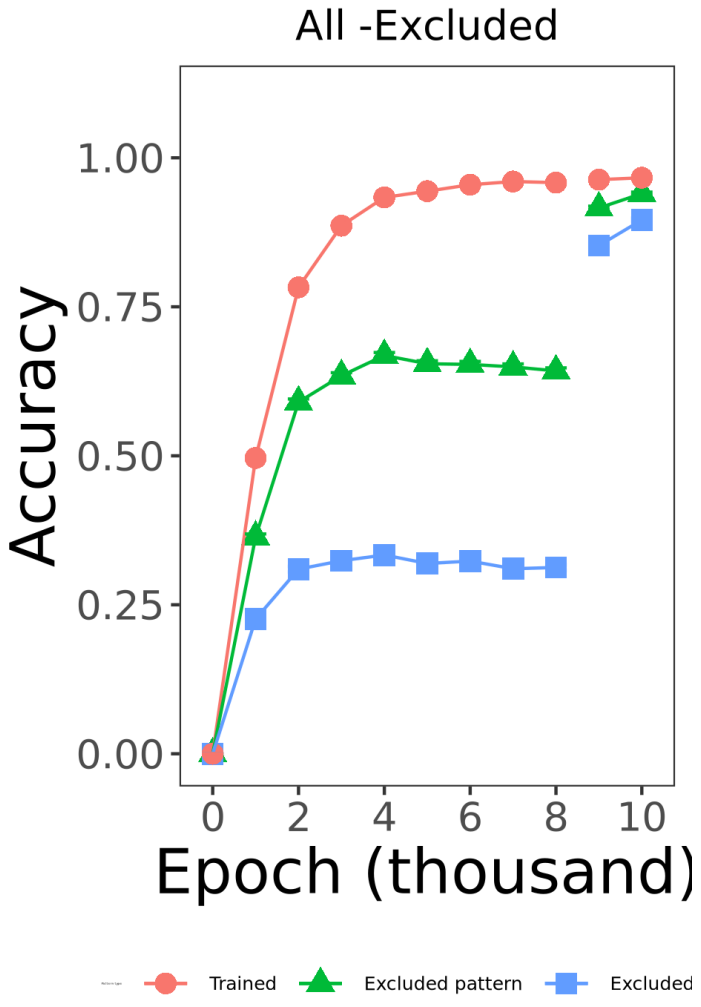

Figure 4. Reproduced Model performance. Accuracy by epoch averaged over ten models.

 Creative Commons CC BY 4.0

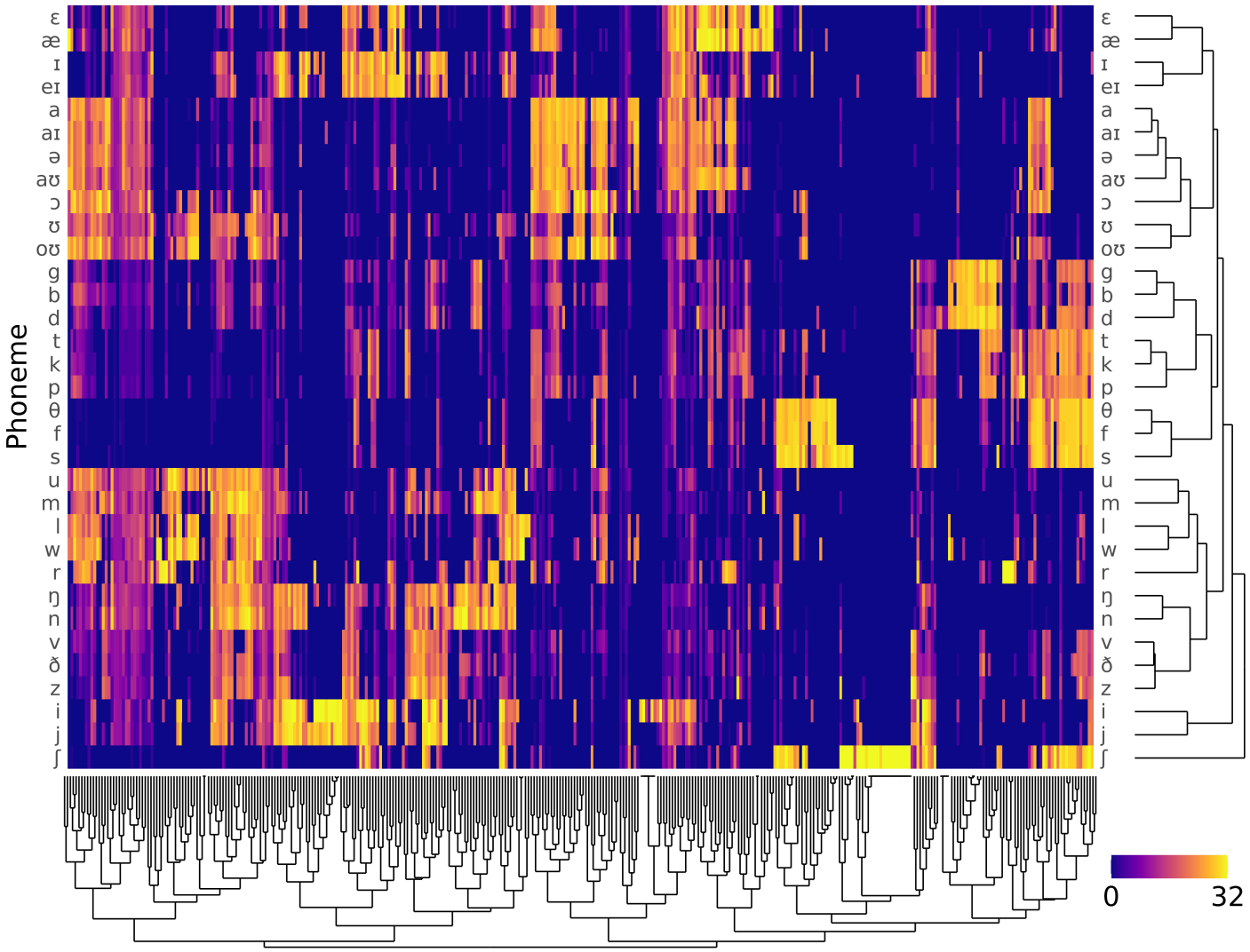

Figure 5. Reproduced phonetic sensitivity plot.

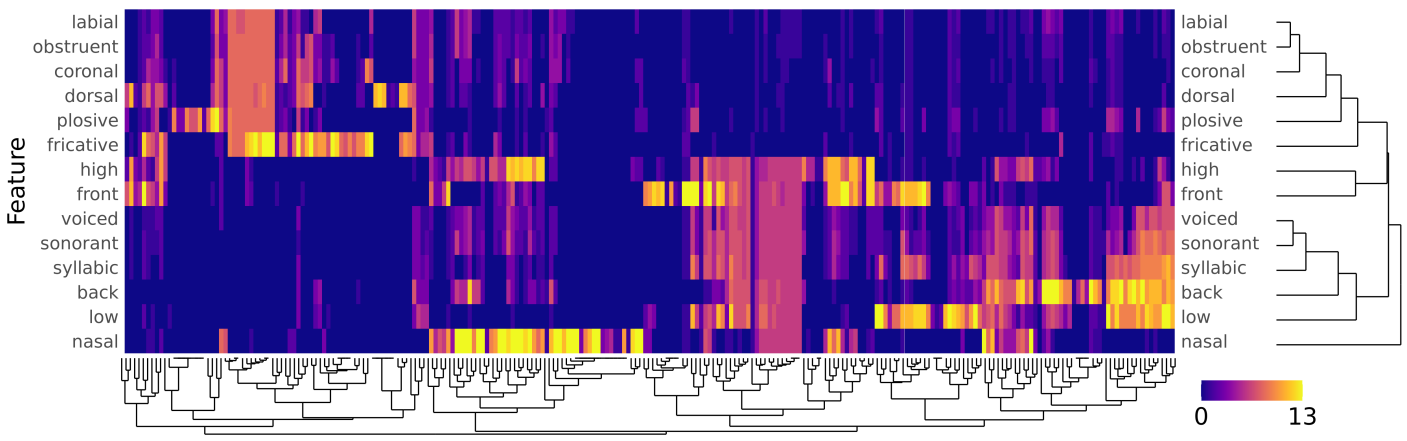

Figure 6. Reproduced feature sensitivity plot.

Creative Commons CC BY 4.0

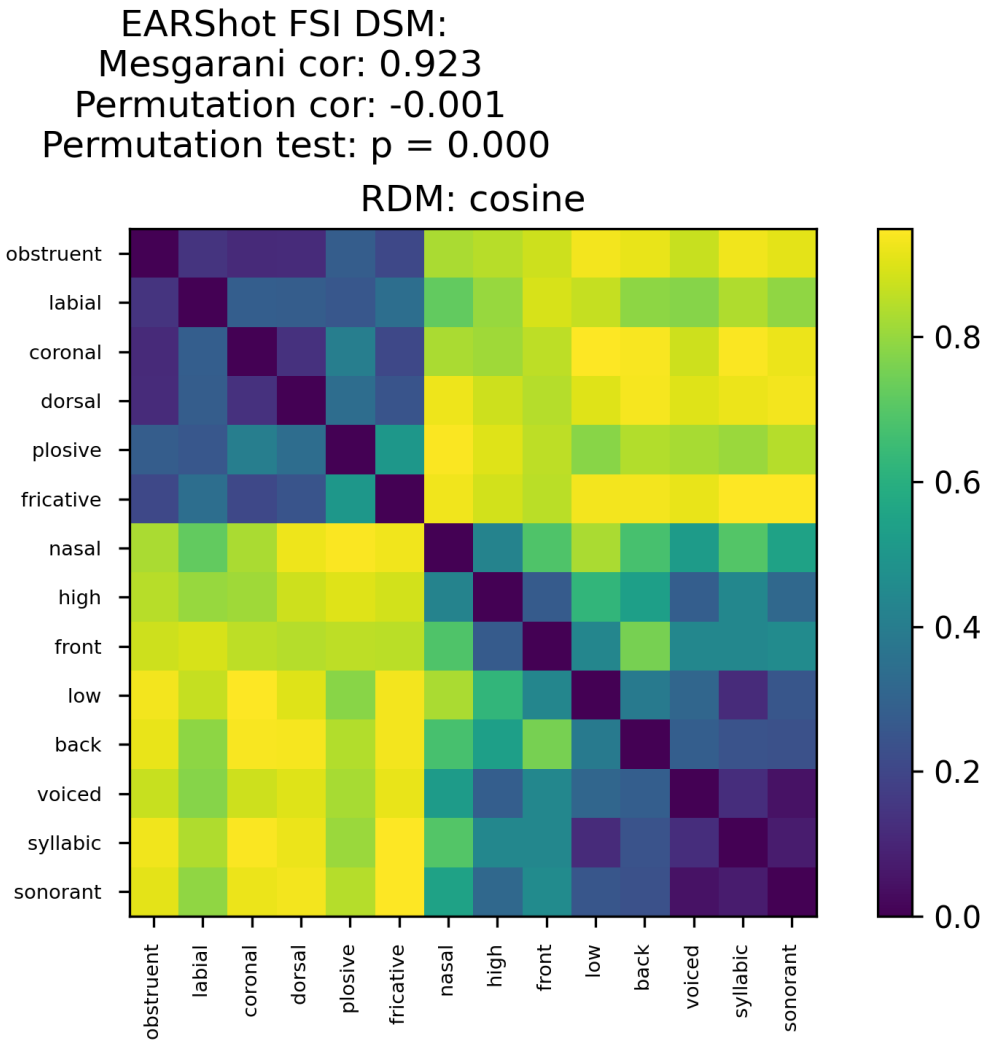

Figure 7. Reproduced RSA (A) plot.

 Creative Commons CC BY 4.0

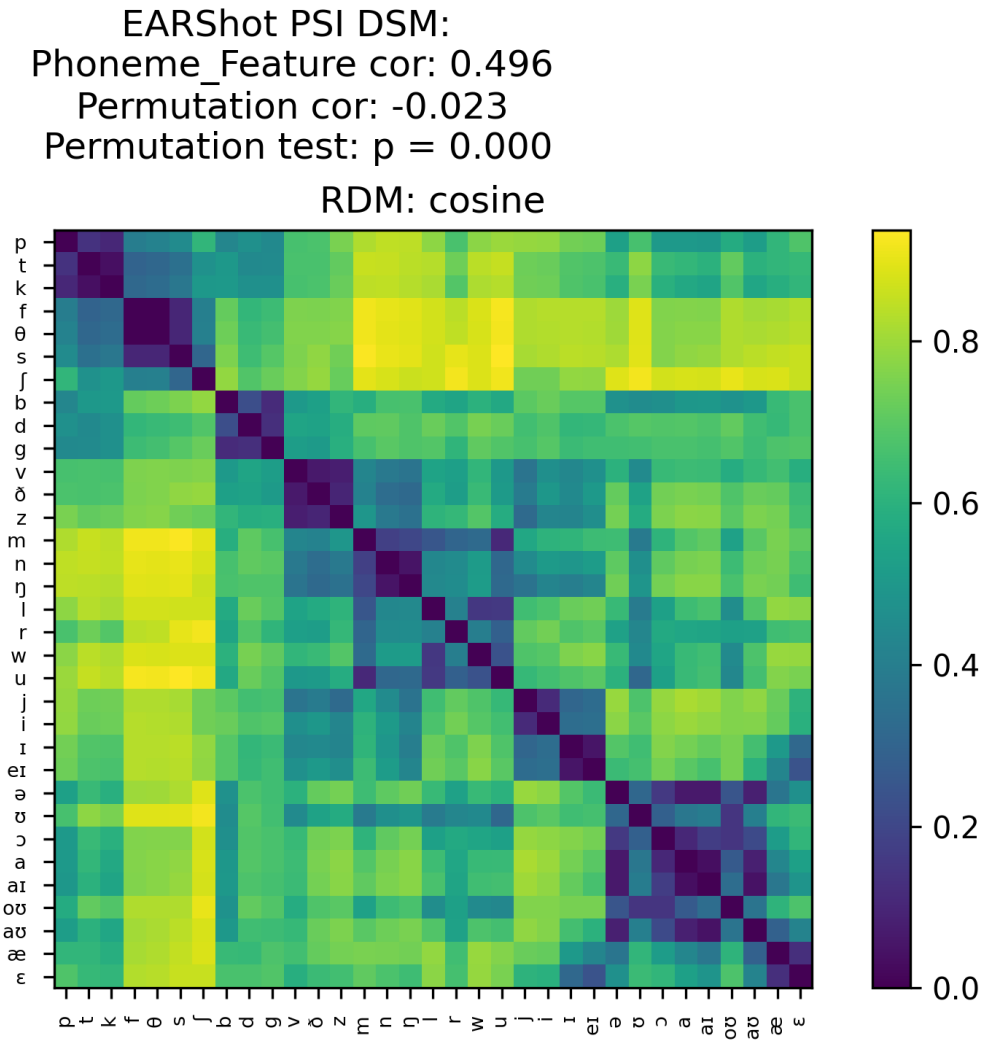

Figure 8. Reproduced RSA (B) plot.

Creative Commons CC BY 4.0

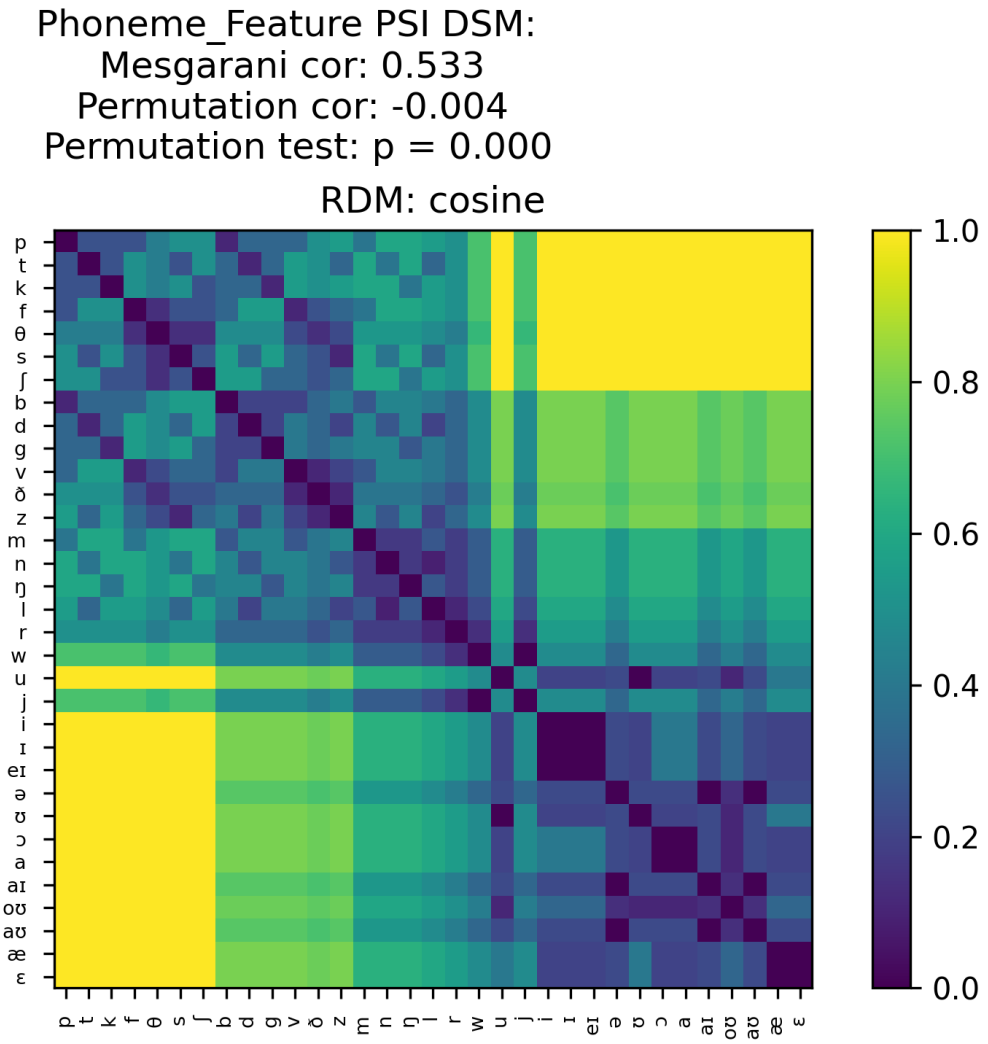

Figure 9. Reproduced RSA (C) plot.

 Creative Commons CC BY 4.0

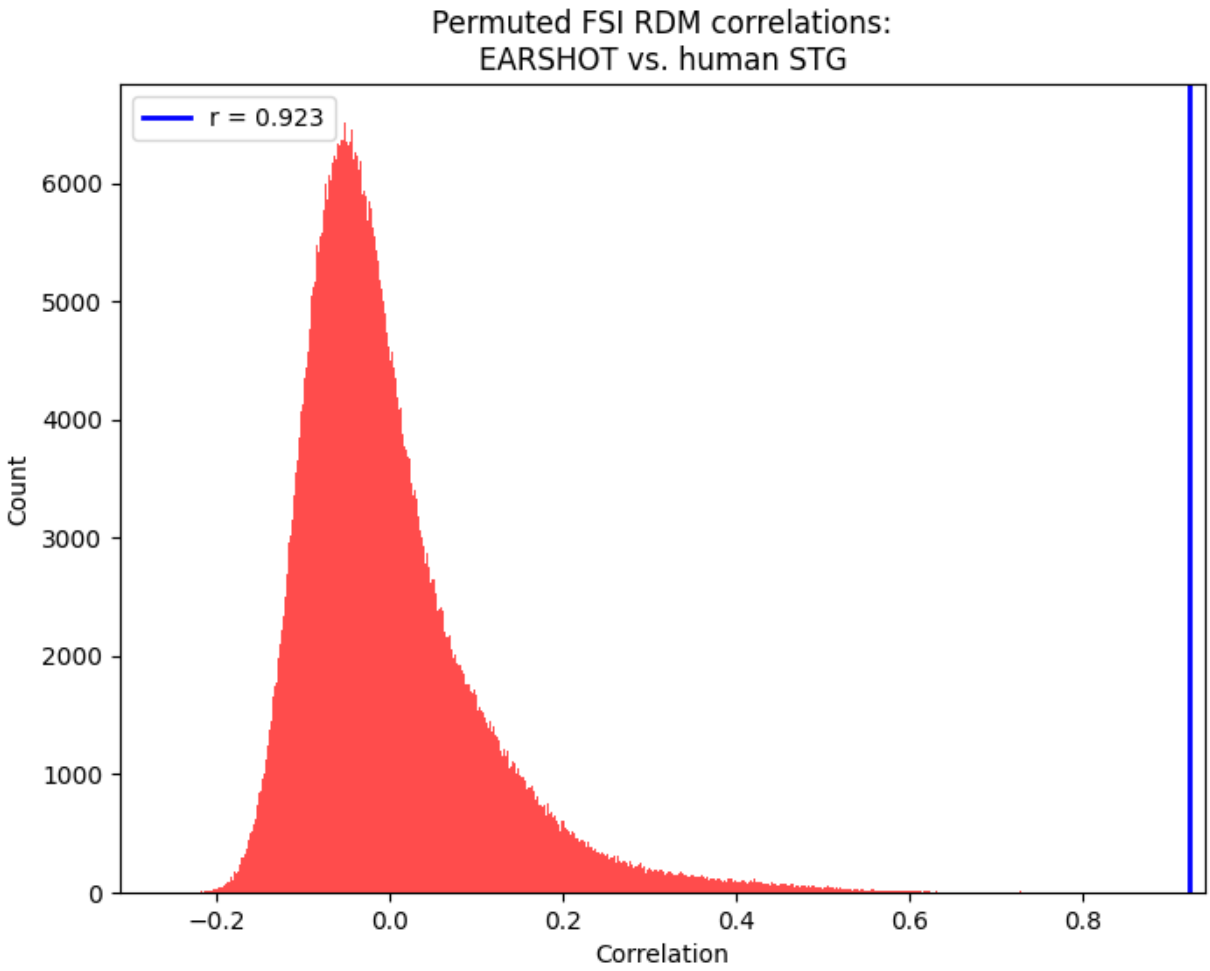

Figure 10. Reproduced plot: Permuted FSI RDM correlations: EARSHOT vs. human STG.

Creative Commons CC BY 4.0