Replicating Human Sound Localization with a Multi-Layer Perceptron

Eric Michael Sumner
University of Iceland
ems36@hi.is

Runar Unnthorsson
University of Iceland
runson@hi.is

Morris Riedel
University of Iceland & Jülich Supercomputing Centre
morris@hi.is

ABSTRACT

One of the key capabilities of the human sense of hearing is to determine the direction from which a sound is emanating, a task known as localization. This paper describes the derivation of a machine learning model which performs the same localization task: Given an audio waveform which arrives at the listener's eardrum, determine the direction of the audio source. Head-related transfer functions (HRTFs) from the ITA-HRTF database of 48 individuals are used to train and validate this model. A series of waveforms is generated from each HRTF, representing the sound pressure level at the listener's eardrums for various source directions. A feature vector is calculated for each waveform from acoustical properties motivated by prior literature on sound localization; these feature vectors are used to train multi-layer perceptrons (MLPs), a form of artificial neural network, to replicate the behavior of single individuals. Data from three individuals are used to optimize hyperparameters of both the feature extraction and MLP stages for model accuracy. These hyperparameters are then validated by training and analyzing models for all 48 individuals in the database. The errors produced by each model fall in a log-normal distribution. The median model is capable of identifying, with 95% confidence, the sound source direction to within 20 degrees. This result is comparable to previously-reported human capabilities and thus shows that an MLP can successfully replicate the human sense of sound localization.

1. INTRODUCTION

The Acoustic and Tactile Engineering Lab (ACUTE) of the Icelandic EuroCC National Competence Center [1] for Artificial Intelligence and High-Performance Computing performs research and product development for societally relevant challenges in many applications together with its European partners (e.g., project *Sound of Vision* won the Tech for Society award in 2018 [2]). This includes the development of wearable assistive devices for visually impaired persons, cochlear implant recipients, and solutions for delivering accurate virtual acoustics.

One particular application of virtual acoustics is the development of spatial audio systems, which are designed to present realistic virtual soundscapes to the listener. In order for a spatial audio system to present a realistic soundscape, it is imperative to stimulate all aspects of an individual's sense of hearing. One important aspect of this is *'localization'*, the ability of a listener to determine the location of a sound source. This requires an accurate characterization of the listener's head-related transfer function (HRTF). A number of approaches have been proposed to estimate these individualized HRTFs, such as selecting a nearest match from a database of measured HRTFs [1] or calculating the HRTF from a numerical simulation of the indivdual's head shape [2]. Unfortunately, they all produce results that are significantly worse than a direct measurement taken in an anechoic chamber [3]. Developing a better HRTF estimation method for use in spatial audio systems is an ongoing project within ACUTE.

A major challenge in this project is evaluating the effectiveness of the HRTF estimates produced. One approach is to develop a model which can replicate the localization response of an individual to arbitrary audio waveforms. This model can then be presented with the waveforms produced by a candidate spatial audio system, and the intended virtual location compared to the model's output, i.e. the listener's perceived audio source direction.

This paper derives a class of individualized machine learning models for this task, based on a multi-layer perceptron (MLP) [4]. The training data for each model consists of an audio signal dataset derived from a white noise generator and an HRTF from the ITA-HRTF database [5], representing sound pressure levels (SPLs) at the individual's eardrums. Features representing both broadband and spectral interaural level difference (ILD) and interaural time delay (ITD) information are extracted from these waveforms and presented to the input layer of an MLP.

Three of these individualized models are simultaneously optimized for both accuracy and training cost, producing a generally-applicable hyperparameter configuration that can be used to train models against arbitrary HRTFs. Models are then trained for the remaining 45 individuals in the ITA-HRTF dataset to validate the hyperparameter configuration's general applicability.

The remainder of this paper is structured as follows: Section 2 provides a brief introduction to the computational models employed and the physical mechanism of sound localization. Section 3 describes the dataset, hardware, and software used for this experiment. Section 4 derives the structure of an individualized sound localization model and

[1] http://ihpc.is/community
[2] https://soundofvision.net/sound-of-vision-at-ict-2018/

describes the hyperparameter optimization process. Individualized models are then trained for 45 additional HRTFs that were not used in the derivation process; Sec. 5 presents an analysis of these models, including overall statistics and a detailed investigation of selected models. The paper concludes with some remarks about the the possible applications of these models in Sec. 6.

2. BACKGROUND

This section provides an introduction to the underlying concepts of sound localization and machine learning. Section 2.1 describes the physical mechanism of sound localization. Section 2.2 then provides a brief overview of the machine learning techniques used, i.e. multi-layer perceptrons and hyperparameter optimization.

2.1 Sound Localization

Sound localization is possible due to the acoustic filtering effected by sound waves interacting with an individual's body, especially the pinnae [6]. As incoming sound waves interact with the pinna, reflections and refractions internal to the pinna mutually interfere, altering the waveform that arrives at the listener's eardrum as illustrated in Fig. 1a. For point sources in the far field, this alteration can be represented as a direction-dependent, linear and time-invariant audio filter known as the HRTF.

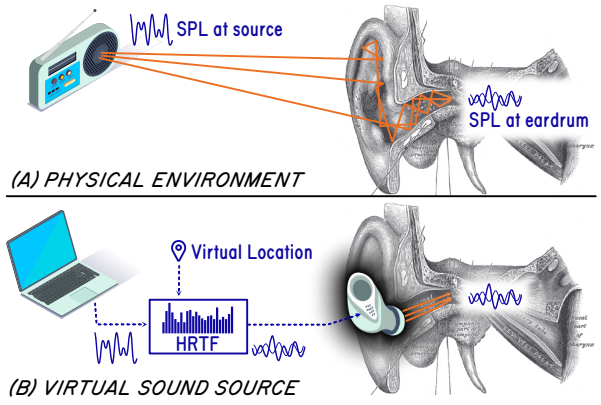

Figure 1. Acoustic filtering of the pinna and its relationship to the HRTF. Adapted from Gray, Plate 907 [7]

As the geometry of each individual's pinnæ are unique, the HRTF is also unique to each individual; determining an analytic relationship between these is a field of active research. With an accurate measurement of an individual's HRTF, it is possible to predict the SPL at each eardrum given a source audio signal and its location, as illustrated in Fig. 1b. When this prediction is played through headphones, bypassing the physical filter, the listener perceives the sound to be coming from the direction of the virtual source [8].

Prior studies have identified the acoustic properties necessary for humans to perform sound localization tasks. For sound sources on the horizontal plane, humans rely primarily on the ITD and ILD [9]. Vertical localization additionally requires spectral information in the 4-16 kHz range [10]. The frequency response of a typical HRTF exhibits a number of notches in this range, with directionally-varying center frequencies. These notches introduce local spectral gradients which the human auditory system uses to estimate the source direction [11].

2.2 Machine Learning

'Machine learning' refers to the class of algorithms that can infer structure from examples, instead of having that structure explicitly stated by a programmer. One of the earliest of these algorithms is still in common use today, the multi-layer perceptron; i.e. a fully-connected, feed-forward artificial neural network, which is often coupled to a domain-specific feature extraction stage.

An MLP is formed of an input layer, several hidden layers, and an output layer, each consisting of several *'neurons'*, or *'nodes'*. The output of each neuron is defined as a linear combination of the previous layer's output values passed through a nonlinear, univariate *'activation function'*. The linear input weights of all the neurons in the network form the parameters that will be learned during training. Training an MLP is a supervised process which requires exemplar data samples labeled with a desired output. Each example is presented to the network's input layer and the corresponding output layer values are compared with the example's desired output. A backpropagation process then determines the contribution of each parameter to this measured error. The error contribution for several examples are grouped together in a *'batch'*, and a gradient descent optimization algorithm uses this information to adjust the internal parameters such that the observed errors approach zero. In this way, the entire *'training set'* of examples is processed many times in so-called epochs, ultimately resulting in an MLP which can reproduce the sample outputs with high accuracy.

Though MLPs are universal estimators on their own, they often require large amounts of computation and training data to produce an acceptable model. When some domain-specific information is known, adding a feature extraction stage prior to the MLP can greatly reduce these costs. Instead of feeding the source data directly into the input layer of the MLP, interesting properties of the source data are calculated and formed into a feature vector. This feature vector usually has a much lower dimensionality than the raw source data, which means that fewer neurons are required within the MLP. This, in turn, means that fewer internal parameters need to be calculated during training.

Most machine learning models have a number of parameters that cannot be automatically learned from the training data. These are called *'hyperparameters'* to indicate that they need to be specified manually before training begins. These can include properties of both the domain-specific feature extraction algorithm and the MLP itself, such as the included features, the topology of the MLP's network, the activation function of each neuron, the concrete optimization algorithm used, and the halting condition, among others.

Choosing appropriate values for these hyperparameters is the primary task of a machine learning engineer, but they often have subtle and non-intuitive effects on the overall

model accuracy. In many cases, the only way to select these is a process of trial-and-error. As the number of hyperparameters increases, performing this search manually becomes intractable. Recently, a number of automated approaches to this *'hyperparameter optimization'* problem have been proposed. These use techniques like Bayesian optimization to select the optimal hyperparameter configuration to test in each trial [12].

3. EXPERIMENTAL SETUP

The model training and evaluation code are written in Python and executed in the Jupyter programming environment [13] on a 6-core Intel i7-10750H CPU. Section 3.1 describes the dataset used for this experiment, and Sec. 3.2 describes the software environment used.

3.1 Dataset

The HRTFs used for this paper come from the ITA-HRTF database [5]. This database contains head geometry information and HRTFs for 48 individuals, stored in Spatially-Oriented Format for Acoustics (SOFA) [14] files. These files consist of impulse responses sampled along azimuth, elevation, and time dimensions. Each HRTF contains $360°$ of azimuth data and $160°$ of elevation data, sampled in $5°$ increments. These 2,304 impulse responses each contain 256 samples at 44.1 kHz, for an overall frequency resolution of 172.3 Hz.

3.2 Software Environment

All of the third-party packages in this paper use NumPy [15] arrays as their data transfer format. NumPy also provides the linear algebra, random number generation, and discrete Fourier transform (DFT) [16] routines needed for the model. Additional signal processing routines are provided by Librosa [17], which is designed for audio feature extraction.

The SOFA standard specifies that files shall be in network common data form [3] (netCDF) and mandates the inclusion of metadata describing the data collection process, such as the sampling rates and properties of the emmitters and recievers used. The implementation uses the PySofaConventions [4] package to make SOFA data available as NumPy arrays.

The MLP training and evaluation routines are provided by Scikit-learn [18], and the hyperparameter optimization routines are provided by the Optuna framework [19]. The plots in this report were generated with the Plotly package for Python [5].

4. MODEL DESIGN

This section describes a class of machine learning models, each of which can replicate a single individual's sound localization behavior; Fig. 2 shows a schematic representation of the model's operation and how it relates to the HRTF data used for training and evaluation. Section 4.1 describes the overall structure and training strategy, and Sec. 4.2 details the calculation of the feature vector. Section 4.3 describes the search used to determine appropriate values for the model's hyperparameters, shown in Fig. 2 in italic.

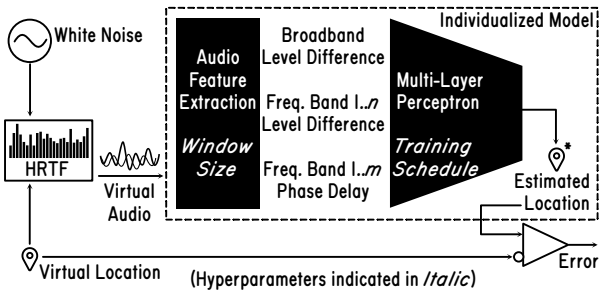

Figure 2. Structure of an individualized model and its relationship to training data

4.1 Model Design and Evaluation

Each model is trained on audio samples transformed by a single HRTF, to replicate the behavior of that individual. All of the source directions $\hat{y}$ present in the HRTF are extracted, and 20% are reserved for the testing (evaluation) set $\mathbf{E}$; the remaining 80% form the training set $\mathbf{T}$.

For each impulse response in the HRTF, NumPy generates half a second of white noise, which is then convolved with the impulse response to produce the virtual waveform which would be present at each of the listener's eardrums. These waveforms are then divided into windows and a feature vector $\vec{x}_i$ is calculated for each window i, as detailed in Sec. 4.2. This feature vector is then presented to the input layer of an MLP with three output nodes representing a vector $\vec{y}_i^*$ in Cartesian coordinates. The error $\Delta\vec{y}_i^*$, which is driven towards zero by the gradient descent optimization algorithm, is calculated as the difference between this estimate and the unit vector $\hat{y}_i$ which represents the true direction to the sound source (Eq. 1).

$$\Delta\vec{y}_i^* = \vec{y}_i^* - \hat{y}_i \tag{1}$$

The number of input nodes for the MLP is dependent on the hyperparameters of the feature extraction stage, which determine the length of the feature vector $\vec{x}_i$. The MLP has a total of five hidden layers; the first contains 128 nodes, which was chosen to maintain an approximate factor of two reduction from the input layer. Each subsequent hidden layer is a factor of two smaller, and the final hidden layer contains eight nodes which feed the three output nodes $\vec{y}^*$.

The input and output nodes use a linear activation function, and all of the hidden nodes use a common activation function f_{hidden}, which is a hyperparameter of the model; this is one of the hyperbolic tangent, the logistic sigmoid function, or the rectified linear unit (ReLU) (Eq. 2).

$$f_{\text{hidden}}(x) \in \left\{ \tanh(x),\ \frac{1}{1+e^{-x}},\ \max(0,x) \right\} \tag{2}$$

[3] https://www.unidata.ucar.edu/software/netcdf/

[4] https://andresperezlopez.github.io/pysofaconventions/

[5] https://plotly.com/python/

The MLP is trained with the adaptive moment estimation (Adam) [20] gradient descent optimization algorithm with L_2 paramater regularization, which introduces three additional hyperparameters: The L_2 penalty coefficient α and the Adam moment decay rates β_1 and β_2. This processes the entire training set $\mathbf{T}$ each epoch, in mini-batches of 200 data points. The training process terminates once the overall training loss fails to improve in 20 consecutive epochs.

As each window of audio is processed separately, the sequence of model outputs over time can be considered a signal of instantaneous direction estimates. In the case where neither the sound source nor the listener are moving, as here, the mean of this signal can be used as an overall direction estimate. It is also desirable for the model to be consistent, i.e. most directions should produce good estimates. Taking these concerns into consideration, the accuracy score A for the model is defined as the 95th percentile of the mean subtended error angle for all source directions in the testing set $\mathbf{E}$ (Eq. 3). Here, $\vec{y}_i^*$ is the estimated direction for audio window i, n is the set cardinality function, P_{95} is the 95th percentile function, and $\cdot$ is the scalar product.

$$A = P_{95}\left(\left\{\frac{\sum_{i\in\{j:\hat{y}_j=\hat{y}\}}\arccos\left(\frac{\vec{y}_i^*\cdot\hat{y}}{||\vec{y}_i^*||}\right)}{n(\{j:\hat{y}_j=\hat{y}\})} : \hat{y}\in\mathbf{E}\right\}\right) \quad (3)$$

4.2 Feature Vector Design

The feature vector $\vec{x}_i$ for a single window i of length N is calculated from the sampled waveforms that arrive at the listener's left and right eardrums, $\vec{l}_i \in \mathbb{R}^N$ and $\vec{r}_i \in \mathbb{R}^N$. It is composed of three parts, the broadband ILD $x_{\text{BB,i}}$, spectral ILD features $\vec{X}_i$, and spectral phase differences $\vec{\phi}_i$ (Eq. 4). This last component effectively encodes the ITD due to the relationship of phase in the Fourier domain to a shift in the time domain.

$$\vec{x}_i = \left[\ x_{\text{BB},i},\ \vec{X}_i,\ \text{Re}\left(\vec{\phi}_i\right),\ \text{Im}\left(\vec{\phi}_i\right)\ \right] \quad (4)$$

The broadband ILD $x_{\text{BB},i}$ is calculated as twice the difference between the left and right channels' log root-mean-square (RMS) value (Eq.5), which is directly proportional to the difference in power level, expressed in decibels.

$$x_{\text{BB}} = \log(\vec{l}_i\cdot\vec{l}_i) - \log(\vec{r}_i\cdot\vec{r}_i) \quad (5)$$

To calculate the spectral features, the DFT of the left and right waveforms are calculated, $\vec{L}_i \in \mathbb{C}^N$ and $\vec{R}_i \in \mathbb{C}^N$. To ensure that the DFT can be calculated efficiently, the window length N is restricted to be a power of two, and ranges from 1024 to 8192 samples. This corresponds to a duration of 23-186 ms. Humans are capable of performing localization on sounds as brief as 250 ms, so a longer window duration than this would be unsupported [21].

Each of the spectral portions of the feature vector are represented by a number of frequency bins corresponding to the Mel scale. There are several competing definitions of the Mel scale [22], but they all attempt to maintain the property that a constant shift along the scale represents a constant perceived pitch change. This paper uses the definition provided by the Librosa Python package [17], where M_k is a filter bank matrix which maps a DFT onto k perceptually-uniform bins. These bins span the 4-16 kHz range known to be significant to the sound localization task [10]. The size of each of filter bank is drawn from a log-uniform distribution that ranges from 8 to 256 bins. Because the Mel scale is logarithmic, the lower-frequency bins will be calculated from fewer DFT samples than higher-frequency bins; increasing the bin count above 256 can result in these lower-frequency bins having a lower bandwidth than the DFT resolution.

The spectral ILD is calculated from the DFT similarly to the broadband ILD except that the contribution of each frequency component is weighted according to the Mel-filterbank M_n (Eq. 6), where n is a hyperparameter representing the filter bank size. Here, the single vertical bars represent the component-wise absolute value and the superscript 2 represents a component-wise squaring operation. This will result in a different scaling factor than the broadband ILD calculation; this is corrected during the MLP training process.

$$\vec{X}_i = \log(|\vec{L}_i|^2 M_n) - \log(|\vec{R}_i|^2 M_n) \quad (6)$$

To calculate the phase portion of the feature vector $\vec{\phi}_i$, a vector $\vec{\phi}_i^*$ of DFT-domain phase differences is first calculated (Eq. 7); each component of this vector is a unit-length complex number. [6] These are then weighted by the coefficients in the Mel-filterbank M_m, where m is a hyperparameter of the model and re-normalized to unit length (Eq. 8). To avoid problems with discontinuities in an angular representation, the real and complex parts of this vector are included in the feature set separately.

$$\forall k \in [1,N] : \vec{\phi}_i^*[k] = \frac{\vec{L}_i[k]\ \left|\vec{R}_i[k]\right|}{\vec{R}_i[k]\ \left|\vec{L}_i[k]\right|} \quad (7)$$

$$\vec{\phi}_i = \frac{\vec{\phi}_i^* M_m}{|\vec{\phi}_i^* M_m|} \quad (8)$$

4.3 Hyperparameter Search

Several hyperparameters are described in Secs. 4.1 and 4.2, which fall into two broad categories. The selection of features can be adjusted by changing the window size N and by choosing how many bins are used in each of the two filter banks, m and n. The network training can be adjusted by changing the activation function f_{hidden}, the regularization coefficient α, or the Adam parameters β_1 and β_2.

Appropriate values for these hyperparameters are obtained using the Optuna optimization framework [19]. [7] Each trial trains models for 3 different HRTFs with the hyperparameters suggested by Optuna, and submits the most pessimistic result as the overall trial result. The study consists of 361 trials with two optimization targets, the model accuracy A and an estimate of the required training effort. This estimate is

[6] The notation $\vec{v}[k]$ here refers to the k-th component of $\vec{v}$.

[7] The source code and hyperparameter search results are available at `https://2-71828.com/smc22/SMC22_files.zip`

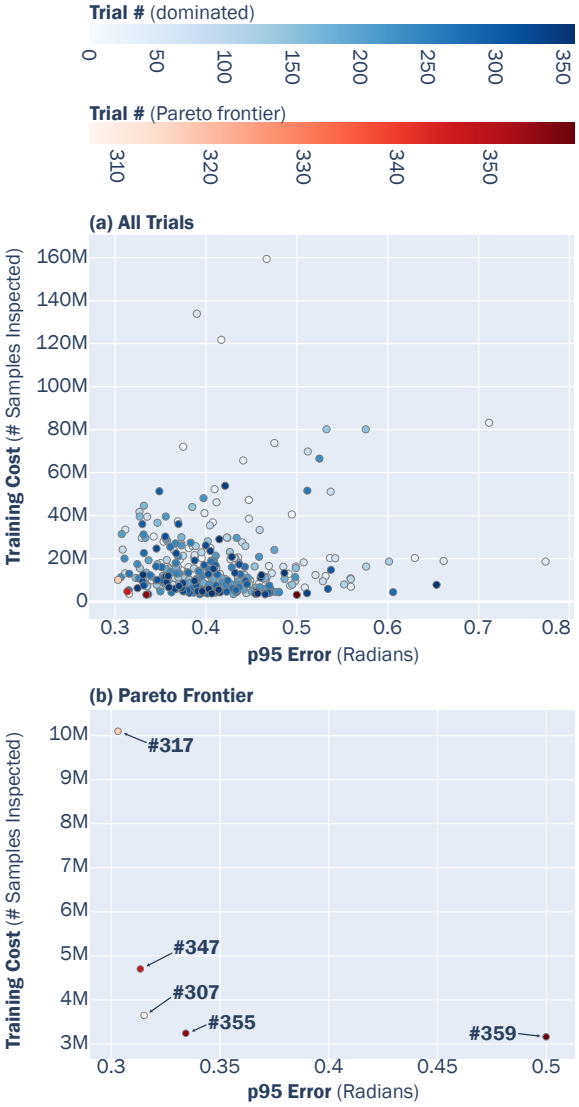

Figure 3. Trial results from Optuna study.

the product of the size of the training set $n(\mathbf{T})$ and the number of epochs processed during training. Notably absent from the estimate is any consideration of the computational cost of each training iteration, such as the differing costs of calculating gradients for the varying activation functions.

Figure 3 summarizes the results of the Optuna study. Plot (a) shows a marker for every trial in the study, and plot (b) show only those trials on the Pareto frontier, i.e. those that are better in some sense than any other trial. On plot (a), the trials on the Pareto frontier [23] are indicated in red. The color saturation indicates the trial order; later trials are shown with a more saturated color. In both plots, the accuracy score A appears on the horizontal axis and the training cost estimate on the vertical. The parameters and optimization values for the five of the trials on the Pareto frontier are reported in Table 1.

The clustering of high saturation points, representing later trials, towards the bottom-left portion of Fig. 3a indicates that the Optuna algorithm has successfully identified some

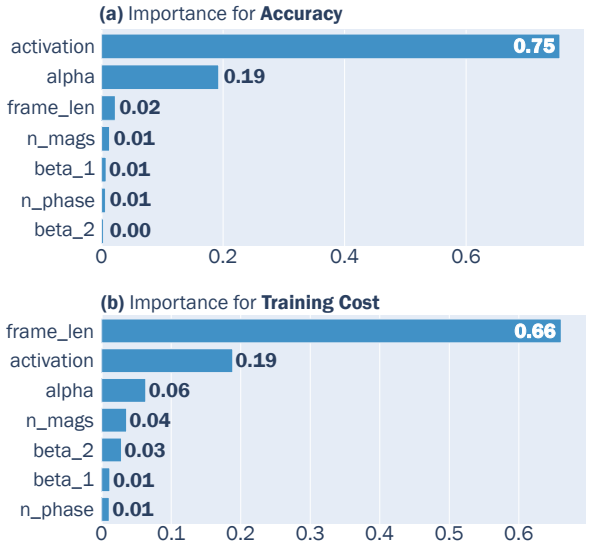

Figure 4. Hyperparameter relative importance.

properties of the hyperparameter space that promote lower training times and higher accuracy. Figure 4 shows Optuna's estimate of the relative importance of the various hyperparameters to each of the optimization targets. Though the order is different, the most important three parameters are the same for both targets: the activation function f_{hidden}, L_2 penalty coefficient α, and window size N.

The accuracy (Fig. 4a) is primarily affected by the choice of activation function f_{hidden} and the L_2 regularization coefficient α. Of the three options evaluated for the activation function, only the hyperbolic tangent appears on the Pareto frontier; this is likely due to the trigonometric character of the underlying problem. The search space for the L_2 coefficient α covered the range $[10^{-5}, 10^{-1}]$ but $\alpha \leq 5.57\times10^{-4}$ on the Pareto frontier, indicating that the upper portion of the search range is unfruitful in this application.

In contrast to the accuracy score, the most important factor in training cost is the audio window size N. Four of the five trials on the Pareto frontier feature the maximum window size of 8192 samples. This maximizes the frequency resolution of the DFT, but also minimizes the number of windows produced for each audio sample. This, in turn, minimizes the size of the MLP training set $\mathbf{T}$. There is a strong negative correlation (-0.62) between $\log_2 N$ and the training cost, and a weak positive correlation (0.16) between $\log_2 N$ and the model's error: Reducing the window size N has the potential of slightly improving accuracy at a large cost in training effort, as can be seen in the results of trial #317.

5. RESULTS AND ANALYSIS

Optuna trial #347 was selected as having the best tradeoff between accuracy and training cost. The parameters from this trial were used to train models for all 48 HRTF records in the ITF database; the results are plotted in Figure 5. With the exception of MRT02, all of the HRTFs form a dense cluster. The accuracy scores for this cluster, which represent the 95th percentile error for each individual model, have

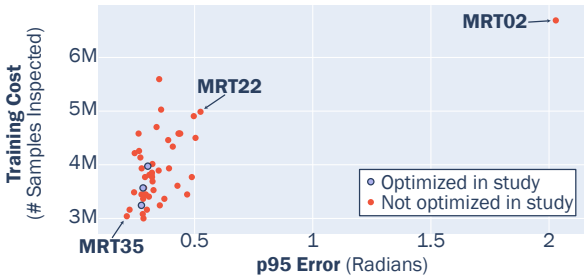

Figure 5. Performance of trained models.

a mean of 19.3° and a standard deviation of 4.45°. As the models used for the parameter search lie in the center of this cluster, there is no evidence that the search over-optimized for those models specifically.

Three of these models were selected for further analysis: the most accurate (MRT35), the least accurate within the main cluster (MRT22), and the outlier (MRT02). Their individual results are plotted in figure 6. The polar plots show the directions that were selected for evaluation and their corresponding errors. The emitter azimuth is plotted on the angular axis, with the listener facing the top of the page. The elevation angle is plotted on the radial axis: The center point represents a vertical alignment and the outer edge represents the horizontal. In each case, the sampling of test vectors is fairly uniform with a bias towards high elevation angles. This bias is inherent in the sampling method used to originally collect the ITA-HRTF data: The same number of azimuth angles were sampled for every elevation angle, so the subtended angle between samples gets denser as the elevation angle increases.

The bottom row of figure 6 shows a kernel-density estimate of the error distribution. In each case, this resembles a log-normal distribution. The maximum likelihood log-normal distribution for each of the 48 trained models predicts the observed quartiles within 14% in all cases. The median estimation error is 3.6% for the lower quartile and 4.3% for the upper quartile.

The color of each marker represents the mean error of sound samples from the indicated direction, where blue points represent a correct prediction and red points indicate a significant error. The two models from the main cluster both exhibit a few randomly-placed outliers and some general regions that have a relatively lower accuracy: MRT35 generally performs well on sounds coming from ahead or above, but performance degrades as the emitter moves to the side. Sounds from very low elevation angles in the left-rear quadrant are particularly troublesome. MRT22 has a reasonable accuracy for sounds that come from above and ahead, but significantly worse performance for all sources below the horizontal.

The outlier, MRT02, still shows good performance for sounds that appear from directly overhead, where the sampling density is highest. Aside from that, the errors appear to be quite uniformly distributed. Where records 22 and 35 sampled elevation angles with a 5° period, the elevation angles in record 2 are sampled with a 10° period. This indicates that the hyperparameter configuration is strongly dependent on the distribution of sample directions in the HRTF.

6. CONCLUSIONS

The hyperparameter search described here takes 3 days to complete on a 6-core Intel i7-10750H processor. This high computational cost severely limits the dimensionality of the hyperparameter search space, rendering larger studies intractable, such as exploring different network topologies. The authors plan to replicate this model in a high-performance computing (HPC) environment to enable such larger studies.

Overall, the developed model performs comparably to humans. When presented with a short-duration sound sample, humans are able to locate a sound source within a p95 confidence interval of (-20.2°, 21.6°) azimuth and (-23.5°, 32.2°) elevation [21]. The average model presented here has a 95^{th} percentile error of 19.3°. Excluding the one outlier, the worst model observed has a 95^{th} percentile error of 30.1°.

The derived hyperparameters show a good resilience to HRTF contents, as long as the distribution of sampled HRTF directions matches the model HRTFs used for the hyperparameter search. Coupled with the fact the features have been derived from characteristics known to be important in the human localization process, this provides strong evidence that this is a good computational model for human localization, suitable for use in evaluating spatial audio systems.

Acknowledgments

This work was performed in the Center of Excellence (CoE) Research on AI- and Simulation-Based Engineering at Exascale (RAISE) receiving funding from EU's Horizon 2020 Research and Innovation Framework Programme H2020-INFRAEDI-2019-1 under grant agreement no. 951733.

The work received support from the NordForsk's Nordic Sound and Music Computing Network (NordicSMC), project number 86892.

Icelandic HPC Competence Center is funded by the EuroCC project that has received funding from the European HPC Joint Undertaking (JU) under grant agreement No 951732. The JU receives support from the EU's Horizon 2020 research and innovation programme.

7. REFERENCES

[1] R. Pelzer, M. Dinakaran *et al.*, "Head-related transfer function recommendation based on perceptual similarities and anthropometric features," *The Journal of the Acoustical Society of America*, vol. 148, no. 6, pp. 3809–3817, 2020. [Online]. Available: https://doi.org/10.1121/10.0002884

[2] M. Dellepiane, N. Pietroni *et al.*, "Reconstructing head models from photographs for individualized 3d-audio processing," *Computer Graphics Forum*, vol. 27, no. 7, pp. 1719–1727, 2008. [Online].

	#307	**#317**	**#347**	**#355**	**#359**
Feature Set Parameters					
Audio samples per window N	8192	2048	8192	8192	8192
Number of spectral ILD bands n	60	177	84	164	242
Number of spectral phase difference bands m	34	60	192	34	34
Training Hyperparameters					
Node activation function f_{hidden}	tanh	tanh	tanh	tanh	tanh
L_2 normalization coefficient α	9.86×10^{-5}	5.57×10^{-4}	1.67×10^{-4}	3.16×10^{-5}	1.04×10^{-4}
Adam coefficient β_1	1.37×10^{-2}	8.88×10^{-2}	6.85×10^{-2}	4.62×10^{-2}	6.20×10^{-2}
Adam coefficient β_2	4.43×10^{-3}	6.86×10^{-3}	8.60×10^{-3}	3.40×10^{-3}	9.70×10^{-3}
Optimization Targets					
95^{th} percentile error (radians)	0.315	0.303	0.313	0.334	0.500
Training cost (# samples inspected)	3.65×10^{-6}	1.01×10^{-7}	4.70×10^{-6}	3.24×10^{-6}	3.16×10^{-6}

Table 1. Hyperparameter and optimization values of the Pareto frontier.

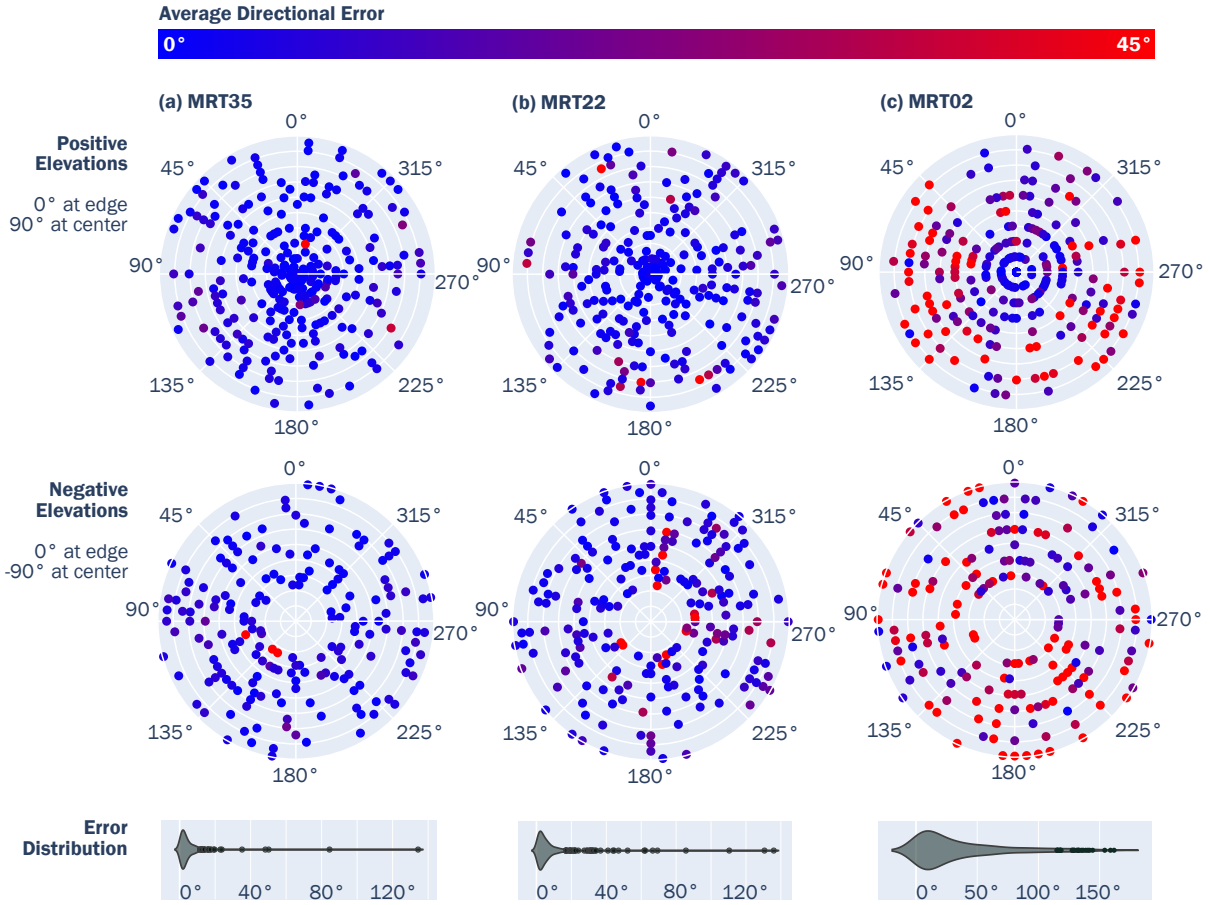

Figure 6. Performance detail of selected models.

Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2008.01316.x

[3] C. Jenny and C. Reuter, "Usability of individualized head-related transfer functions in virtual reality: Empirical study with perceptual attributes in sagittal plane sound localization," *JMIR Serious Games*, vol. 8, no. 3, p. e17576, Sep 2020. [Online]. Available: http://games.jmir.org/2020/3/e17576/

[4] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[5] R. Bomhardt, M. de la Fuente Klein, and J. Fels, "A high-resolution head-related transfer function and three-dimensional ear model database," *Proceedings of Meetings on Acoustics*, vol. 29, no. 1, p. 050002, 2016. [Online]. Available: https://asa.scitation.org/doi/abs/10.1121/2.0000467

[6] P. M. Hofman, J. G. Van Riswick, and A. J. Van Opstal, "Relearning sound localization with new ears," *Nature Neuroscience*, vol. 1, no. 5, pp. 417–421, Sep 1998. [Online]. Available: https://doi.org/10.1038/1633

[7] H. Gray and W. H. Lewis, *Anatomy of the Human Body*, 20th ed. Philadelphia: Lea & Febiger., 1918, [Online]. Available: Bartleby.com, 2000. www.bartleby.com/107/.

[8] H.-J. Kim, D.-G. Jee *et al.*, "The real-time implementation of 3d sound system using dsp," in *IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall. 2004*, vol. 7, 2004, pp. 4798–4800 Vol. 7.

[9] J. Sodnik and S. Tomazic, "Directional information in head related transfer functions," vol. A, 12 2004, pp. 100 – 103 Vol. 2.

[10] J. Hebrank and D. Wright, "Spectral cues used in the localization of sound sources on the median plane," *The Journal of the Acoustical Society of America*, vol. 56, no. 6, pp. 1829–1834, 1974. [Online]. Available: https://doi.org/10.1121/1.1903520

[11] R. Baumgartner, P. Majdak, and B. Laback, "Modeling sound-source localization in sagittal planes for human listeners," *The Journal of the Acoustical Society of America*, vol. 136, no. 2, p. 791—802, August 2014. [Online]. Available: https://europepmc.org/articles/PMC4582445

[12] S. Falkner, A. Klein, and F. Hutter, "BOHB: Robust and efficient hyperparameter optimization at scale," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 1436–1445.

[13] T. Kluyver, B. Ragan-Kelley *et al.*, "Jupyter notebooks - a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Scmidt, Eds. Netherlands: IOS Press, 2016, pp. 87–90. [Online]. Available: https://eprints.soton.ac.uk/403913/

[14] *AES Standard for file exchange - Spatial acoustic data format*, AES69-2020, Audio Engineering Society, Inc., 2020.

[15] C. R. Harris, K. J. Millman *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[16] W. Cochran, J. Cooley *et al.*, "What is the fast fourier transform?" *Proceedings of the IEEE*, vol. 55, no. 10, pp. 1664–1674, 1967.

[17] B. McFee, C. Raffel *et al.*, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8. Citeseer, 2015, pp. 18–25.

[18] F. Pedregosa, G. Varoquaux *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] T. Akiba, S. Sano *et al.*, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19, New York, USA, 2019, p. 2623–2631. [Online]. Available: https://doi.org/10.1145/3292500.3330701

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[21] A. Bronkhorst, "Localization of real and virtual sound sources," *Journal of the Acoustical Society of America*, vol. 98, 11 1995.

[22] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of mfcc," *Journal of Computer science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[23] H. T. Kung, F. Luccio, and F. P. Preparata, "On finding the maxima of a set of vectors," *J. ACM*, vol. 22, no. 4, p. 469–476, oct 1975. [Online]. Available: https://doi.org/10.1145/321906.321910