Pitch Extraction from EEG Data

Rutvik Choudhary * rutvikc2

Sean Farhat-Sabet * seanf2

Jessica Myers * jmmyers3

Abstract

In this paper, we explore pitch extraction from EEG data obtained during music perception. Various methods of preprocessing are investigated, including usage of one-hot and thresholded multi-hot chromagram labels, as well as frequency band extraction and converting the EEG data to images. We study the effectiveness of different methods of classification, including standard and boosted logistic regression and both fully-connected and convolutional neural networks. We found the best method to be convolutional neural networks using the image EEG data, which achieved a test accuracy of 58.63%. Our code is publicly available [3].

1 Introduction

What if you could read someone's mind? Music information retrieval from EEG data is a relatively young field of research. EEG data is known to be noisy due to the brain's simultaneous processing of other activities, such as blinking. Recently, with respect to EEG data analysis in general, deep learning has shown to be promising because it performs automatic feature extraction, reducing the need for domain expertise and de-noising, selecting the most useful features which humans might not have thought of. The focus of this paper is to extract pitch from EEG data obtained while subjects were listening to music. To our knowledge, this is novel because we have not found any prior work attempting this. Progress on machine learning in the EEG domain is important because EEG has the advantage of being cheaper and noninvasive over fMRI.

2 Related Work

While we were unable to find any prior work extracting individual pitches from EEG data, there is a history of work in obtaining other musical information from EEG. A study was done on pitch variation in EEG using principal factor analysis which confirmed that EEG changes with variation in musical pitch [7]. Although it is not clear from this study whether individual notes can be extracted, it provides a basis for our project because it confirms that the EEG data does hold some information relating to pitch. Solidifying this idea further, in a similar study exploring perception of musical styles, the authors found noticeable differences in the extracted ICA features across different style classes but similarities within each class [6]. Attempts have been made to classify songs with EEG, with some researchers finding success with using KNN with alpha, beta, gamma, delta, and theta band features and other researchers finding convolutional neural networks to work better [5] [10]. Tempo extraction and rhythm classification have been explored with moderate success, yet extracting rhythms themselves has been found to be more challenging [13] [12]. Another paper developed a way to reconstruct amplitude fluctuations but found it worked better for speech than for music [15].

*Authors are listed in alphabetical order

3 Training Data

3.1 Dataset

We used the OpenMIIR public domain dataset of EEG recordings designed for music imagery information retrieval [11]. In this dataset, 10 subjects listened to and imagined 12 well-known music fragments, each lasting approximately 7-16 seconds [14]. The dataset was originally designed for classifying what song a subject was perceiving or imagining. The EEG data contains event markers denoting the start times of audio cue clicks, which precede the music fragments. Provided with the dataset is a preprocessing notebook that performs ICA and uses domain-specific techniques to clean the data. The dataset also contains the full audio files containing the cue clicks and music, in addition to separate audio files with solely the cue clicks, some of which were specific to certain subjects.

3.2 EEG Preprocessing

To reduce complexity, we decided to exclude music that contained lyrics. Our goal was to get one or more pitch labels corresponding to each time sample of the EEG data. In order to do this, it was necessary to align the audio files with the EEG data in a 1:1 mapping. The MNE Python library was used to extract event start indexes from the EEG data, which corresponded to the starts of the cue clicks. Then, the audio files were used to compute the start times for the music in the EEG data. The result was a matrix with 69 rows corresponding to 69 EEG channels for each song trial for each subject. We only used data corresponding to music perception. The EEG data was eventually flattened into a 2D matrix and randomly shuffled, along with corresponding labels.

3.3 Chromagram Labels

We obtained a label for each EEG time sample by creating a *chromagram* from the corresponding audio. A chromagram is a representation of audio data where the frequency data is mapped into 12 pitches corresponding to the semitones within a single octave in Western classical music. We chose this representation because we believed that extracting *pitch* from EEG data would bear more fruit than extracting *frequency*. This intuition is guided by our experiences with perceiving music. A Fourier transform representation would differentiate between a wave with *frequency* 440Hz and 880Hz, but a human would most likely recognize that they are the same *pitch*. Thus, it's possible they would correspond to nearby vectors in EEG-space.

The chromagram representation of our data is a single matrix $C \in \mathbb{R}^{N \times 12}$ where N is the number of time samples. Each time sample was normalized to have floating-point entries between 0 and 1. However, in order to use the chromagrams with the logistic regression technique described in Section 4.1, we needed to transform the chromagram matrix C such that all entries are *either* 0 or 1. There were two ways in which we did so. The first method was to create a one-hot version C^{OH} where for each time sample, the dimension with the highest component was set to 1 while the others were set to 0. The second method was to create a multi-hot version C^{MH} by performing a thresholding operation. I.e., $C^{\text{MH}}_{ij} = 1$ if $C_{ij} > T$ and 0 otherwise. T is a hand-selected threshold value. For our experiments we found that $T = 0.75$ was a decent choice. For a visual depiction of the transformation from C to C^{MH}, see the bottom row of Figure 2. A key feature of C^{MH} is its sparsity. On average, $\sim 88\%$ of entries in C^{MH} are 0.

Our usage of a multi-hot version of the chromagram was motivated by the fact that humans listen not only to single pitches, but full chords. For instance, both a C major and C minor chord may have C as the most prominent pitch and so would be indistinguishable from each other when a one-hot transformation was applied. However, even to a "musically naive" listener, these two chords elicit different responses (minor chords are typically referred to as sounding sadder than major chords) [1]. The multi-hot transformation would have more notes as part of the label, giving the two chords different representations.

3.4 Advanced Preprocessing

We also investigated using alternate representations of the EEG data, building upon previous feature engineering methods that have been shown to work in various Brain-Machine tasks.

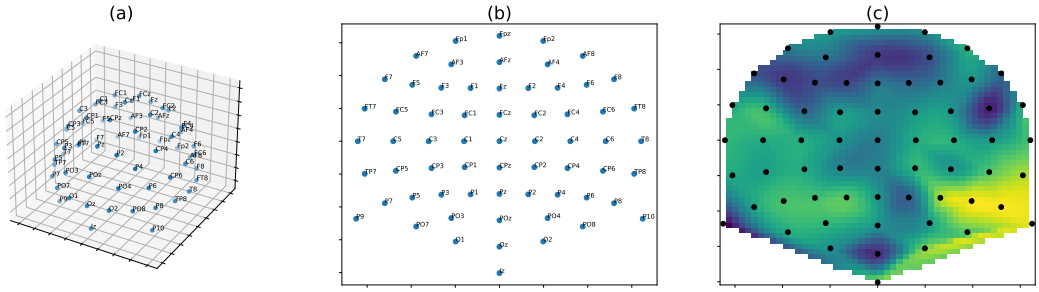

Figure 1: The process of converting each EEG vector to an image. First, the 3 dimensional positions of the EEG electrodes were gathered from the provided .fif files (a). Then, these positions were projected into a 2 dimensional space that preserved the Euclidian distance to the center of the head (the Azimuthal Equidistant Projection) (b). Finally, given the signal values at each electrode, a Bezier-curve based interpolation algorithm (Clough-Tocher) was used to fill in the space in-between (c).

3.4.1 Bands

A well-known result from the neuroscience community is the importance of certain frequency bands of EEG signals [9]. Originally found empirically in sleep studies, these bands are referred to as: delta (1–3 Hz), theta (4–7 Hz), alpha (8–12 Hz), beta (13–30 Hz), and gamma (30–100 Hz). More recently, these bands have been shown to be meaningful features in other tasks as well [2]. For our experiments, we decided to use the delta, theta, alpha, and beta bands. These bands were extracted by passing the raw, 69 channel, signal through appropriate band-pass filters, leaving us with a new $\mathbb{R}^{69\times 4}$ matrix of EEG signals.

3.4.2 EEG Images

Bashivan et. al [2] pointed out that converting EEG signals to a channel-dimensional time series loses spatial information about the electrodes used to gather the signals. They proposed utilizing the electrode locations (provided in standard .fif EEG files) to create an 'EEG Image' at each timestep. Figure 1 shows how this process is done. At each timestep, an EEG image was generated for each band. At the end, the resulting 50x50 images were stacked band-wise in the channel dimension, leaving us with an image of shape (4, 50, 50) for each time-step.

It is worth mentioning that the last 5 channels of the EEG signal did not have any provided locations, so they were omitted, leaving us with 64 channels. In addition, our method differed from the original paper as they generated one image for each group of multiple frames (which varied), whereas we did one per timestep. Lastly, as of writing, it was realized that the same 3D positions were used for all subjects, though we believe that the spatial variance across subjects is probably insignificant.

4 Classification Techniques

4.1 Logistic Regression

For training and testing logistic regression, we used a 90-10 train-test split to partition C^{MH} into $C^{\text{MH,train}}$ and $C^{\text{MH,test}}$. We then trained the logistic regression models using $C^{\text{MH,train}}$. The overall architecture of our system is depicted in Figure 2. We made an assumption that the 12 pitches are independent of each other. Thus, we used 12 separate models $M_1, \ldots, M_{12}$ (one for each feature of $C^{\text{MH,train}}$) and trained each M_i independently with the full EEG training data and column i of $C^{\text{MH,train}}$. The library we used to train our models used the LBFGS optimization algorithm [8]. We configured the optimizer to use class balancing to combat the over-abundance of 0's in the labels.

For evaluation, each M_i performs logistic regression on the full EEG test data to generate column i of a predicted chromagram $\widetilde{C}^{\text{MH}}$. With the output of all 12 models concatenated, we obtain the full predicted chromagram which has the same shape as $C^{\text{MH,test}}$. To evaluate our trained model, we simply compare $C^{\text{MH,test}}$ and $\widetilde{C}^{\text{MH}}$ on a dimension-by-dimension basis.

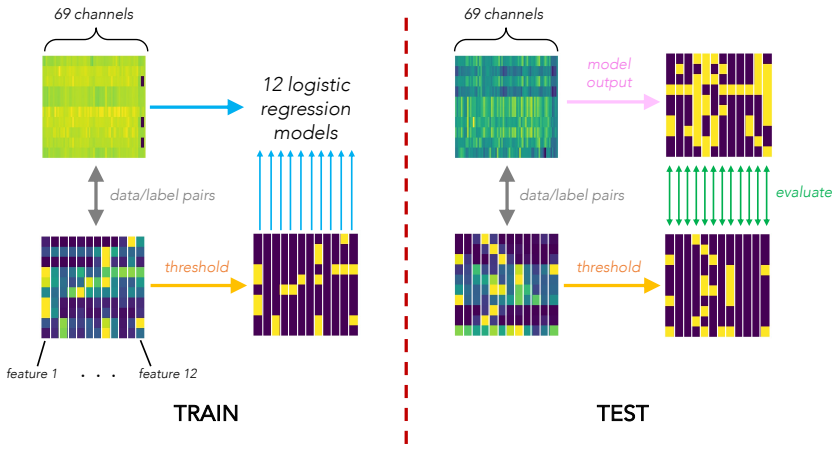

Figure 2: Our architecture for attempting to learn pitch from a multi-hot chromagram consists of 12 logistic regression models. In the training phase, each model is trained with the full EEG training data and the corresponding feature of the thresholded training chromagram. In the testing phase, the EEG test data is passed through each model and their outputs are concatenated to form a prediction chromagram matrix. It is then compared with the thresholded test chromagram.

4.2 Boosted Logistic Regression

As will be seen in Section 5, the multi-hot logistic regression scheme suffers from false positives. This is due to prevalence of 0's in C^{MH} in mentioned in Section 3.3. Our attempt to circumvent this issue was to use the AdaBoost meta-esimator [4] with 50 balanced logistic regression sub-estimators. The architecture is similar to that depicted in Figure 2, only each of the twelve logistic regression models is replaced with an AdaBoost meta-estimator.

4.3 Neural Networks

4.3.1 Dataset

The overall dataset has around 1.8 million samples. Given our resource constraints, we decided to take a subset of the dataset, namely the first 100,000 samples. These samples were shuffled and then separated into a random 80-20 train-test split. Given that the data was in order of subjects over time, the subset we grabbed effectively only covered the first couple subjects, which could lead to poor generalization issues. This is discussed more in the results section.

4.3.2 Vanilla Neural Network

Since neural networks can learn more complex representations, we decided to utilize the band representation of the raw signals as described in Section 3.4.1. Specifically, we stacked the delta, theta, alpha, and beta bands into an $\mathbb{R}^{69\times4\times N}$ matrix (N timesteps) and trained our network on this input. For the outputs, we utilized the one-hot encodings as the ground truth "classifications".

We trained a vanilla neural network made up of Linear and ReLU layers, ending with a LogSoftmax for the 12 chromagram classes. As this was a multi-class classification task, we used Negative Log-Likelihood Loss (essentially the same as Cross Entropy Loss). For the hyperparameters, we used the ADAM optimizer with learning rate 0.001 and a batch size of 32.

Our architecture was as follows:

Vanilla Neural Network	
Layer 1	Linear(64*4, 512) + ReLU
Layer 2	Linear (512, 128) + ReLU
Layer 3	Linear(128, 12) + ReLU
Final Layer	LogSoftmax

4.3.3 Aside: Regressing the Chromagram

It is worth mentioning that we tried an alternate, albeit more interesting, learning problem: if we interpret the chromagram as a probability distribution over the tones, then we could train our vanilla neural network to regress the distribution. We attempted this, training the network to minimize the Kullback–Leibler divergence of its output with the un-thresholded chromagram vector. However, after 10 epochs the loss was not decreasing, so this approach was set aside in favor of the one-hot encoding, where we achieved promising results. We believe that this poor performance is probably due to the simple nature of the architecture, in addition to the low number of training epochs.

4.3.4 Convolutional Neural Networks

With the EEG Images mentioned in Section 3.4.2, it naturally made sense to try a convolutional model, since these are suited to image tasks. Our outputs were the same one-hot vectors used for the vanilla network. The loss, optimizer, and hyperparameters were the same as the vanilla case. Our architecture was composed of standard Convolution layers with 3x3 kernels and no other special changes (dilation, stride, etc.). Between some of the layers were MaxPooling in 2D which were computed with a 2x2 window and a stride of 2. The full details are below:

Convolutional Neural Network	
Layer 1	Conv(4, 32, (3,3)) + ReLU
Layer 2	Conv(32, 32, 3) + ReLU + MaxPool
Layer 3	Conv(32, 64, 3) + ReLU
Layer 4	Conv(64, 64, 3) + ReLU + MaxPool
Layer 5	Conv(64, 128, 3) + ReLU + MaxPool
Layer 6	Linear(128 * 3 * 3, 512) + ReLU
Final Layer	LogSoftmax

5 Results and Discussion

5.1 Guessing Smartly

For the one-hot chromagram training data, the class that occurs most often is class 8 with a frequency of 22.3%. Thus, a model that smartly guesses class 8 every single time would achieve an accuracy of 22.3%. For the multi-hot chromagram training data, $\sim 87\%$ of the matrix's entries are 0's. Thus, by smartly predicting a matrix of all 0's, a model could achieve 87% accuracy.

5.2 Multi-Hot Logistic Regression

The idea of learning with C^{MH} instead of C^{OH} was to provide more information to the classifier. The process of training was described in Section 4.1 and the results are shown in Figures 3 and 4. Looking at Figure 3, we see that while we appear to achieve decent test accuracy scores, we perform significantly worse that the smart guessing strategy. Furthermore, our precision scores are quite low. Looking at the confusion matrices in Figure 4 shows that this is due to the presence of false positives. Our accuracy scores remain high though since our true-negative prediction rate is the highest, and as mentioned before, 0-entries dominate C^{MH}. Our recall scores are high compared to our precision scores because our models predict 1 too often, and so it's natural that the set of true 1's and the set of predicted 1's will have an intersection of non-trivial size.

5.3 Boosted Logistic Regression

Analyzing our results, we see that employing AdaBoost simply helped push the logistic regression models to guess smartly. Looking at Figure 5, we can see that the accuracy scores have jumped up significantly but the precision and recall scores have plummeted. All the models with 0 precision and recall are models that never predicted 1. Thus, the accuracy of model M_i is simply equal to the frequency of 0's in column $C^{\text{MH,test}}_{:,\,i}$ for all $i \notin \{4, 6, 9, 12\}$.

Comparing the confusion matrices in Figures 4 and 6, we see that for the most part, the issue with false positives was corrected by turning them into true negatives.

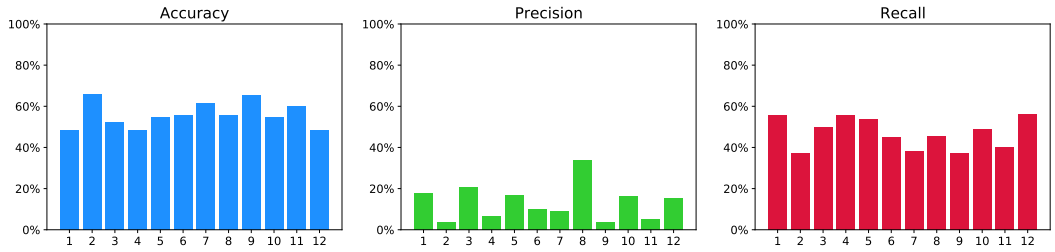

Figure 3: Results for each model, trained on multi-hot chromagram data

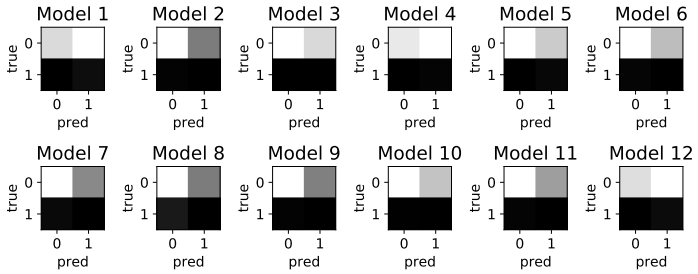

Figure 4: Confusion matrices for each model. Note that true negatives and false positives dominate.

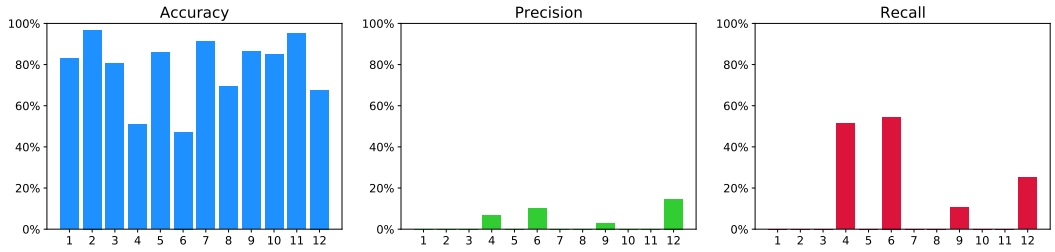

Figure 5: Results for each AdaBoost ensemble, trained on multi-hot chromagram data

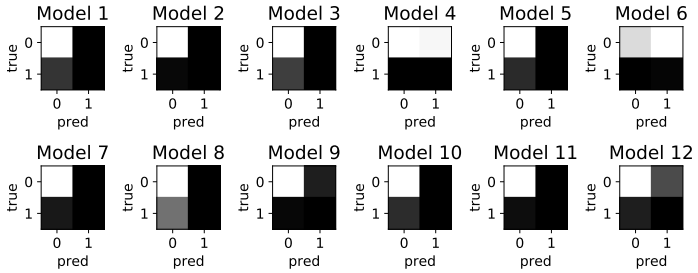

Figure 6: Confusion matrices for each ensemble. Observe that false positives have been mostly eliminated and now only true negatives remain.

5.4 Neural Networks

We trained the Vanilla Neural network for 10 epochs and achieved a final training accuracy of 38.31% with a final test accuracy of 38.21%. Given time constraints, these results seemed promising enough to pursue the specialized convolutional model and devote more resources to that method.

For the Convolutional Neural Network, we were able to get a steadily increasing accuracy over 100 epochs, culminating in a training accuracy of 62.31% and test accuracy of 58.63%. As the test accuracy never steadily decreased, we believe that the model did not overfit the training set, so it may

be of interest to train it for longer to possibly get better performance. However, given the size of the dataset, 100 epochs took around 3 days to train, so we were constrained by resources to explore longer training times.

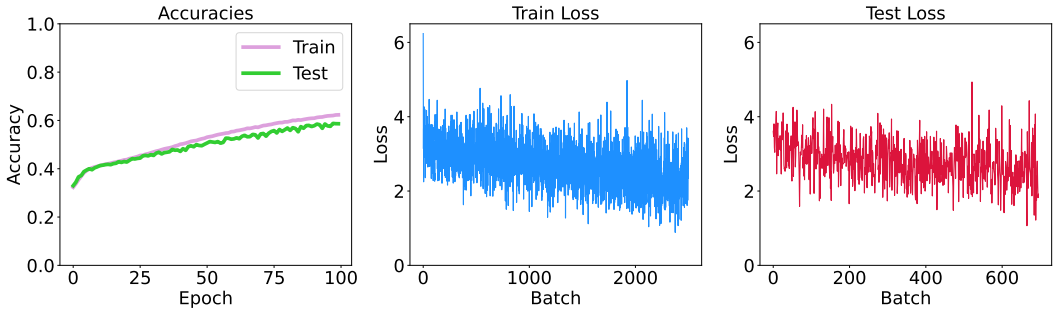

Figure 7: Results of Convolutional Model

It is certainly worth noting that the loss decreased at a much smaller scale than the accuracy, and even then was very noisy. We believe that this indicates a poor generalization to out-of-distribution data. This is because, by choosing the first 100,000 samples (as mentioned in Section 4.3.1), which were in order of subjects over time. we effectively took a subset of the subjects. Therefore, assuming a degree of randomness in the shuffle, the test set is drawn from the same 'subject distributions' as the training data.

6 Conclusion and Future Work

We found the best classification method for identifying pitches in EEG data to be convolutional neural networks, achieving a test accuracy of 58.63%, beating the baseline of 22.3%.

In the future, it would be worthwhile to reduce dimensionality through PCA before performing ICA. Because the dataset already came with ICA performed, we were unable to use PCA. Fewer dimensions might have made it possible to use classification methods more efficiently and try methods that were not possible with the full dataset. Some methods, such as support vector machines, did not finish running or ran out of resources due to the size of the data and number of samples. With hardware limitations removed, feature selection might have been an interesting way to reduce dimensionality as well as gain insights into the EEG data. Also with more processing power, more hyperparameter optimization might have been possible, both in neural networks and in finding an optimal threshold for the multi-hot chromagram labels.

Given the success of the convolutional model on the one-hot task, it would be interesting to see if a) any architecture optimizations could be made to improve the performance on this task, and b) if it could produce good results on the regression task that wasn't pursued past the vanilla model. In addition, our convolutional model's generalizability should be investigated. Specifically, how our model performs on the subjects that were not included in the training data would be quite informative as EEG data varies much more across subjects than within subjects.

References

[1] BAKKER, D. R. ; MARTIN, F. H.: Musical chords and emotion: major and minor triads are processed for emotion. In: *Cogn Affect Behav Neurosci* 15 (2015), Mar, Nr. 1, S. 15–31

[2] BASHIVAN, Pouya ; RISH, Irina ; YEASIN, Mohammed ; CODELLA, Noel: *Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks*. 2016

[3] CHOUDHARY, Rutvik ; FARHAT-SABET, Sean ; MYERS, Jessica: *Source Code for Pitch Extraction from EEG Data*. – URL `https://drive.google.com/file/d/1p99zEXK2P4AoLDplL3yumq6X-bGxCcfQ/view?usp=sharing`

[4] FREUND, Yoav ; SCHAPIRE, Robert E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. In: *Journal of Computer and System Sciences* 55

(1997), Nr. 1, S. 119–139. – URL https://www.sciencedirect.com/science/article/pii/S002200009791504X. – ISSN 0022-0000

[5] LAWHATRE, Prashant ; SHIRAGUPPI, Bharatesh R. ; SHARMA, Esha ; MIYAPURAM, Krishna P. ; LOMAS, Derek: *Classifying Songs with EEG*. 2020

[6] LIN, W. C. ; CHIU, H. W. ; HSU, C. Y.: Discovering EEG Signals Response to Musical Signal Stimuli by Time-frequency analysis and Independent Component Analysis. In: *Conf Proc IEEE Eng Med Biol Soc* 2005 (2005), S. 2765–2768

[7] M., Sreedevi ; A., Ajesh ; R., Ajithnath ; L. S., Binu: A Study of Effect of Music Pitch Variation in EEG Using Factor Analysis and Neural Networks. In: *2009 2nd International Conference on Biomedical Engineering and Informatics*, 2009, S. 1–3

[8] NOCEDAL, Jorge: Updating quasi-Newton matrices with limited storage. In: *Mathematics of Computation* 35 (1980), Nr. 151, S. 773–782. – URL https://doi.org/10.1090/s0025-5718-1980-0572855-7

[9] SABY, J. N. ; MARSHALL, P. J.: The utility of EEG band power analysis in the study of infancy and early childhood. In: *Dev Neuropsychol* 37 (2012), Nr. 3, S. 253–273

[10] SONAWANE, Dhananjay ; MIYAPURAM, Krishna P. ; RS, Bharatesh ; LOMAS, Derek J.: GuessTheMusic: Song Identification from Electroencephalography Response. In: *8th ACM IKDD CODS and 26th COMAD*. New York, NY, USA : Association for Computing Machinery, 2021 (CODS COMAD 2021), S. 154–162. – URL https://doi.org/10.1145/3430984.3431023. – ISBN 9781450388177

[11] STOBER, Sebastian: *openmiir*. https://github.com/sstober/openmiir. 2017

[12] STOBER, Sebastian ; CAMERON, Daniel J. ; GRAHN, Jessica A.: Classifying EEG Recordings of Rhythm Perception. In: *ISMIR*, 2014

[13] STOBER, Sebastian ; PRÄTZLICH, Thomas ; MÜLLER, Meinard: Brain Beats: Tempo Extraction from EEG Data. In: *ISMIR*, 2016

[14] STOBER, Sebastian ; STERNIN, Avital ; OWEN, Adrian M. ; GRAHN, Jessica A.: Towards Music Imagery Information Retrieval: Introducing the OpenMIIR Dataset of EEG Recordings from Music Perception and Imagination. In: *ISMIR*, 2015

[15] ZUK, Nathaniel J. ; MURPHY, Jeremy W. ; REILLY, Richard B. ; LALOR, Edmund C.: Envelope reconstruction of speech and music highlights stronger tracking of speech at low frequencies. In: *PLOS Computational Biology* 17 (2021), 09, Nr. 9, S. 1–32. – URL https://doi.org/10.1371/journal.pcbi.1009358