



HAL
open science

Applying advanced machine learning models to classify electro-physiological activity of human brain for use in biometric identification

Iaroslav Omelianenko

► **To cite this version:**

Iaroslav Omelianenko. Applying advanced machine learning models to classify electro-physiological activity of human brain for use in biometric identification. [Research Report] NewGround LLC. 2017. hal-01571828

HAL Id: hal-01571828

<https://hal.science/hal-01571828v1>

Submitted on 3 Aug 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons CC0 - Public Domain Dedication 4.0 International License

Applying advanced machine learning models to classify electro-physiological activity of human brain for use in biometric identification

Iaroslav Omelianenko

NewGround LLC

Kiev, Ukraine

Email: yaric@newground.com.ua

Abstract—In this article we present the results of our research related to the study of correlations between specific visual stimulation and the elicited brain’s electro-physiological response collected by EEG sensors from a group of participants. We will look at how the various characteristics of visual stimulation affect the measured electro-physiological response of the brain and describe the optimal parameters found that elicit a steady-state visually evoked potential (SSVEP) in certain parts of the cerebral cortex where it can be reliably perceived by the electrode of the EEG device. After that, we continue with a description of the advanced machine learning pipeline model that can perform confident classification of the collected EEG data in order to (a) reliably distinguish signal from noise (about 85% validation score) and (b) reliably distinguish between EEG records collected from different human participants (about 80% validation score). Finally, we demonstrate that the proposed method works reliably even with an inexpensive (less than \$100) consumer-grade EEG sensing device and with participants who do not have previous experience with EEG technology (EEG illiterate). All this in combination opens up broad prospects for the development of new types of consumer devices, [e.g.] based on virtual reality helmets or augmented reality glasses where EEG sensor can be easily integrated.

The proposed method can be used to improve an online user experience by providing [e.g.] password-less user identification for VR / AR applications. It can also find a more advanced application in intensive care units where collected EEG data can be used to classify the level of conscious awareness of patients during anesthesia or to automatically detect hardware failures by classifying the input signal as noise.

Keywords—*biometric identification; electro-physiological brain activity; machine learning; auto-encoder; visually evoked potential; SSVEP; EEG*

I. INTRODUCTION

A steady increase in the computing power of mobile devices allows them to be fully integrated with a variety of portable or wearable consumer devices such as virtual reality (VR) helmets, augmented reality (AR) glasses, monitors of the health and fitness, and so on. At the same time, they have enough computational resources to execute advanced machine learning models for processing of biometric data received from various sensors in the real time. One of these non-intrusive sensors that can be used to collect biometric information is the sensor to monitor the electro-physiological activity of the brain - electroencephalography (EEG). To collect EEG data it’s enough to place electrodes along the scalp of the

user. Moreover, the EEG sensors can be easily integrated into VR helmets or AR glasses, also providing easy means for specific visual stimulation. As has been demonstrated in recent studies [1], [2], exposing a person to external stimuli, such as aural tone or a specific visualization, can cause a specific response in the electro-physiological activity of the brain. And this stationary cerebral cortex response, called steady-state visually evoked potentials (SSVEP) [2], demonstrates a strong correlation with the parameters of visual stimuli, such as flicker frequency and visual appearance (configuration).

In [1], it was found that a reliable SSVEP response can be detected in the Delta and Alpha (Low and High) frequency bands of the EEG signal collected at stimulation with a frequency of visual stimuli in the range: 4 – 10Hz. That is, visual stimuli with a certain frequency and configuration can elicit a reliable SSVEP response in certain parts of the cerebral cortex, which leads to a resonance effect that can be easily detected by EEG sensors placed in specific positions on the scalp of the participant’s head. (This can be partially explained if we consider neural networks of the cerebral cortex as a structural dynamic network of oscillators [3] which can be entrained by specific stimulus; where different regions of cerebral cortex having specific resonance frequencies amplifying its response on stimulation.) Later, the amplified response detected by EEG sensors can be processed using advanced machine learning classification methods to extract useful patterns.

In this work, we use a consumer-grade EEG sensing device with a single electrode located above the FP1 position, which collects the electrical activity of the frontal and pre-frontal cerebral cortex. Because of this limitation, we are most interested in finding specific visual stimuli capable of eliciting a reliable SVEP response in these areas. From the work of other researchers [1], we know that the frontal cortex has a strong response in the High Alpha band, so we focused our efforts on creating a main visual stimulus with a frequency in the range of 8 – 10Hz and auxiliary visual stimulus with the frequency of one harmonic down. In [1] it was found that with simultaneous use of several visual stimuli with frequencies related on the harmonic scale, the SSVEP response of certain parts of neural networks of the cerebral cortex can be amplified.

The purpose of this study is to create an advanced machine learning classification model suitable for processing of collected EEG records and capable of revealing the differences:

- between *signal* and *noise*,

- between EEG records collected from different human participants.

This paper is organized as follows: In Section II, we describe the raw input signals received from the EEG device, visual stimulation details, and basic information about the involved participants. It is followed in Section III by details about collected data records and how it was pre-processed before the analysis. In Section IV, we provide description of machine learning classification model applied to the data corpus. Then, the Section V follows, describing the methodology used to find the optimal configuration of visual stimulation and to find optimal hyper-parameters of the machine learning classification model. After that in Section VI we consider achieved results (validation scores) for both signal / noise and participants identification tasks. Finally, in Section VII and VIII we describe our plans for future experiments and review the results obtained during the research.

II. EXPERIMENT SETUP DESCRIPTION

A. EEG monitoring device and raw input signals

In our experiments, we use NeuroSky MindWave device that monitors the brain's electro-physiological activity using one electrode at the *FPI* position and other at the ear lobe. It has a sampling rate of 512 Hz and decomposes the raw input signal on several frequency bands at the hardware level using a fast Fourier transform (FFT) [4]. After FFT decomposition, the MindWave device broadcasts the decomposed EEG signal once per second (i.e., the refresh rate is 1Hz). See Table I

TABLE I: The ranges of frequency bands for FFT decomposed EEG signal

Band name	Frequency range, Hz
Delta, Δ	1-3
Theta, Θ	4-7
Low Alpha, $L\alpha$	8-9
High Alpha, $H\alpha$	10-12
Low Beta, $L\beta$	13-17
High Beta, $H\beta$	18-30
Low Gamma, $L\gamma$	31-40
High Gamma, $H\gamma$	41-50

Additionally, it provides two *complex synthetic signals* calculated by combining data from several frequency bands, namely:

- *Attention* to indicate users attention level in range [1, 100]
- *Meditation* to indicate users contemplation level in range [1, 100]

We use the mentioned *complex synthetic signals* as control signals that trigger the recording / processing of EEG data when a certain threshold value is reached.

B. Visual stimulation and data collection

Following research [1], we created an advanced software library to produce a certain visual stimulation. The visualization library provides tools for configuring various visualization

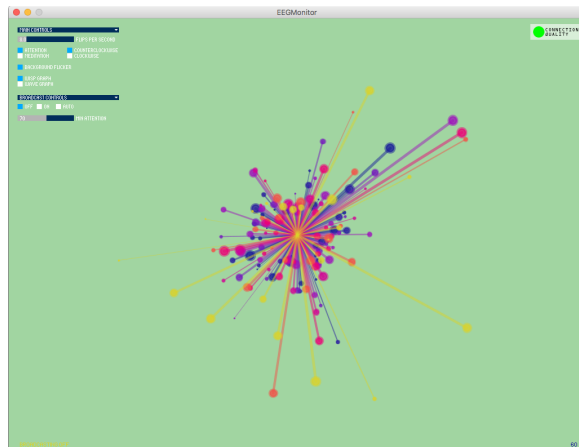


Fig. 1: The screen shoot of visual stimulator window in *WISP* rendering mode.

options and applying various visual modeling schemes: *WISP* and *WAVE*. This allows a series of experiments with different parameters to be performed to find the optimal parameters for eliciting proper SSVEP response.

The visual stimuli created by the visualization library have a strong correlation with the collected EEG data in real time, providing positive feedback to the user on the activity of the cerebral cortex. In the *WISP* rendering mode, this correlation is represented as the relative position of points in polar coordinates and as the color of points and wisplines (see Figure 1). The position of the points relative to the center, determined by the strength of the EEG signal for a particular frequency band. Each frequency band has a certain color assigned to it. To elicit appropriate SSVEP response the visualization has two flickering areas:

- 1) wisplines with dots - flickers with selected primary frequency
- 2) background - flickers with secondary frequency (one harmonic down)

The background color saturation is determined by the current value of selected control signal - *Attention* or *Meditation*, where higher values result in a more saturated color. Thus, the saturation of the background color provides positive feedback for the participant about current value of the control signal.

The central idea of the described scheme of visual stimulation is to achieve aesthetically pleasing and attractive visualization, which can be used in consumer devices: helmets VR, AR glasses, etc. But at the same time, it should provide appropriate visual stimulation to elicit proper SSVEP response from cerebral cortex.

Another studied mode of visual stimulation, we called *WAVE*. Its main difference from *WISP* is that it displays only dots, skipping lines that connect them to the center point (see Figure 2). We found that different visual modes have a variable effect on elicited SSVEP response in different frequency ranges of stimuli. In the *WISP* rendering mode the most effective primary stimulation frequency is 10Hz (High Alpha) and secondary - 5Hz. And for the *WAVE* this is: primary - 8Hz (Low Alpha) and secondary - 4Hz.

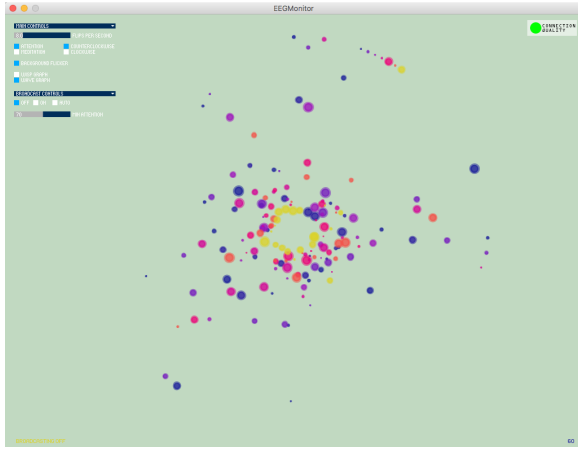


Fig. 2: The screen shoot of visual stimulator window in *WAVE* rendering mode.

The graphic interface of the created visualization library also allows you to set a certain control signal threshold that starts / stops recording the received EEG signals. We learned that with higher threshold values, it is much more difficult for a participant to maintain that level of control signal (*Attention / Meditation*), but the collected data is of better quality. Thus, it is important to find the right balance between the quality of the collected data and ease of use for the participants. In our experiments, we applied four threshold values from 80 down to 50 in steps of ten. The optimal threshold value for the *WISP* rendering mode is 70 providing a good balance between the quality of the collected data and ease of use for the participant.

We measure quality of the collected data and the applicability of a specific configuration of visual stimuli, by checking Pearson product-moment correlation [5] between the processed data from the recorded sessions for each participant per each configuration. (For more information on calculating correlation coefficients, see Section: "Finding Optimal Configuration for Visual Stimulation".) And for further analysis, the configuration of visual stimuli creating the most correlated results is considered. Our hypothesis for this is that the maximum correlation between recording sessions is the result of properly elicit SSVEP response from a specific parts of the cerebral cortex that has a steadily recognizable pattern among all recorded sessions.

Applying the above-mentioned method to estimate quality of the collected EEG data, following visualization parameters were chosen for all participants as optimal:

- *WISP* rendering mode
- 10Hz primary and 5Hz secondary flicker frequencies
- *Attention* complex synthetic signal as a control signal
- 70 as the control signal threshold value

In a series of experiments, we found that reliable data processing requires at least *forty* collected data samples (EEG events) per session. The MindWave device broadcasts data event signals (with 8 frequency bands values) once per second. This means that one recording session should take at least 40

seconds, if participant can maintain a continuous high-level value of the control signal. But this can take much longer if the value of the control signal drops below the threshold and the recording pauses until participant can again reach the threshold value.

C. Experiment Participants

In our experiments we collected EEG data records from three participants:

- 1) male, 43 years, -1 myopia, astigmatism
- 2) female 28 years, normal sight
- 3) male 35 years, -7 myopia

For each participant, we collected from six to eighteen sessions of EEG records with various schemes of visual stimulation and at different times. It is very important that EEG sessions be recorded at different times in order to observe the electro-physiological response of the participants' brain in different conditions.

III. DATA CORPUS DESCRIPTION

The data collected from the EEG device during a particular recording session is stored as comma-separated records with 40 rows, 11 columns wide. Each line contains three meta-data variables (time stamp of record, frequency of primary and secondary stimuli) and eight values for each frequency band.

The frequency bands data collected from EEG device represent a power spectrum and its values vary exponentially in the range $[0, 32767]$. But performance of a majority of machine learning methods is best with small floating point values centered around zero. Thus, in order to improve the quality of the collected data and to balance the values of the data points for different frequency bands, we scaled it to fit in range $[0, 1]$ during the preprocessing. And the pre-processed data is later used as input for the machine learning classification model.

IV. MACHINE LEARNING CLASSIFICATION PIPELINE MODEL DESCRIPTION

Essentially, the collected data corpus of EEG records is a time series. Therefore, it may seem natural to analyze it using machine learning methods, which are usually considered to have the best architecture for time series data analysis, e.g. Recurrent Neural Networks (RNN) and its modern reincarnation - Long Short Time Memory Neural Network (LSTM) [6]. But for RNN model to learn useful patterns from input data at least two conditions should be met: (a) the data samples should be collected at equal time intervals between events; (b) it is important to collect a significant number of data samples for each recording session. In our experiment, no above condition can be fulfilled. First of all, it is unpractical to collect a lot of data samples for consumer device setup, because people easily get bored during long recording sessions. Furthermore, we collect EEG data events only when the level of the control signal exceeds a certain threshold, which makes our time series rather inconsistent with different intervals between time stamps of collected events.

Taking into account the specific conditions of our experiment mentioned above, we decided to explore a novel approach



Fig. 3: The machine learning processing pipeline scheme from left to right: the EEG input source, the auto-encoder, and the classifier.

to the analysis of time series, which allows us to extract useful patterns (essence) from the analyzed EEG signal records and classify it later. Our idea is to create advanced machine learning classification pipeline, where the input EEG data is processed in two stages (see Figure 3):

- at the first stage, raw input data are analyzed using unsupervised machine learning method based on the auto-encoder neural network with single hidden layer [7]
- at the second stage, the learned weights of the auto-encoder hidden layer (data encoding) are used as input for the classifier of choice to train the supervised classification model

In the following we describe considered methods of machine learning used at both stages, with more details.

A. The First stage - Auto-Encoder Selection

In our experimental setup - per each individual session - we have the data corpus consisting of eight frequency bands values collected for forty data events. That gives us the features space dimension width of 320 for each EEG recording session. At the same time, we have very limited number of recording sessions (6 to 18) for every participant, which is much less than the dimensionality of the features space. Such characteristics of the input data make it unacceptable for analysis by almost any machine learning classification model, since ideally the number of data samples must significantly exceed the number of features and not vice versa. This forced us to look for a specific method of machine learning that can extract its reliable representation (encoding) from the raw EEG input signal and accordingly reduce the dimensionality of the features space.

Our attention was captured by family of unsupervised machine learning methods called auto-encoders [7]. An auto-encoder takes an input $\mathbf{x} \in [0, 1]^d$ and first maps it (with an encoder) to a hidden representation $\mathbf{h} \in [0, 1]^d$ through a deterministic mapping, e.g.:

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (1)$$

where σ is a non-linearity such as the sigmoid.

The latent representation \mathbf{h} , or **encoding** is then mapped back with a decoder into a reconstruction \mathbf{z} of the same shape as \mathbf{x} . The mapping happens through a similar transformation, e.g.:

$$\mathbf{z} = \sigma(\mathbf{W}'\mathbf{h} + \mathbf{b}') \quad (2)$$

\mathbf{z} should be seen as a prediction of \mathbf{x} , given the code \mathbf{h} . Optionally, the weight matrix \mathbf{W}' of the reverse mapping may be constrained to be the transpose of the forward mapping: $\mathbf{W}' = \mathbf{W}^T$. This is referred to as tied weights. The parameters of this model (\mathbf{W} , \mathbf{b} , \mathbf{b}' and, if one doesn't use tied weights, also \mathbf{W}') are optimized such that the average reconstruction error is minimized.

The reconstruction error can be measured in many ways, depending on the appropriate distributional assumptions on the input given the code. The traditional squared error $L(\mathbf{x}, \mathbf{z}) = \|\mathbf{x} - \mathbf{z}\|^2$, can be used as well. If the input is interpreted as either bit vectors or vectors of bit probabilities, the cross-entropy of the reconstruction can be used:

$$L_H(\mathbf{x}, \mathbf{z}) = - \sum_{k=1}^d [\mathbf{x}_k \log \mathbf{z}_k + (1 - \mathbf{x}_k) \log(1 - \mathbf{z}_k)] \quad (3)$$

The \mathbf{h} can be viewed as lossy compression of \mathbf{x} and optimization makes it a good compression for training examples and hopefully for all other examples as well. But this can not be guaranteed for arbitrary inputs. The simple auto-encoder gives low reconstruction error on test samples from the same distribution as train samples, but usually gives higher error values for inputs randomly chosen from input space.

To prevent auto-encoder from learning identity function of input signal and to force it to learn something useful about input signal in its hidden units, several methodologies can be applied: (a) addition of sparsity to inputs, (b) addition of randomness to transformation (encoding) or (c) introducing specific explicit regularizer to the objective function.

In this research we study two types of auto-encoders with various learning enhancements applied:

- Denoising auto-encoder (addition of sparsity to inputs)
- Contractive auto-encoder (the explicit regularizer to the objective function)

The denoising auto-encoder [8] is a stochastic version of simple auto-encoder where stochastic corruption process randomly sets many of inputs (maximum half of them) to zero. Then we train it to reconstruct correct input from a stochastically corrupted version of it. As result, the higher level representations of input data (encoding) are relatively stable and robust to the corruption of input data.

The contractive auto-encoder [9] adds explicit regularizer to the objective function that forces model to obtain a robust representation of the input space not sensitive to slight variations of input data. This regularizer corresponds to the Frobenius norm (L2) of the Jacobian matrix of the hidden representation with respect to the input. The final objective function has the following form:

$$L = - \sum_{k=1}^d [\mathbf{x}_k \log \mathbf{z}_k + (1 - \mathbf{x}_k) \log(1 - \mathbf{z}_k)] + \lambda \sum_{i=1}^d \sum_{j=1}^n \mathbf{J}_{ij}^2 \quad (4)$$

where $\mathbf{z} = \sigma(\mathbf{W}'\mathbf{h} + \mathbf{b}')$ is the reconstruction of input vector, $\mathbf{J}_i = \mathbf{h}_i(1 - \mathbf{h}_i) \cdot \mathbf{W}'_i$ is Jacobian of \mathbf{h} with respect to \mathbf{x} , and $\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$ is the projection of the input into the latent space \mathbf{h} .

Both mentioned auto-encoder variations are robust to the slight corruption of input data. This is especially useful for our experiment, since it is well known that the electro-physiological activity of the brain is non-stationary and varies greatly, depending on the circumstances. I.e. the collected EEG data for each subsequent recording session under the same type of visual stimuli can give the same essence (compressed representation), but can introduce various non-stationary deviations from previous sessions. Using an auto-encoder to analyze the input data at the first stage, we hope to reduce non-stationary deviations and extract a useful compressed representation. Then, at the second stage, the extracted compressed representation with significantly reduced dimensionality of features space will be fed into the classifier of choice.

The above-mentioned auto-encoders was implemented in Python programming language using Theano framework [20].

B. The Second Stage - Classifiers Evaluation

At the end of the first stage of the processing pipeline, the dimensionality of data corpus features space is considerably reduced. After that, at the second stage, the data is fed to a set of classifiers from simple to advanced in order to find the one that has the most powerful prediction model.

To automate the evaluation of classifiers against different sets of hyper-parameters, we use exhaustive grid search with cross-validation. This method exhaustively generates candidates from the grid of parameter values specified for each classifier, and then performs a *3-fold* cross-validation (CV) of the predictive performance of each classifier with respect to the input data [10]. In the *k*-fold cross-validation method, the training set is split into smaller sets and the following procedure is performed for each of the folds:

- 1) a model is trained using $k - 1$ of the folds as training data
- 2) the resulting model is validated on the remaining part of the data (i.e., it is used as a test set to compute a performance measure such as accuracy)

The resulting classifier performance measure is done by averaging scores computed for each fold. Thus, we can avoid the effect of model overfitting, when it shows perfect score for train data but fails to predict anything useful for unseen data. Refer to Table II for a list of all tested classifiers and all tested values of corresponding hyper-parameters per each estimator.

Hereafter we provide short description of each tested classifier.

Random Forest classifier [11] is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control overfitting.

Ada Boost classifier [12] is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Decision Tree classifier [13] is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an

TABLE II: The classifiers used in the study with range of hyper-parameter values applied for exhaustive grid search.

Classifier	Parameters space
Random Forest	max_depth: [3, 5, 8, None], max_features: [1, 3, 4], min_samples_split: [2, 3, 4], min_samples_leaf: [1, 3, 10], bootstrap: [True, False], criterion: ["gini", "entropy"], n_estimators: [10, 20, 50, 100, 200]
Ada Boost	learning_rate: [0.01, 0.1, 1], n_estimators: [10, 20, 50, 100]
Decision Tree	max_depth: [3, 5, 8, None], min_samples_split: [2, 3, 4], min_samples_leaf: [1, 3, 10, 20, 30]
Gaussian Process	warm_start: [True, False]
Multi Layer Perceptron	alpha: [0.0001, 0.001, 0.01], learning_rate_init: [0.001, 0.01, 0.1, 0.5], momentum: [0.9, 0.99, 0.999], solver: ["lbfgs", "sgd", "adam"], activation: ["logistic", "tanh", "relu"], hidden_layer_sizes: [4, 8, 10]
K-neighbors	n_neighbors: [2, 3, 5], algorithm: ["ball_tree", "kd_tree", "brute"]
Gaussian Nave Bayes	priors: [None]
Quadratic Discriminant Analysis	priors: [None], reg_param: [0.0, 0.01, 0.1, 0.9]
Support Vector Machine with RBF kernel	C: [0.1, 0.5, 1.0], gamma: [0.1, 0.5, 1.0, 2.0, 3.0, 'auto']
Support Vector Machine with Linear kernel	C: [0.001, 0.005, 0.01, 0.025, 0.05, 0.1, 0.25, 0.5, 1.0]

attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. This structure allows to go from observations (branches) to the conclusions about the item's target value (leaves).

Gaussian Process classifier [14] based on Laplace approximation, which is used for approximating the non-Gaussian posterior by a Gaussian.

Multi-layer Perceptron (MLP) classifier [15] is an artificial neural network based classifier optimizing log-loss function using LBFGS or stochastic gradient descent.

K-neighbors classifier [16] use a type of instance-based learning or non-generalizing learning where classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point.

Gaussian Naive Bayes [17] is a model assuming that the continuous values associated with each class are distributed according to a Gaussian distribution.

Quadratic Discriminant Analysis [18] is a classifier with a quadratic decision boundary, generated by fitting class conditional densities to the data and using Bayes rule.

Support Vector Machine classifier [19] builds a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on on which side of the gap they fall.

We implemented exhaustive grid search with *3-fold* cross-validation for all above mentioned classifiers in Python programming language using Scikit-learn framework [21].



Fig. 4: The heat map with correlation coefficients between eight recording sessions for one of the experiment participants. It can be seen that majority of sessions highly correlate with each other and only session #6 is a bit outlier. Such high correlation values is a good sign that applied configuration of the visual stimuli is optimal to elicit robust SSVEP response which can be compressed to extract useful pattern (encoding).

V. FINDING OPTIMAL EXPERIMENT CONFIGURATION

The details of the experiment described above suggest that in order to obtain reliable classification, it is important to find optimal configuration parameters for both parts of the experiment: visual stimulation and machine learning classification pipeline.

Hereafter we consider in more details how optimal configuration was found.

A. Finding Optimal Configuration for Visual Stimulation

To achieve the objectives of the experiment, first, we need to find the optimal configuration of visual stimuli that elicit a reliable SSVEP response in the cerebral cortex of participants. The optimal configuration should give comparable results for different EEG sessions recorded on different times. As mentioned in Subsection: "Visual stimulation and data collection" to do this we conducted series of experiments (recording sessions) and evaluate collected data by calculating Pearson product-moment correlation [5] between outputs from the first stage of machine learning pipeline (auto-encoder). Our hypothesis is that the visual stimulation scheme leading to outputs with the maximal correlation can be considered optimal.

The correlation coefficients can be calculated as following:

- 1) The collected EEG raw input data preprocessed by scaling it down to fit range $[0, 1]$ (see Section: "Data Corpus Description").
- 2) After that, the scaled data corpus is fed into auto-encoder of choice. The input data is filtered to include only *Delta* and *High Alpha* bands. These bands were chosen based on the results of [1], which state that the elicited SSVEP response should have bands frequencies close to the frequencies of the visual stimuli

(10/5Hz). It was found that for best results the auto-encoder should have only two units in the hidden layer.

- 3) The learned weight matrix of the auto-encoder's hidden layer (compressed input data representation) per each recording session then collected as representation data matrix with columns holding compressed data for each session.
- 4) Finally, the Pearson product-moment correlation coefficients calculated from representation matrix and heat map plot with correlations are rendered (see Figure 4).

B. Finding Optimal Configuration for Machine Learning Classification Pipeline Model

The optimal configuration of classification pipeline was found in a series of experiments by applying various hyper-parameters to auto-encoders and classifiers. After that, the classification model with the maximum validation score was chosen as optimal. To automate the process of selecting the optimal model, a software framework was created that performs the following procedures:

- 1) starts preprocessor to process the raw input data corpus consisting of collected EEG recording sessions,
- 2) uses auto-encoder against preprocessed input data to extract useful patterns,
- 3) perform classifiers evaluation against extracted patterns as described in Section: "The Second Stage - Classifiers Evaluation".

The experimental framework allows to separately define various configuration parameters for each processing stage. Thus, the configuration of each experiment can be encapsulated in a separate Python script, which makes it easy to reproduce a specific experiment in the future.

We tested a variety of hyper-parameters combinations per each auto-encoder with different sets of filtered input data samples (only values for specific frequency bands are included). As result, we found that learning rate, contraction / corruption level and the number of learning epochs can be fixed by the values presented in Table III. Only the batch size, the number of auto-encoder's hidden units and frequency bands included into input data corpus, can be changed when searching for the best validation score.

VI. RESULTS

In this paper, we considered the possibility of creating reliable machine learning classification pipeline model able to:

- 1) confidently distinguish between signal and noise in collected EEG records
- 2) confidently classify EEG records of different participants, that is, provide means to determine which EEG record belongs to which participant.

Hereafter we consider achieved results in more details.

TABLE III: The optimal hyper-parameters for auto-encoders for signal / noise classification task.

Hyper-parameter	Tested values range	Contractive auto-encoder	Denoising auto-encoder
Batch size	[1, 10]	1 or 10	5
Learning rate	[0.001, 0.1]	0.1	0.1
Number of units in hidden layer	[2, 10]	5	5
Number of learning epochs	[10^4 , $5 \cdot 10^4$]	$5 \cdot 10^4$	$5 \cdot 10^4$
Contraction level (contractive auto-encoder)	[0.1, 0.3]	0.1	-
Corruption level (denoising auto-encoder)	[0.1, 0.3]	-	0.1
Included frequency bands	Δ , Θ , $L\alpha$, $H\alpha$, $L\beta$, $H\beta$, $L\gamma$, $H\gamma$	Δ , Θ , $L\alpha$, $H\alpha$, $L\beta$, $H\beta$	Δ , Θ , $L\alpha$, $H\alpha$, $L\beta$, $H\beta$

A. Signal / Noise Classification

For this task our main goal is to build classifier capable of distinguishing between *signal* and *noise* records collected under different conditions and at different times. The *signal* records include SSVEP affected records plus EEG records obtained from participants performing other cognitive tasks (sky watching, reading, etc). The basic requirement for the proper *signal* record - is to be collected when the value of the control signal (Attention or Meditation) is greater than or equal to the specified threshold value. The *noise* records are collected when the electrodes of the EEG device are improperly installed on the participants (for example, the base electrode is not connected) or when the device is turned on but not connected to the participant at all. At total, we studied data samples collected over 26 *signal* and 26 *noise* sessions.

TABLE IV: The comparison of the best validation scores for signal / noise classification per particular set of included frequency bands and specific values of contractive auto-encoder hyper-parameters. The score value is given as the average of 3-fold cross validation with standard deviation among individual values for each fold.

Frequency bands included	Batch size	# of hidden units	Best classifier	Validation score
Δ , $H\alpha$	5	2	Multi Layer Perceptron	0.736 (std: 0.184)
Δ , $L\alpha$, $H\alpha$	5	4	Multi Layer Perceptron	0.736 (std: 0.050)
Δ , Θ , $L\alpha$, $H\alpha$	5	3	SVM with linear kernel	0.755 (std: 0.034)
Δ , Θ , $L\alpha$, $H\alpha$	1	2	Gaussian Nave Bayes	0.833 (std: 0.068)
Δ , Θ , $L\alpha$, $H\alpha$, $L\beta$	5	4	Multi Layer Perceptron	0.774 (std: 0.098)
Δ , Θ , $L\alpha$, $H\alpha$, $L\beta$, $H\beta$	10	5	Multi Layer Perceptron	0.849 (std: 0.058)
Δ , Θ , $L\alpha$, $H\alpha$, $L\beta$, $H\beta$, $L\gamma$	5	9	Random Forest	0.792 (std: 0.075)

We began experiments with finding a best validation score for a specific set of auto-encoder's hyper-parameters when only two frequency bands were included in the input data corpus and continued until all bands were involved. We found that in all experiments the classification pipeline based on contractive auto-encoder was superior to that based on the denoising auto-encoder. Therefore, we will provide only the best results for the contractive auto-encoder together with the classifier name and the included frequency bands (see Table IV).

The best signal / noise classification score (0.849) obtained for the machine learning classification pipeline, including the contractive auto-encoder, followed by the classifier based on multi-layer perceptron architecture. See Table III, IV for used auto-encoder hyper-parameters. Optimum hyper-parameters of the most effective classifier (MLP) - momentum: 0.999, activation: 'relu', solver: 'sgd', alpha: 0.0001, learning_rate_init: 0.001, hidden_layer_sizes: 8

As a result of the experiments, we found that for the signal / noise classification the SSVEP response elicited by visual stimulation is not important. Only the level of the control signal (Attention or Meditation) during the recording of EEG events is of primary importance. The best result was achieved when almost all frequency bands is included in the analyzed data corpus, which confirms that the elicited by visual simulation SSVEP response is not important for this task.

Thus, we can conclude that the proposed method of signal / noise classification can be used without any visual stimulation. The scope of this method application can be related both to the monitoring of EEG hardware failures and to control the user's level of consciousness in intensive care units.

B. The Classification of Participants

The second goal of our research is to build a model of machine learning classification pipeline that allows correctly labeling EEG records for specific participants, that is, allowing to classify a participant on the basis of her unique fingerprint in the collected EEG data.

In our experiments we decided to apply visual stimulation with main flicker frequency at 10Hz and secondary - at 5Hz, which corresponds to Theta (Θ) and High Alpha ($H\alpha$) bands. In accordance with [1], we assumed that elicited SSVEP response should be detected at these or adjacent frequency bands and that the best validation score will be obtained if only these frequency bands are included in the input data corpus.

A series of experiments was performed on the collected signal-only EEG records with various frequency bands included and the best result (0.800) was achieved with data corpus consisting of signal-only records with Delta (Δ) and High Alpha ($H\alpha$) bands included. It almost perfectly fits into the expected range - the Delta (Δ) band is adjacent to the Theta (Θ) band. This congruence is a good indication that SSVEP stimulation is important when collecting EEG records to be classified by participants. See Table V for details about the best results found for various data corpus and hyper-parameters configurations.

VII. FUTURE WORKS

Our current research based on very limited data corpus collected from only three participants. Because of this limi-

TABLE V: Comparison of the best validation scores for signal-only classification per particular set of included frequency bands and specific values of hyper parameters of contractive auto-encoder. The score value is given as an average of 3-fold cross validation with standard deviation between individual values for each fold.

Frequency bands included	Batch size	# of hidden units	Best classifier	Validation score
$\Theta, H\alpha$	5	3	Multi Layer Perceptron	0.750 (std: 0.165)
$\Delta, H\alpha$	5 or 1	2	Random Forest	0.800 (std: 0.041)
$\Delta, L\alpha, H\alpha$	5	2	Multi Layer Perceptron	0.750 (std: 0.065)
$\Delta, \Theta, L\alpha$	5	2	Multi Layer Perceptron	0.600 (std: 0.267)
$\Delta, \Theta, L\alpha, H\alpha$	5	5	Random Forest	0.650 (std: 0.138)
$\Delta, \Theta, L\alpha, H\alpha, L\beta$	5	4	Random Forest	0.700 (std: 0.135)
$\Delta, \Theta, L\alpha, H\alpha, L\beta, H\beta$	1	5	Random Forest	0.750 (std: 0.102)

tation, we may have missed some important patterns in EEG records, which can further improve the validation scores of studied machine learning classification pipeline methods. In the future, we plan to collect and process EEG records from a much wider audience of participants.

VIII. CONCLUSION

In this paper, we demonstrated how to build machine learning classification pipeline capable of processing and reliably classifying EEG records collected from different participants. We described a visual stimulation scheme suitable to elicit specific SSVEP response from visual cortex and hypothesized how to measure its quality using the collected EEG recordings. In addition, special attention was paid to creating an aesthetically attractive scheme of visual stimulation in the assumption of its use in consumer devices.

We managed to get a fairly good estimate of confidence score with signal / noise classification task (best validation score: 0.849, see Table IV). This confirms that the proposed method is suitable for automatic processing of EEG records, when it is important to distinguish the signal from noise.

We also demonstrate that, using the proposed method, it is possible with sufficient confidence to classify EEG records for each participant (best validation score: 0.800, see Table V). We believe that further improvement of the method can provide sufficiently reliable means for use even for user identification based on the collected EEG data.

REFERENCES

- [1] Jian Ding, George Sperling, Ramesh Srinivasan (2006). *Attentional modulation of SSVEP power depends on the network tagged by the flicker frequency*. Cereb Cortex (2006) 16 (7): 1016-1029. DOI: 10.1093/cercor/bhj044
- [2] Regan D. (1977). *Steady-state evoked potentials*. J Opt Soc Am 67:14751489.
- [3] Evelyn Tang, Danielle S. Bassett (2017). *Control of Dynamics in Brain Networks*. arXiv preprint: arXiv:1701.01531
- [4] Heideman, M. T.; Johnson, D. H.; Burrus, C. S. (1984). *Gauss and the history of the fast Fourier transform*. IEEE ASSP Magazine. 1 (4): 1421. DOI: 10.1109/MASSP.1984.1162257
- [5] Gain, A. K. (1951). *The frequency distribution of the product moment correlation coefficient in random samples of any size draw from non-normal universes*. Biometrika. 38: 219-247, 1951.
- [6] Hochreiter, Sepp; and Schmidhuber, Jrgen (1997) *Long Short-Term Memory*. Neural Computation, 9(8):17351780, 1997
- [7] Bengio, Y. (2009). *Learning Deep Architectures for AI*. Foundations and Trends in Machine Learning. 2. DOI:10.1561/2200000006
- [8] P. Vincent, H. Larochelle Y. Bengio and P.A. Manzagol (2008) *Extracting and Composing Robust Features with Denoising Autoencoders*. Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML08), pages 1096 - 1103, ACM, 2008.
- [9] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio *Contractive Auto-Encoders: Explicit Invariance During Feature Extraction*. Proceedings of the Twenty-eight International Conference on Machine Learning (ICML11)
- [10] Kohavi, Ron (1995). *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. San Mateo, CA: Morgan Kaufmann. 2 (12): 1137-1143.
- [11] L. Breiman (2001) *Random Forests*. Machine Learning, 45(1), 5-32, 2001.
- [12] J. Zhu, H. Zou, S. Rosset, T. Hastie (2009) *Multi-class AdaBoost*. Statistics and Its Interface, 2, 349-360. 2009.
- [13] L. Breiman, J. Friedman, R. Olshen, and C. Stone (1984) *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.
- [14] C. E. Rasmussen, C. K. I. Williams (2006) *Gaussian Processes for Machine Learning*. The MIT Press, 2006. ISBN 0-262-18253-X.
- [15] Hinton, Geoffrey E. (1989) *Connectionist learning procedures*. Artificial intelligence 40.1 (1989): 185-234.
- [16] Altman, N. S. (1992) *An introduction to kernel and nearest-neighbor nonparametric regression*. The American Statistician. 46 (3): 175185. DOI:10.1080/00031305.1992.10475879
- [17] Chan, Golub, and LeVeque (1979) *Stanford CS tech report STAN-CS-79-773*. Retrieved from: <http://i.stanford.edu/pub/cstr/reports/cs/tr/79/773/CS-TR-79-773.pdf>
- [18] Hastie T., Tibshirani R., Friedman J. (2008) *The Elements of Statistical Learning*. Springer, Section 4.3, p.106-119, 2008. ISBN 978-0-387-84858-7
- [19] Cortes C., Vapnik V. (1995). *Support-vector networks*. Machine Learning. 20 (3): 273297, (1995). DOI:10.1007/BF00994018
- [20] Rami Al-Rfou et al. *Theano: A Python framework for fast computation of mathematical expressions*. arXiv preprint: 1605.02688
- [21] Fabian Pedregosa; Gal Varoquaux; Alexandre Gramfort; Vincent Michel; Bertrand Thirion; Olivier Grisel; Mathieu Blondel; Peter Prettenhofer; Ron Weiss; Vincent Dubourg; Jake Vanderplas; Alexandre Passos; David Cournapeau; Matthieu Perrot; douard Duchesnay (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research. 12: 28252830, (2011).