EALab (Eye Activity Lab): A MATLAB toolbox for variable extraction, multivariate analysis and classification of eye-movement data

Javier Andreu-Perez
Department of Computing, Imperial College London, United Kingdom
javier.andreu@imperial.ac.uk; Tel.: +44 (0)777 142 5969

Celine Solnais
Department of Marketing and Market Research, Faculty of Economic and Business Sciences, University of Granada, Spain

Kumuthan Sriskandarajah
Department of Surgery and Cancer, St Mary's Hospital, Imperial College London, United Kingdom

Abstract

Recent advances in the reliability of the eye-tracking methodology as well as the increasing availability of affordable non-intrusive technology have opened the door to new research opportunities in a variety of areas and applications. This has raised an increasing interest within disciplines such as medicine, business and education for analysing human perceptual and psychological processes based on eye-tracking data. However, most of the currently available software requires programming skills and focuses on the analysis of a limited set of eye-movement measures (e.g. saccades and fixations), thus excluding other measures of interest to the classification of a determined state or condition. This paper describes 'EALab', a MATLAB analytics toolbox aimed at easing the extraction, multivariate analysis and classification stages of eye-activity data collected from commercial and independent eye trackers. The processing implemented in this toolbox enables to evaluate variables extracted from a wide range of measures including saccades, fixations, blinks, pupil diameter and glissades. Using EALab does not require any programming and the analysis can be performed through a user-friendly graphical user interface (GUI) consisting of three processing modules: 1) eye-activity measure extraction interface, 2) variable analysis and selection interface, and 3) classification interface.

Keywords

eye tracking; machine learning; multivariate analysis; classification; computer software; neuroinformatics.

Information Sharing Statement

The software presented in this paper is available at https://ealab-matlabtoolbox.rhcloud.com. Video tutorials on how to install and use the software are provided in the tutorial section at https://ealab-matlabtoolbox.rhcloud.com/tutorial. Frequently ask questions (FAQ) and other instructions can be found at https://ealab-matlabtoolbox.rhcloud.com/faq. The software runs in MATLAB and some functions may require the use of the MATLAB Statistics toolbox. EALab has been tested in MATLAB versions v7.14 (MATLAB2012a and latest versions such as MATLAB2015a, including the latest graphical engine). A standalone compilation of EALab that runs over MATLAB Runtime (license free shared libraries to execute MATLAB applications) is also provided. EALab is distributed free and open source in BSD public license, which shares this software with the same license as other third party libraries also included in this software; these are: PRT toolbox[1] for MATLAB by Kenneth Morton and New Folder Consulting, and Variable Selection DEMO[2] package by Dimitros Ververidis and Constantine Kotropoulos.

Acknowledgments

The authors wish to thank the anonymous reviewers as well as the editorial team of Neuroinformatics for their detailed feedback and very helpful comments in enhancing the quality of this manuscript and associated toolbox.

[1] http://newfolder.github.io/

[2] http://www.mathworks.co.uk/matlabcentral/fileexchange/22970-feature-selection-using-matlab

Introduction

In recent years, the increasing number of non-invasive eye-tracking techniques relying on video-based systems has raised growing interest in eye-movement research. Improvements in the sensing capabilities of pervasive and wearable eye-tracking technology have indeed allowed more natural experimental conditions and enhanced ecological validity. In addition, the availability of affordable devices embedded in monitor screens, glasses and PC peripherals has widened the range of applications offered by the eye-tracking methodology within a variety of disciplines such as medicine, business and education.

Eye tracking provides a valuable source of physiological data regarding the allocation of information processing resources through ocular activity as well as human visual/spatial orientation processes, which are closely linked to the underlying neural networks in the brain (Mayer et al. 2004). In particular, there is extensive evidence of the association of eye-tracking measures including saccades, fixations, blinks and pupil size with cognitive functions. Specifically, pupillary response is known as a psychophysiological measure of cognitive activity and attention and was found to be positively associated with increased cognitive processing intensity and arousal (Hess and Polt 1964; Dionisio et al. 2001; Bradley et al. 2008). Thus, pupil size has been suggested as a metric for assessing memory load during complex visual tasks (Gabay et al. 2011), which are associated with higher pupil dilation (Otero et al. 2011). Pupil response has equally proved useful for clinical applications. For example, it has been suggested as a biomarker for the early detection of Alzheimer’s disease (Frost et al. 2013). Similarly, the number of saccades has been proposed as a marker for schizophrenia diagnosis (Holzman et al. 1973; Eriksson and Papanikotopoulos 1997). As regards fixation duration, it has been demonstrated to provide a reliable measure of professional expertise to discriminate between novices and experts in disciplines such as surgery (Law et al. 2004) and sport (Vickers 1993). In addition, both the timing and amount of blinks have been found to be relevant indicators for detecting driver fatigue (Eriksson and Papanikotopoulos 1997).

In the recent literature, information from multiple eye-activity variables has been used to make reliable predictions of a given cognitive state or human factor (Tsai et al. 2007; Regis et al. 2012; Li et al. 2013; Jang et al. 2014). Specifically, multivariate machine learning techniques have been successfully applied to combine a variety of measures from eye-activity data (Liang et al. 2007; Marshall 2007; Miyaji et al. 2009; Jang et al. 2014; Jaques et al. 2014). Examples of algorithms that have been employed in this aim include linear discriminant analysis (LDA) (Richstone et al. 2010; Jang et al. 2014), support vector machines (SVM) (Jang et al. 2014; Jaques et al. 2014), Bayesian inference systems (Diard et al. 2013; Jaques et al. 2014), Neural Networks (Nets) (Richstone et al. 2010; Marshall 2007), Adaptive Boosting (AdaBoost) (Miyaji et al. 2009) and Random Forest (Ren et al. 2013; Jaques et al. 2014). Whilst the individual contribution of each single eye-activity metric is not directly assessed by these algorithms, the process of *variable selection* has provided machine learning with the ability to define this contribution as well as a framework to identify the most relevant variables in cases where the most informative eye-activity measures are unknown.

To date, a number of software packages exist for the analysis of eye-tracking data with varying functionality, which we compare in Table 1. To the best of the authors' knowledge, the main toolboxes are currently as follows:

- TobiiSDK: a Software Development Kit made available by the eye-tracker manufacturer Tobii for developers to implement their own applications and export the data to MATLAB for further analysis (Tobii Technology Inc, 2014)
- EyelinkToolbox: manufactured by SR Research, it provides a series of MATLAB functions aimed at easing the creation of experiments and importing raw data collected from EyeLink eye trackers into MATLAB (Cornelissen et al. 2002).
- ILAB: this software for post-experimental eye-movement analysis is based on MATLAB and focuses on the display of gaze trajectories prior to the computation of fixations and saccades (Gitelman 2002).
- Mirametrix Toobox: another MATLAB toolbox that enables to import eye-tracking data into MATLAB (Hennessey and Duchowski 2010).
- PyGaze and its extension PyGazeAnalyzer: an open-source toolbox in Python that is focused on the creation of eye-tracking experiments as well as gaze visualization (Dalmaijer et al. 2013).
- GazeAlyze: a toolbox that provides a functionality similar to PyGaze but is based on MATLAB and includes a GUI environment (Berger et al. 2012).
- FuncSim toolbox: another open-source toolbox that focuses on the specific goal of measuring similarity between gaze scan paths (Foerster and Schneider 2013).
- EYE-EEG: a plugin of the EEGLAB MATLAB-based toolbox for multimodal data analysis enabling to import eye-tracking data (fixations, saccades, pupil rate) and synchronize it with EEG data recorded during the same experiment (Plöchl et al. 2012).

Table 1. Comparative review of existing software for eye-tracking data analysis

Software	**Inputs for Analysis**	**Primary Language**	**Experiment Loader**	**Raw Data Importer**	**GUI**	**Classification**	**Variable Selection**	**Variable Significance Test**
TobiiSDK	Fixations Saccades Pupil size Blinks	.Net C/C++ MATLAB (binding)	Yes	Yes	No	No	No	No
EyelinkToolbox	Fixations Saccades Pupil size Blinks	MATLAB	Yes[1]	Yes	No	No	No	No
Mirametrix Toobox	Fixations Saccades	MATLAB	Yes	Yes	No	No	No	No
PyGaze Analytics	Fixations Saccades	Python	Yes	Yes	No	No	No	No
ILAB	Fixations Saccades	MATLAB	Yes	Yes	Yes	No	No	No
GazeAlyze	Fixations Saccades	MATLAB	Yes	Yes	Yes	No	No	No
FuncSim	Fixations Saccades	MATLAB	Yes	Yes	No	No	No	No
EEGLAB + EYE-EGG	Fixations Saccades Pupil size Blinks	MATLAB	Yes[2]	Yes	Yes	No	No	No
EEGLAB + EYE-EGG + BCILAB	Fixations Saccades Pupil size Blinks	MATLAB	Yes[2]	Yes	Yes	Yes[2]	No	No
EALab	Gaze Fixations Saccades Pupil size Blinks Glissades	MATLAB	Yes[3]	Yes	Yes	Yes	Yes	Yes

[1] via PsychToolbox
[2] only in combination with EEG data
[3] via map files (no programming required).

Despite the existence of software and toolboxes for dealing with eye-tracking data, there is still a lack of accessibility to multivariate pattern analysis for users with no programming background. Most of the existing software indeed requires basic programming skills in languages such as Python, C/C++ and Matlab. Previous research has reported the development of an eye-detection system from a smart camera using a visual programming environment called LabView (Mehrubeoglu et al., 2011). However, the data analysis stage was not addressed and the code was not made public. In addition, the large majority of existing software toolboxes is focused on the analysis and display of gaze data (i.e. fixations and saccades) while other eye-activity measures of interest such as pupil size, blinks and glissades are not considered. As a result, to date the only way of including data from all aspects of eye activity is to do so in combination with other sensor modalities such as EEG by using an EEG toolbox.

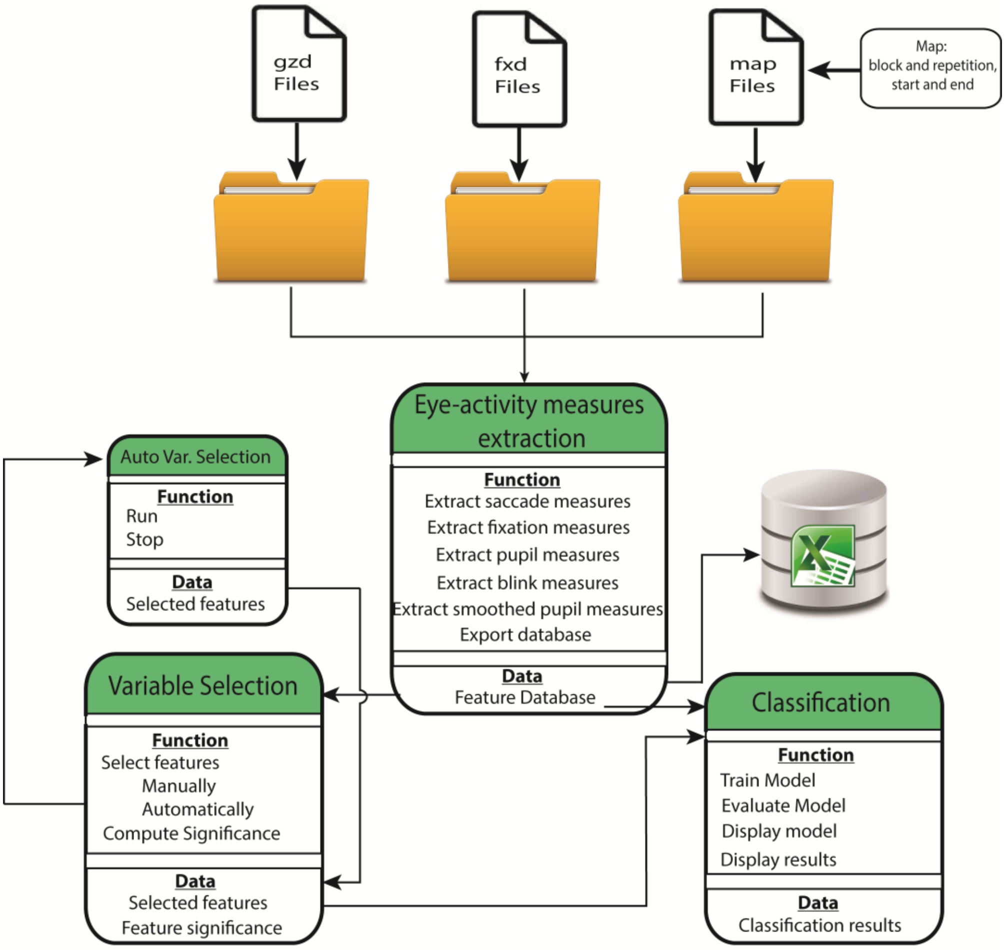


Fig. 1 EALab framework: At the top, the icons illustrate the structure of the raw data organised into three folders or directories. Below these folders, each text box represents a module, which name is highlighted with a green background. Each module corresponds to an interface of the toolbox. The actions that can be performed within each module are listed inside the box, followed with the resulting dataset below the double line.

In this context, the purpose of the present paper is to fill the identified gaps by presenting a new toolbox, called EALab. Specifically, the proposed software addresses the need for performing multivariate pattern analysis on a large range of eye-tracking data accounting for all potential measures of interest to the study of a given phenomenon. Thus, the goal of this toolbox is to enable to identify the most relevant variables (i.e. the best predictors) and discard the redundant ones while evaluating their predictive power in the context of the experiment in consideration. To do so, the toolbox consists of three main modules that correspond to each stage of the analysis workflow (Figure 1). The first module enables to perform the raw data processing by computing all potential eye-activity measures of interest as listed in Figure 1. Once this extraction stage is completed, the software allows running two additional modules of analysis on the obtained variables, namely variable selection and pattern classification. This functionality is provided by means of a user-friendly GUI that allows a large number of users from diverse disciplines and backgrounds to analyse their data without any programming requirements. In addition, the software makes it possible to export the generated descriptors to comma-separated values (CVS) to facilitate their importation into generic analytic tools such as Excel and SPSS.

Raw data requirements

The software is able to parse raw data from both Tobii © and SensoMotoric Instruments (SMI ©) eye-tracking techniques (e.g. SMI eye-tracking glasses (Bohme et al. 2006; Fromberger et al. 2012)). The standard raw files from ClearView EyeTracker recorders, i.e. GZD files (gaze data files) from the SMI BeGaze recorder and FXD formats (fixation data files) are supported. Both types of files are encoded into standard ASCII text files, which is a format commonly used in eye-tracking research through popular technology including Tobii LCD screen products such as T60, T120, TX300 (e.g. (Shahimin et al. 2014; Thiessen et al. 2014; Jaques et al. 2014; Wilkinson and Mitchell 2014; Kaneko and Tomonaga 2014). Data collected with SR research EyeLink eye-trackers are equally supported provided that EDF (Eye Link Distribution Format) files are first converted to text files (ASC format). Finally, we have included an additional functionality for data from alternative eye-tracking manufacturers and formats, which can be exploited through the ‘brand free’ option provided in the GUI. In this case, the gaze data and timestamps must be provided as a csv file where the first column contains the timestamp, the second and third columns contain the X and Y coordinates of the left eye, respectively, the fourth and fifth columns contain the X and Y coordinates of the right eye, respectively, the sixth and seventh columns contain left and right pupil diameter (optional), and the eighth and ninth columns contain the left and right distance, respectively.

Methodology

Eye-activity measure extraction (module 1)

The first module is responsible for processing the raw data collected with the eye-tracker hardware during a given experiment in order to generate the database of eye-activity measures (please see Table A.1 of Appendix 1 for the full list of measures extracted by EALab and their abbreviated names, as well as Table 2 for a definition of the terms used to refer to each measure). The raw data are typically a set of x and y coordinates for each eye, which represent the horizontal and vertical positions of gaze points on a normalized scale, respectively. This normalisation is usually performed using the screen size as a reference. EALab allows for the raw data files to contain five types of metrics as computed by the eye tracker simultaneously for both eyes, namely *gaze points, fixation points, pupil diameter, blinks* and *dynamic features* such as *glissades* and *visual angle.* When analysing eye activity, a common requirement is to study the relevance of the information provided by these measures for the prediction of a given condition of interest, which is temporally determined with respect to experimental epochs.

In the case of gaze data, two types of measures can be extracted with EALab, namely saccades and fixations. A *saccade* measures the displacement of the gaze focus between two fixation points. Sequential gaze reference samples, i.e. in the absence of a new fixation, are considered part of the saccade. Therefore, the total length of the saccade is the sum of all contiguous distances while the time elapsed from the first gaze point to the last one within a saccade forms the saccade duration. The latter is useful for computing saccade measures such as saccade velocity. The total duration of the session or epoch allows computing frequency measures. As for *fixations*, commonly used measures can be computed from their duration and frequency within a block.

As regards *pupil diameter*, a variety of measures can be extracted such as average, maximum value and correlation between both eyes, which are known for their relevance in the literature (Hess and Polt 1964; Gabay et al. 2011; Otero et al. 2011). Information about the predictability of pupil change like entropy has been evaluated as a potential predictor of surgical experience (Tien et al. 2014). Additionally, the pupil diameter rate of change provides a measure of the dynamicity of the pupil with respect to time. Differences in pupil activity can be seen during fixations, saccades, pre-blink and post-blink periods, hence the need for extracting independent measures for these periods in addition to the whole block. Although it is dependent on the applied sensing technology, the use of wavelet filters to remove low-frequency components can serve to improve the signal-to-noise ratio. The aforementioned pupil measures can additionally be extracted from the pupil size signal after wavelet smoothing as implemented in Marshall (2007). EALab decomposes the signal using Daubechies level 4 wavelets and selects its coefficients using Stein’s unbiased risk estimate to construct the smoothed signal (Mallat 2008).

Another set of measures can be extracted from blinks, which are identified by examining input raw data where the eye is closed. Blink reflex is considered a simultaneous loss of tracking of the cornea for both eyes. Depending on the blink duration, three different types of measures are computed by EALab: 1) *simple blinks* if their duration is less than 300ms, 2) *slow-blinks* if their duration is comprised between 300ms and 500ms, 3) *micro-sleeps* if their duration is comprised between 500ms and 1s, and 4) any further eye closure or phenomena where the pupil is not detected by the eye tracker for more than 1s. We consider the samples as missing data if the segment indicating loss of tracking has not been homogenous (i.e. intermittent sequence of null and valid data) or if the loss of tracking has only involved one eye. A commonly used measure of blinks is the frequency

3of occurrence within an experimental block. These measures can be considered variables of interest when delineated with respect to a representative label (a task or condition) of the block.

Table 2. Definition of the terms used in relation to the eye-activity measures extracted by EALab

Saccades	
Term	*Description*
Number	Count of the number of saccades during a block.
Frequency	Total number of saccades divided by the duration of a block in seconds
Length	Distance covered by a saccade, which is spanned by line segment connecting two contiguous gaze samples.
Maximum	Maximum duration in seconds or maximum distance covered by a saccade in a block.
Average	Average duration in seconds or average distance covered by a saccade in a block.
Velocity	Ratio between the length covered and the duration of a saccade.
Fixations	
Term	*Description*
Number	Count of the number of fixations during a block
Frequency	Total number of fixations divided by the duration of a block in seconds.
Maximum	Maximum duration in seconds of a fixation during a block.
Average	Average duration in seconds of a fixation during a block.
Pupil	
Term	*Description*
Maximum	Maximum value of the pupil diameter or pupil rate of change in a block.
Average	Average value of the pupil diameter or pupil change of rate in a block.
Entropy	Computed as the Shannon entropy of the pupil diameter signal during a block.
Correlation	Spearman correlation between the pupil diameter signals of both eyes during a block.
Rate of change	Sum of the ratios between the differences in pupil diameter of adjacent samples and the sample frequency, divided by the total number of samples in a block.
Wavelet smoothed	Filtered pupil diameter using a de-noising process through wavelets (Daubechies 4 wavelets and Stein's unbiased risk as threshold selection)
Wavelet Energy	Sum of the sums of the squared-wavelets coefficients for eight levels of decomposition.
Pre-blink	Pupil diameter before a blink (simple blink, micro-sleep or short-sleep).
Post-blink	Pupil diameter after a blink (simple blink, micro-sleep or short-sleep).
Blinks	
Blink	Simple blinks with duration less than 300ms
Slow blinks	Blinks with duration between 300ms and 500ms
Micro sleeps	Blinks with duration between 500ms and 1s
Eye closure	Missing data from both eyes during more than 1s

Variable selection and analysis (module 2)

The set of samples handled by EALab is composed of observations or independent variables (the eye-activity measures) and a target variable (the condition to be predicted). The second module of the toolbox is aimed at scoring the variables of the database generated in the previous module so as to select the best predictors. Indeed, depending on the design of the experiment and the target variable to be predicted, some of the extracted eye-activity measures may be redundant and therefore penalise the accuracy of the applied classifier. To address this issue, the analysis performed via this module enables to discard less relevant variables and select a subset of variables that improve a classification criterion. In addition, the selected variables can be ranked based on an average score and the robustness of this score (significance analysis), which is performed via a permutation test to provide information about the relevance and quality of the selected measures as predictors of the target variable. The statistical significance may equally serve for conducting hypothesis-driven research over the overall set of eye-activity measures. A detailed explanation of the methods in question is presented in the subsections below.

Overview of variable selection methods

Several methods exist for selecting a set of variables. When a scoring approach is used and a threshold is applied over this ranking to define a subset of variables, this selection method is known as *filtering*. Other machine learning algorithms make use of a built-in mechanism, which makes it possible to compute a score from its internal model parameters. This is the case of support vector machines, random forest, logistic regression and Gaussian mixture models, for example. These types of variable selection strategies are the so-called *embedded* methods. Alternatively, instead of a direct scoring or internal built-in mechanism, the scoring function may be based on an evaluation procedure through a classifier that is used as a “black box” to score the predictive power of a subset of variables. These variable selection methods are known as *wrappers*. As a difference to the embedded methods, wrappers enable to consider the evaluation inside the variable search. Thus, cross-validation folds can be implemented inside the variable search process to be able to estimate how the classifier will perform in practice with a given test set of variables, which is the approach adopted in EALab.

To implement the wrapper approach, one option is to perform an exhaustive search given all possible combinations of variables, which can require a very large number of computations. Instead, it is desirable to adopt an efficient strategy that searches for an optimal subset without the need for testing all possible combinations. The *greedy stepwise search* approach provides a solution to this issue. Common search strategies within this methodology make use of *sequential* algorithms, which can be of two types, namely *forward selection* and *backward elimination.* The former starts with an empty set of variables and sequentially adds new ones until there is no further improvement given the decision criterion. On the contrary, the backward elimination algorithm starts with a candidate subset that contains all variables and sequentially eliminates variables until the criterion worsens. Sequential forward selection performs best when the set of optimal variables is small. At the beginning of the procedure, large combinations of variables can potentially be evaluated, however, the more variables to evaluate, the smaller the flexibility of the search. Besides, some variables can become of minor importance by the end of the procedure. On the contrary, sequential backward selection performs best when the optimal set of variables is large, but a disadvantage is that variables are not further evaluated once they have been discarded. These strategies are typically implemented by evaluating the performance of a learning machine in a wrapper fashion.

The weaknesses of both sequential search types can be addressed by implementing a bi-directional correction in the sequential steps, called *floating* sequential (FS) variable selection (Pudil et al. 1994; Ververidis and Kotropoulos 2008, 2006), which is categorized into two types: sequential forward floating selection (SFFS) and sequential backwards floating selection (SBFS). Thus, in this approach the search is not strictly sequential towards a single direction, it rather 'floats' forwards first and then backwards (SFFS) or backwards first and then forwards (SBFS) with the objective of improving the criterion. An explanation of the floating mechanism implemented in EALab is presented in Appendix 3, taking the example of the SFFS.

By applying a scoring function over this set of samples, a set of importance scores can be obtained for each variable. This score is typically interpreted as increasing with the predictive power of a specific variable with respect to the target. This approach is used to directly obtain a ranking of a set of variables based on their scores. Examples of *scoring approaches* that do not require learning include those using correlation-based metrics such as mutual information, information gain and distribution comparison tests. On the other hand, *embedded* methods can be considered scoring approaches that require learning; thus its mechanism is based on the importance of each variable, particularly for solving a classification problem. However, if the ranking is used to select a set of variables, it is necessary to define a cut-off threshold, which could be arbitrary (selected by the user) or based on statistical assumptions about the underlying null distribution.

EALab implements Relief-F, which is considered one of the most popular approaches to rank variables and determine their importance. It has been used in a variety of bioinformatics areas such as tumour classification (Metsis et al. 2012) and gene selection (McKinney et al. 2013; Cordell 2009; Ye et al. 2008; Saeys et al. 2007). Most scoring approaches assume the conditional independence of the variables; as a result, they are less reliable for experiments where the variables may involve interactions. Relief-F does not make this assumption. It aims at estimating the quality of the variables according to how well their values distinguish between instances that are near to each other. An explanation of the mechanism of this algorithm is provided in Appendix 3.

To test the robustness of the most important variables selected, the statistical significance of the scores can be computed. In this aim, EALab applies a non-parametric method that consists in computing the scores by permuting the target variable several times, i.e. a permutation test (Efron and Tibshirani 1994). This approach produces different ranking values obtained through several runs of the scoring function under the null hypothesis. An empirical probability distribution of the scores can be generated for each eye-activity measure from these rankings. The resulting distributions can be used to test the null hypothesis. If the original score value of a variable lies far outside most of the mass of the distribution, then this variable can be considered as significantly predictive of the target variable. This allows computing the statistical significance of the original test. The statistical significance of the importance score of a variable provides an indicator of its *stability*. Whilst the statistical significance may suggest the superior reliability of some of the scores, this information should not be used as a method for selecting the most important variables. To this aim, it is more suitable to apply a search algorithm over the whole set of variables, which will enable to find the best subset of variables that minimizes a given decision criterion (based on the classification error for instance). Following this, a permutation test can be applied over the selected variables to provide them with a score and determine the robustness of these scores.

Eye-activity variable selection process in EALab

A wrapper approach is first designed; a Bayesian classifier with Gaussian probability density function is used to predict the class label. Bayesian classifiers (Williams and Barber 1998; Rasmussen 2006) have been widely used in a variety of pattern recognition applications from signals such as biomarker discovery (Chu et al. 2005), emotion recognition, (Ververidis and Kotropoulos 2008) and the detection of cognitive states (Wang and Mitchell 2002). This type of classifier presents the advantages of being very fast to compute as well as providing reasonably good generalization properties. This classifier is wrapped into a k-fold cross-validation (CV) evaluation process, being *k* the number of folds (*k* partitions of the whole data). One fold is taken as a test set while the remaining *k*-1 is used for training the algorithm. The average numbers of positive classification values is taken as criterion function or heuristics. Alternatively, this criterion can be defined through a statistical test by computing the variance of the correct-classification rates (CCR) within each cross-validation. This can be done by defining either a lower bound or both lower and upper bounds on the confidence interval of the current variable criterion in order to accept a new variable (Ververidis and Kotropoulos 2008). Depending on the user’s choice, the software implements a sequential selection search that is either backward or forward, with or without floating, using as a criterion the fraction of correctly classified observations in each cross-validation fold, divided by *k*. Once the variables have been selected, the Relief-F algorithm is applied by EALab to repeatedly compute the ranks over a selected number of times by applying permutation tests (Efron and Tibshirani 1994). Selected confidence intervals are derived from the null distribution of the scores of all the variables and the position of the original score in the distribution is assessed for significance.

Classification algorithms (module 3)

The toolbox enables to opt for different classification algorithms. Specifically, four learning algorithms and two meta-classifiers are implemented; these have been reported to be successful in recognizing mental states in previous work. From left to right in Figure 4, the first three are linear discriminant analysis (LDA) (Duda et al. 1999), quadratic discriminant analysis (QDA) (McLachlan 2004) and a Bayesian classifier with Gaussian density function and maximum-a-posteriori inference (Murphy 2012). For example, in Richstone et al. (2010), LDA has previously been used to assess surgical skills from eye-activity measures whilst Bayesian inference systems have been applied to stress detection by Barreto et al. (2007). In LDA all classes are assumed to have the same covariance matrix while classes may have different covariance in QDA. Another classification algorithm available in EALab is support vector machine (SVM) (Cristianini and Shawe-Taylor 2000; Cortes and Vapnik 1995), which is a popular large margin classifier algorithm as it is aimed at finding a hyper-plane that maximizes the distance between classes (Drucker et al. 1997). As an example of application, SVM and Bayesian classifiers have been used to detect drivers' fatigue in Senaratne et al. (2007). The fourth learning algorithm that can be selected in EALab is a neural network (Neural nets), which is a three-layer feed-forward neural network with sigmoid activation functions. The back-propagation algorithm is implemented as described in Duda et al. (1999). For example, neural nets have been applied to grade surgeon experience based on eye-tracking data (Richstone et al. 2010; Marshall 2007).

The next two algorithms are meta-classifiers because they implement a base classifier and then apply a given method to enhance its performance. These algorithms are composed of a combination of simpler classifiers, which together make a "better" one. The first is *adaptive boosting* (AdaBoost) (Zhu et al. 2009; Hastie et al. 2009), which aims at minimizing the weighted error by sequentially training a set of simple and fast classifiers. Each classifier outputs a weight impurity score for each observation. The resultant weights are used for the training of the next sequential (boosted) classifier. The second meta-classifier implemented in EALab is *random forest* (RF) (Breiman 2001), which creates a group of decision trees where each tree is able to see a subset of the overall dataset. Each of these subsets is built by means of a method called bootstrap-aggregating (Breiman 2001). RF combines the inference provided by each decision tree by means of a voting strategy (e.g. taking the class with the highest number of votes).

User guide

In this section we provide guidance on how to use the EALab toolbox for eye-tracking research based on a dataset example. The demo data are provided along with the toolbox in the EALab source files folder. One set of demo data was collected using a remote Tobii eye-tracking system of 50 Hz (model 2140C) and another demo experiment was recorded with wireless SensoMotoric Instruments (SMI) eye-tracking glasses of 30 Hz (model ETG 2W). Once the appropriate eye-tracking manufacturer is selected in the first interface of EALab, the modules of the toolbox and steps of analysis are the same regardless of the eye tracker. The user guide provided in this section is based on the demo data collected with the Tobii eye-tracking system. The experiment consisted in a simulated bi-manual motor task performed by thirteen subjects, among which seven were sleep-deprived and six were rested. Each subject repeated the task in three different sessions. Each session was divided into three time intervals (epochs) corresponding to each of the three separate actions of the motor tasks.

In this example, the multivariate analysis methodology provided by EALab enables to 1) generate an exhaustive dataset of eye-activity measures extracted from the raw data, 2) identify the eye-tracking variables that best predict fatigue, and 3) predict whether a subject is fatigued or rested by allocating them into the correct group or class at a given time interval. The detailed design as well as the interpretation of the results for this particular experiment are out of the scope of this paper.

Data preparation and file directories

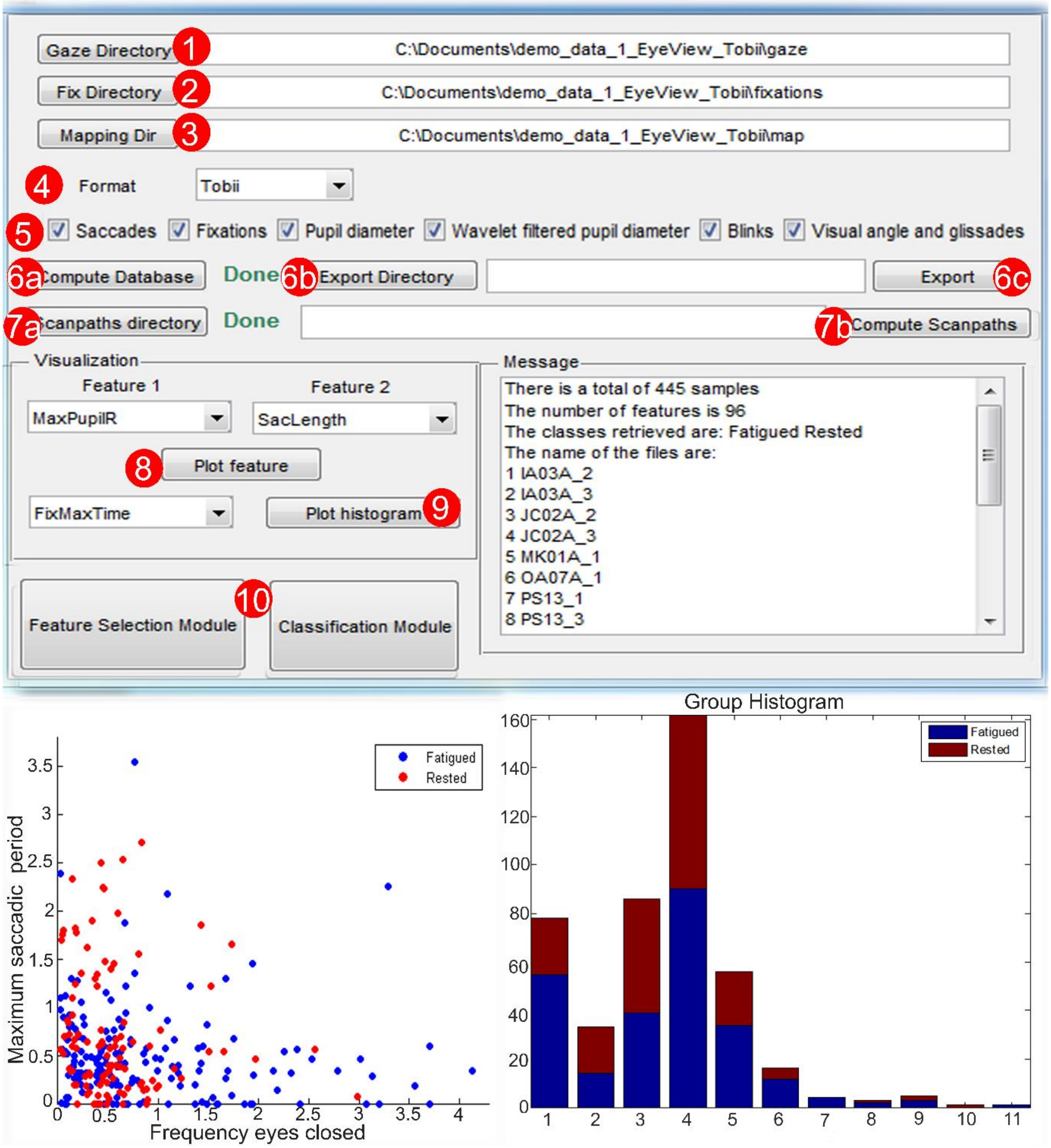
In order to perform the analysis workflow provided by the software, the raw eye-tracking data must be organised into three separate directories: one for the gaze data, a second one for the fixation data and a third one for the experimental design. When the selected brand is SMI, the fixation directory is not required. In the case of SR Research/EyeLink data format, only a single ASC file is required for each experimental session.

The raw data generated by the eye-tracking recorder software are typically contained in two different types of files, namely the gaze and fixation data files. These files must be renamed using the following name format: [NAME]_[SESSION ID].gzd for the gaze files and [NAME]_[SESSION ID].fxd for the fixation files, where the name is a unique alphanumeric name and the session ID is an integer. The same file name and session ID must be used in both formats for the same subject and session. e.g. SUBJ1_1.gzd and SUBJ1_1.fxd. Once the files have been renamed accordingly, the user places the files in two separate folders: one for the fixations files and one for the gaze files.

In addition, a third type of files is required to provide EALab with the necessary information that will define the experimental design and task. These must be created by the user and placed into a separate folder, which we will refer to as the “map directory”. This directory must consist of a series of Excel files (one for each subject), with the following name format: [NAME]_[SESSION ID]_[CLASS LABEL].xls. Each Excel file must contain four columns in the following order: “Block”, “Repetition”, “Start”, and “Finish”. The first column is an index to determine different types of block; the second column is an index of the repetition of the block; the third column is an indicator of the time when the epoch starts and the fourth column indicates the time at which the epoch finishes (both expressed as a number of gaze samples, knowing that gaze samples multiplied by the frequency of recording is equal to timestamp). The time count starts at the beginning of the recording and the start and finish times can be annotated using a standard chronometer during the experiment. Thus, there is no need for restarting the recording for every block or task. Alternatively, this information can be obtained through eye-tracking visualisation tools provided by the manufacturer such as Tobii Studio.

Eye-activity Measure Extraction Interface

When EALab is launched, the extraction interface is the first to open, as shown in Figure 2. The first three buttons at the top of this interface allow the user to indicate the path of the directories where the raw data and map files are located. Once this information is provided, the user can select the eye-tracking measures of interest that they wish to extract from the raw data (highlighted by number 5 in the figure). These will make the database that is generated by clicking on the "compute database" button (6a). A description of the outcome of the database generation is shown in the message panel box, which also lists the correspondence between each subject ID and file name. The interface gives the option to export the database to an Excel file for further exploration (6b and 6c). The database consists in one column for each eye-tracking measure computed during the extraction process and the last three columns are dedicated to the Epoch, Class and Subject ID, respectively. Furthermore, the interface provides visualization capabilities to explore the generated variables. The visualization functionality enables to select two eye-activity variables of interest to visualise them in a scatter plot and generate a histogram for a specific variable, grouped by classes.



Gaze Directory: C:\Documents\demo_data_1_EyeView_Tobii\gaze

Fix Directory: C:\Documents\demo_data_1_EyeView_Tobii\fixations

Mapping Dir: C:\Documents\demo_data_1_EyeView_Tobii\map

Format: Tobii

☑ Saccades ☑ Fixations ☑ Pupil diameter ☑ Wavelet filtered pupil diameter ☑ Blinks ☑ Visual angle and glissades

Compute Database: Done

Scanpaths directory: Done

Feature 1: MaxPupilR

Feature 2: SacLength

FixMaxTime

Message:

There is a total of 445 samples
The number of features is 96
The classes retrieved are: Fatigued Rested
The name of the files are:
1 IA03A_2
2 IA03A_3
3 JC02A_2
4 JC02A_3
5 MK01A_1
6 OA07A_1
7 PS13_1
8 PS13_3

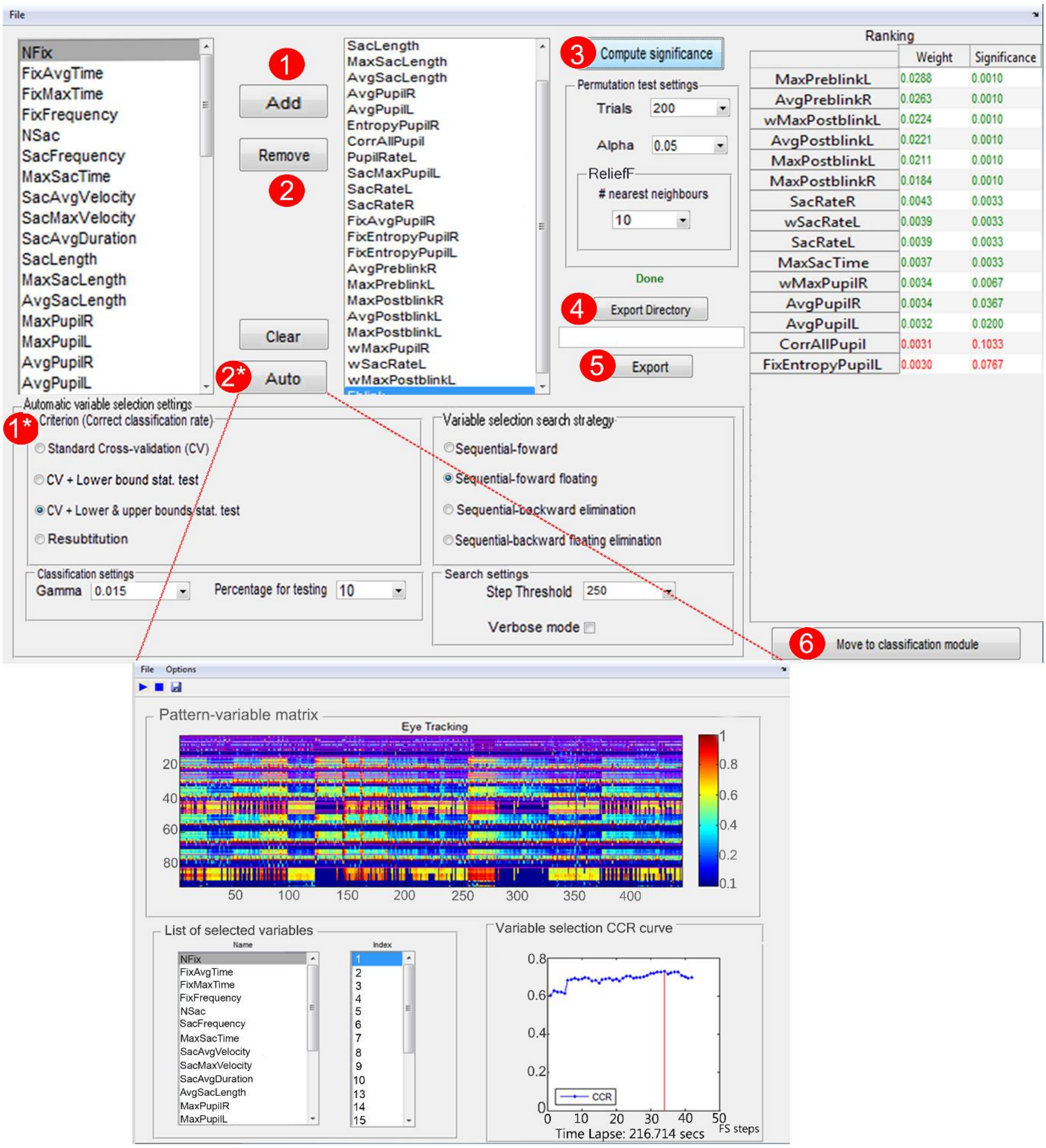
Fig. 2 Eye-activity Measure Extraction Interface. The numbers in red highlight the key elements of this interface. They are ordered with respect to the suggested sequence of steps a user can perform, as follows: (1)(2)(3) show the selection of the file directories where the raw data are located; (4) is the scrolling menu for selecting the raw data format; (5) corresponds to the independent tick boxes for selecting the eye-activity measures of interest; (6a) is the button to import the raw data and compute the measures selected in (5); (6b) enables to select the directory where to export the database; (6c) launches the exportation process. The buttons starting with 7 are not compulsory for the analysis. They provide the option to generate graphic files to show the scanpath signatures from gaze and fixations. To do so, (7a) opens a window where the user can select the file directory where to save the figures and (7b) initiates the generation (which can take a lot of processing time in case of a large dataset like in the example of demo 1). (8)(9) show the visualization options; and finally (10) allows moving on to one of the other two interfaces for further analysis.

Eye-activity Measure Selection Interface

The selection interface (Figure 3) is able to perform two types of analysis: automatic variable selection and ranking. The selection process can be performed manually by selecting the desired variables from the panel and clicking on “add” (or “remove” to unselect a variable). Alternatively, the toolbox provides an auto-mode variable selection. This can be chosen by clicking on “auto”, which starts a new process and opens a new interface (bottom part of Figure 3). The configuration of the automatic selection process can be customised by updating the settings available in this interface. Specifically, the software provides four different alternatives for setting the criterion (correct classification rate, CCR) and calculating the importance scores of the variables: 1) cross-validation, 2) cross-validation plus a lower bound test on the CCR confidence interval, 3) cross-validation plus lower and upper bound tests on the CCR confidence interval, and 4) resubstitution. In the resubstitution option, the overall dataset is used for training and then again for the testing phase. Nevertheless, it is worth mentioning that this method is known for yielding optimistically biased CCR. The other three methods are more robust as they employ independent sets for training and testing. In addition, the criterion settings allow setting the threshold parameter of the confidence interval (gamma) that is used in case a statistical variable elimination method is selected, as well as the percentage of data that will be used for testing (e.g. if it is set at 10%, then 10 cross-validation folds will be created from the data). As for the selection strategy, four methods are available, namely sequential forward, sequential-forward floating, sequential-backward elimination and sequential-backward floating elimination.

Taking the examples of the demo data, a set of relevant eye-tracking measures can be found with a maximum correct classification rate of 0.75. Hence, one can rank the selected subset of measures by relevance for the condition/stimulus of interest. The larger the number of trials selected during the significance test the more robust the analysis and the more reliable the ranking. The highest ranked eye-activity measure for this example of subsets is the frequency of eye closure (FEyesClosed) [Permutation test, p=0.0010, N=200, alpha=0.05]. Previous research has studied blink rate as a possible measure of fatigue and concluded that the rate of blinking is related to ‘mental tension’, a type of relief mechanism controlled by the nervous system (Stern et al., 1994). Additionally, one can find that significant measures such as MaxSacTime [Permutation test, p=0.001, N=200, alpha=0.05], SacAvgVelocity [Permutation test, p=0.001, N=200, alpha=0.05] and SacFrequency [Permutation test, p=0.026, N=200, alpha=0.05] are related to saccades. Previous research has hypothesised that the level of fatigue can be determined by slow saccades with noticeable slowing at large amplitudes. Additionally, ocular measures of pupil size have been found to be relevant for mental workload

(Recarte and Nunes, 2003) and a surrogate marker of brain module activation (Granholm and Steinhauer, 2004). Therefore, users can benefit from the variable exploration in EALab to guide their analysis over these different measures and examine the discrimination power for the concerned conditions/stimuli and the selected variables in the classification interface.



NFix
FixAvgTime
FixMaxTime
FixFrequency
NSac
SacFrequency
MaxSacTime
SacAvgVelocity
SacMaxVelocity
SacAvgDuration
SacLength
MaxSacLength
AvgSacLength
MaxPupilR
MaxPupilL
AvgPupilR
AvgPupilL

SacLength
MaxSacLength
AvgSacLength
AvgPupilR
AvgPupilL
EntropyPupilR
CorrAllPupil
PupilRateL
SacMaxPupilL
SacRateL
SacRateR
FixAvgPupilR
FixEntropyPupilR
FixEntropyPupilL
AvgPreblinkR
MaxPreblinkL
MaxPostblinkR
AvgPostblinkL
MaxPostblinkL
wMaxPupilR
wSacRateL
wMaxPostblinkL

Permutation test settings
Trials 200
Alpha 0.05
ReliefF
# nearest neighbours 10

Ranking

	Weight	Significance
MaxPreblinkL	0.0288	0.0010
AvgPreblinkR	0.0263	0.0010
wMaxPostblinkL	0.0224	0.0010
AvgPostblinkL	0.0221	0.0010
MaxPostblinkL	0.0211	0.0010
MaxPostblinkR	0.0184	0.0010
SacRateR	0.0043	0.0033
wSacRateL	0.0039	0.0033
SacRateL	0.0039	0.0033
MaxSacTime	0.0037	0.0033
wMaxPupilR	0.0034	0.0067
AvgPupilR	0.0034	0.0367
AvgPupilL	0.0032	0.0200
CorrAllPupil	0.0031	0.1033
FixEntropyPupilL	0.0030	0.0767

Automatic variable selection settings
Criterion (Correct classification rate)
○ Standard Cross-validation (CV)
○ CV + Lower bound stat. test
● CV + Lower & upper bounds stat. test
○ Resubtitution

Variable selection search strategy
○ Sequential-foward
● Sequential-foward floating
○ Sequential-backward elimination
○ Sequential-backward floating elimination

Classification settings
Gamma 0.015
Percentage for testing 10

Search settings
Step Threshold 250
Verbose mode □

Name	Index
NFix	1
FixAvgTime	2
FixMaxTime	3
FixFrequency	4
NSac	5
SacFrequency	6
MaxSacTime	7
SacAvgVelocity	8
SacMaxVelocity	9
SacAvgDuration	10
AvgSacLength	13
MaxPupilR	14
MaxPupilL	15

Time Lapse: 216.714 secs

Fig. 3 Eye-activity Measure Selection Interface. The numbers in red show the elements of this interface. They are ordered with respect to the suggested sequence of step a user can perform, as follows: (1)(2) buttons enable to select variables manually by choosing from the left-hand side list of extracted measures to add them to the right-hand side list of selected variables and by removing the unwanted one; (1*) to customise the settings for the auto-variable selection mode, (2*) to launch the auto-variable selection panel, which is shown in the screenshot below the interface window. Once the variables have been selected, either manually or through the auto mode process, then the user can click on “compute significance” (3). (4) enables to select the folder to where the user wishes to export the ranking results in comma-separated value (csv) format and (5) performs the

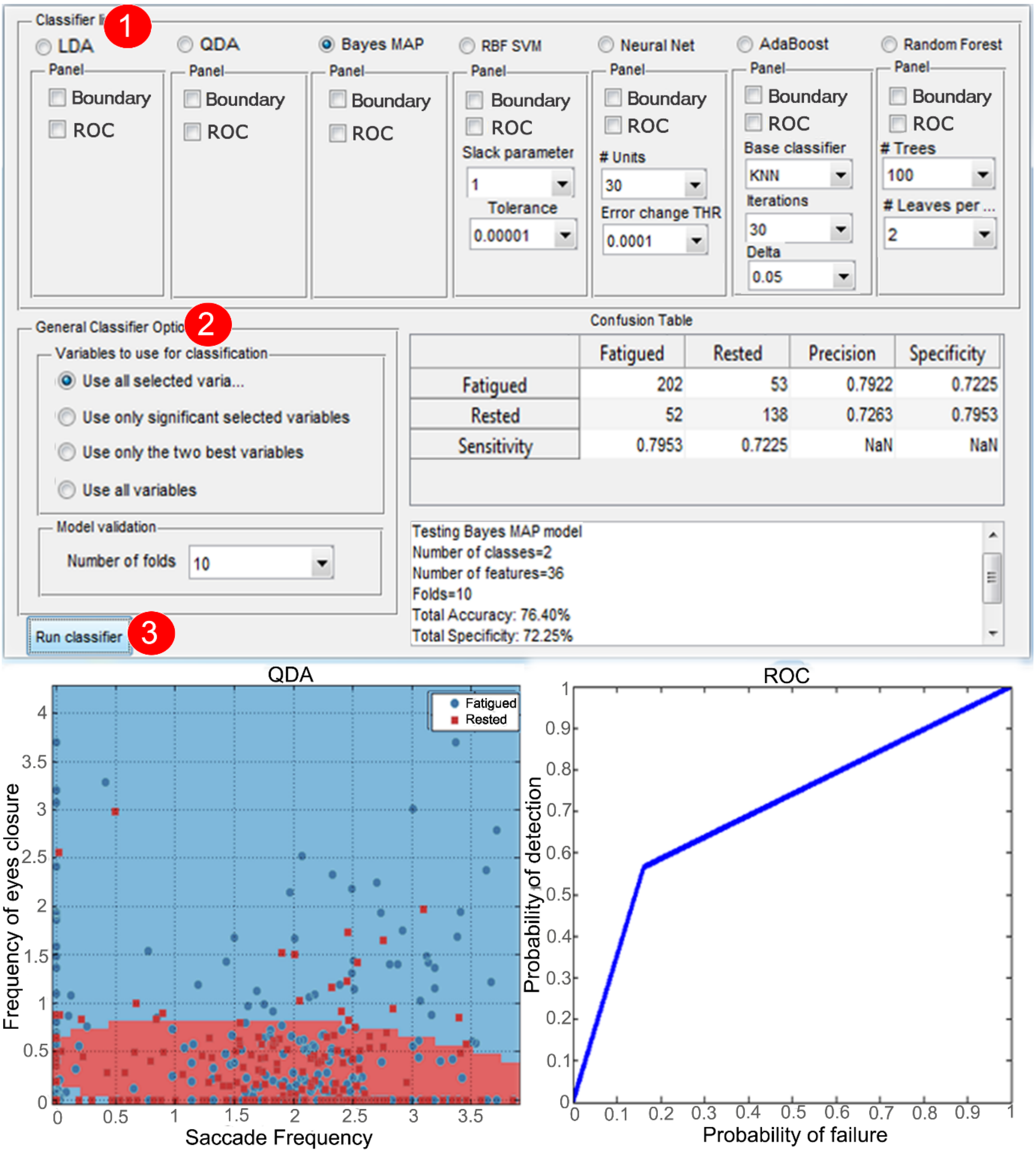
exportation process to the selected folder. Finally the user can choose to move on to the classification module by clicking on (6).

Classification Interface

The third interface is designed to classify the subjects with respect to the target condition in consideration using the selected variables. Taking the example of the demo data, the objective of this module is to correctly predict the group to which each subject belongs by identifying whether they are fatigued or rested based on the eye-tracking variables. To do so, the classification process is to be able to verify the ability of the eye-activity variables to predict the condition.

The first panel of this interface is the “Classifier selection” (Figure 4). This stage enables the user to choose from six different classifiers, which correspond to the ones previously discussed in the methodology section. The next stage of this interval provides general settings of classification through which the user is able to select the number of variables that will be used to train the classifier according to four different options: 1) use all the variables selected in the previous module, 2) use all selected variables which score is significant, 3) use the best two variables only, and 4) use all available variables regardless of whether they have been selected in the previous module or not. It is worth mentioning that the third option may not provide the best results, however, it enables the user to explore two eye-activity variables independently.

In case the user is interested in testing the classification power of two independent eye-activity variables, the eye-activity measure selection interface (Figure 3) enables to manually select the two variables of interest from the left-hand side panel of extracted measures ensuring that all other measures are removed from the right-hand side panel of selected measures. Using two variables it is possible to plot the 2D decision boundaries of the selected classifier. Additionally, in binary classification scenarios, the toolbox provides the possibility to plot the ROC curve. All classifiers are evaluated using a k-fold cross validation for which the number of folds can be selected through the interface. The results of the classifier are displayed in a confusion table where the rows correspond to the predicted values and the columns correspond to the real values. The values of sensitivity, precision and specificity are also appended to the confusion matrix. Finally, the message box located at the bottom of the window provides the total values for accuracy, specificity, sensitivity, precision and area-under the curve (AUC) for binary class problems. Taking the fatigue example into consideration, the classification through multivariate quadratic discriminant analysis and following a k-fold cross-validation evolution enables to obtain 76.4% accuracy, 72.6% precision, 79.22% sensitivity and 72.25% specificity considering all selected measures. It is worth noting that using a single measure would not enable to discriminate certain conditions in a reliable way, hence the importance of a multivariate approach.



Classifier li

1

○ LDA ○ QDA ● Bayes MAP ○ RBF SVM ○ Neural Net ○ AdaBoost ○ Random Forest

Panel: □ Boundary □ ROC

RBF SVM: Slack parameter 1; Tolerance 0.00001

Neural Net: # Units 30; Error change THR 0.0001

AdaBoost: Base classifier KNN; Iterations 30; Delta 0.05

Random Forest: # Trees 100; # Leaves per ... 2

General Classifier Optio

2

Variables to use for classification

● Use all selected varia...

○ Use only significant selected variables

○ Use only the two best variables

○ Use all variables

Model validation

Number of folds 10

Run classifier

3

Confusion Table

	Fatigued	Rested	Precision	Specificity
Fatigued	202	53	0.7922	0.7225
Rested	52	138	0.7263	0.7953
Sensitivity	0.7953	0.7225	NaN	NaN

```
Testing Bayes MAP model
Number of classes=2
Number of features=36
Folds=10
Total Accuracy: 76.40%
Total Specificity: 72.25%
```

Fig 4. Classification Interface. The numbers in red show the components of this interface. They are ordered with respect to the suggested sequence of steps a user can perform, as follows: (1) selection of the classifier and configuration of its settings, (2) setting the variables to use for the classification and the number of folds for the cross-validation, and finally (3) running the classifier. In case the boundary and ROC (Receiver Operator Characteristics) tick boxes are marked, then the plots will appear as shown in the bottom part of the figure. The one in the left-hand side represents the decision boundaries of the classifier applied to the example data provided with the software, which was a quadratic discriminant analysis (QDA) in this example. The samples are shown in dark red and blue for the conditions of interest (i.e. fatigued and rested, respectively) considering the following eye-movement measures: the frequency of eye closure (FEyesClosed) and saccade frequency (SacFrequency), as shown in the axes labels. The prediction made by the classifier about the conditions associated with each sample is marked by the light red and blue areas. As for the graph in the right-hand side of

Figure 4, this shows the ROC curve based on the probability of detection against the probability of failure, therefore providing a visual representation for evaluating the performance of the classifier.

Discussion

This paper developed EALab, a MATLAB-based toolbox for analysing a wide range of eye-tracking data while providing an intuitive and easy-to-use GUI intended for users from a variety of areas. A gap was identified through a detailed comparison of existing toolboxes, highlighting the need for this type of software. In addition, similar analytical approaches to the one implemented by EALab have been proposed in the research literature (Jang et al. 2014; Richstone et al. 2010; Marshall 2007; Ren et al. 2013; Frost et al. 2013), which supports the relevance of the proposed toolbox for the analysis of eye activity as a predictor of human cognition within a growing, heterogeneous research community. The need for performing the analysis steps provided by EALab using a large set of eye-activity measures was equally identified within our own research team, as well as the need for a GUI and reproducibility. As regards the methods implemented in this toolbox, users are encouraged to read the references provided in the corresponding sections as well as additional material for a complete background on the detailed concepts underlying each method.

Throughout its development, the software has been tested and assessed by scientific personal with no programming skills in order to design it accordingly with a particular attention on user-friendliness. Other research groups have developed graphical tools for designing experiments and loading raw data. As a result of our experience, we have observed that non-specialist users are more comfortable with editing and creating files with Microsoft Excel than with ad-hoc interfaces created for this purpose. For this reason, EALab addresses the need for defining the mapping files through a protocol created over Excel files (.xlsx), with which users are commonly familiar regardless of their technical background. Thus, this format also presents the benefit of simply requiring widespread software such as Microsoft Office suite and any open-source variants including LibreOffice/OpenOffice. The only additional requirement for using EALab is a licensed copy of MATLAB and the statistics Toolbox for some functions.

Following the release of the first version of this toolbox, we aim to work on improving and extending its functionality. In the current version, the toolbox is able to import raw data from Tobii (Clearview) and SMI (BeGaze) eye-trackers, while SR Research data are only supported if previously converted to ASC format. This represents a limitation in the usability and contribution of EALab, which is due to the lack of access to EyeLink eye-tracking systems within our centre. In order to improve the compatibility of EALab and facilitate its use for a wide range of researchers, we aim to further develop the possibility to import EyeLink data in the near future and are hopeful that other research groups will contribute by providing raw data from different manufacturers to increase the number of technologies supported by EALab. Additionally, in the current version, the software makes use of the event markers provided in the raw data to distinguish between saccades and fixations, which are defined as per the standard of the device manufacturer. In future releases, we could work on providing the functionally of re-labelling these markers, allowing for the user to re-program these events by re-defining the fixation radius, for example. Additionally, we will consider improving the processing speed for large datasets by parallelizing some procedures and making use of the GPU for CPU-intensive operations.

Overall, EALab is aimed at completing the range of existing software in support for eye-tracking research. We hope that a wide number of researchers will benefit from its use and that a fruitful cooperative environment of research will be developed around it.

Conflict of Interest

All authors state that they have no conflicts of interest to declare, financial or otherwise.

Appendices

Appendix 1. Table of eye-activity measures and abbreviations used in EALab

Table A1. Eye-activity measures extracted by EALab

NFix	*Number of fixations*	MaxPostblinkL	*Maximum pupil diameter during post-blink periods (left eye)*
FixAvgTime	*Average fixation time*	wMaxPupilR	*Wavelet-smoothed maximum pupil diameter (right eye)*
FixMaxTime	*Maximum fixation time*	wMaxPupilL	*Wavelet-smoothed maximum pupil diameter (left eye)*
FixFrequency	*Fixation frequency*	wAvgPupilR	*Wavelet-smoothed average pupil diameter (right eye)*
NSac	*Number of saccades*	wAvgPupilL	*Wavelet-smoothed average pupil diameter (left eye)*
SacFrequency	*Saccade frequency*	wEntropyPupilR	*Wavelet-smoothed pupil entropy (right eye)*
MaxSacTime	*Maximum saccade time*	wEntropyPupilL	*Wavelet-smoothed pupil entropy (left eye)*
SacAvgVelocity	*Average saccade velocity*	wCorrAllPupil	*Wavelet-smoothed pupil diameter correlation between both eyes*
SacMaxVelocity	*Maximum saccade velocity*	wPupilRateL	*Wavelet-smoothed pupil diameter rate of change (left eye)*
SacAvgDuration	*Average saccade duration*	wPupilRateR	*Wavelet-smoothed pupil diameter rate of change (right eye)*
SacLength	*Total saccade length*	wave_energyR	*Wavelet energy of pupil diameter (right eye)*
MaxSacLength	*Maximum saccade length*	wave_energyL	*Wavelet energy of pupil diameter (left eye)*
AvgSacLength	*Average saccade length*	wSacMaxPupilR	*Wavelet-smoothed maximum pupil diameter during saccades (right eye)*
MaxPupilR	*Maximum pupil diameter (right eye)*	wSacMaxPupilL	*Wavelet-smoothed maximum pupil diameter during saccades (left eye)*
MaxPupilL	*Maximum pupil diameter (left eye)*	wSacAvgPupilR	*Wavelet-smoothed average pupil diameter during saccades (right eye)*
AvgPupilR	*Average pupil diameter (right eye)*	wSacAvgPupilL	*Wavelet-smoothed average pupil diameter during saccades (left eye)*
AvgPupilL	*Average pupil diameter (left eye)*	wSacEntropyPupilR	*Wavelet-smoothed pupil entropy during saccades (right eye)*
EntropyPupilR	*Pupil entropy (right eye)*	wSacEntropyPupilL	*Wavelet-smoothed pupil entropy during saccades (left eye)*
EntropyPupilL	*Pupil entropy (left eye)*	wSacCorrPupil	*Correlation of the wavelet-smoothed pupil diameters of both eyes during saccades*
CorrAllPupil	*Pupil diameter correlation between both eyes*	wSacRateL	*Wavelet-smoothed pupil rate of change during saccades (left eye)*
PupilRateL	*Pupil diameter rate of change (left eye)*	wSacRateR	*Wavelet-smoothed pupil rate of change during saccades (right eye)*
PupilRateR	*Pupil diameter rate of*	SacWave_energyR	*Energy of the wavelet-smoothed*

	change (right eye)		*pupil diameter during saccades (right eye)*
SacMaxPupilR	*Maximum pupil diameter during saccades (right eye)*	SacWave_energyL	*Energy of the wavelet-smoothed pupil diameter during saccades (left eye)*
SacMaxPupilL	*Maximum pupil diameter during saccades (left eye)*	wFixMaxPupilR	*Wavelet-smoothed maximum pupil diameter during fixations (right eye)*
SacAvgPupilR	*Average pupil diameter during saccades (right eye)*	wFixMaxPupilL	*Wavelet-smoothed maximum pupil diameter during fixations (left eye)*
SacAvgPupilL	*Average pupil diameter during saccades (left eye)*	wFixAvgPupilR	*Wavelet-smoothed average pupil diameter during fixations (right eye)*
SacEntropyPupilR	*Pupil entropy during saccades (right eye)*	wFixAvgPupilL	*Wavelet-smoothed average pupil diameter during fixations (left eye)*
SacEntropyPupilL	*Pupil entropy during saccades (left eye)*	wFixEntropyPupilR	*Wavelet-smoothed pupil entropy during fixations (right eye)*
SacCorrPupil	*Correlation between the pupil diameters of both eyes during saccades*	wFixEntropyPupilL	*Wavelet-smoothed pupil entropy during fixations (left eye)*
SacRateR	*Pupil rate of change during saccades (right eye)*	wFixCorrPupil	*Correlation of the wavelet-smoothed pupil diameters of both eyes during fixations*
SacRateL	*Pupil rate of change during saccades (left eye)*	wFixRateL	*Wavelet-smoothed rate of change of pupil diameter during fixations (left eye)*
FixMaxPupilR	*Maximum pupil diameter during fixations (right eye)*	wFixRateR	*Wavelet-smoothed rate of change of pupil diameter during fixations (right eye)*
FixMaxPupilL	*Maximum pupil diameter during fixations (left eye)*	FixWave_energyR	*Wavelet-smoothed pupil diameter energy during fixations (right eye)*
FixAvgPupilR	*Average pupil diameter during fixations (right eye)*	FixWave_energyL	*Wavelet-smoothed pupil diameter energy during fixations (left eye)*
FixAvgPupilL	*Average pupil diameter during fixations (left eye)*	wAvgPreblinkR	*Wavelet-smoothed average pupil diameter during pre-blink periods (right eye)*
FixEntropyPupilR	*Pupil entropy during fixations (right eye)*	wAvgPreblinkL	*Wavelet-smoothed average pupil diameter during pre-blink periods (left eye)*
FixEntropyPupilL	*Pupil entropy during fixations (left eye)*	wMaxPreblinkR	*Wavelet-smoothed maximum pupil diameter during pre-blink periods (right eye)*
FixCorrPupil	*Correlation of both pupil diameters during fixations*	wMaxPreblinkL	*Wavelet-smoothed maximum pupil diameter during pre-blink periods (left eye)*
FixRateL	*Pupil rate of change during fixations (left eye)*	wAvgPostblinkR	*Wavelet-smoothed average pupil diameter during post-blink periods (right eye)*
FixRateR	*Pupil rate of change during fixations (right eye)*	wAvgPostblinkL	*Wavelet-smoothed average pupil diameter during post-blink periods (left eye)*
AvgPreblinkR	*Average pupil diameter during pre-blink periods*	wMaxPostblinkR	*Wavelet-smoothed maximum pupil diameter during post-blink*

	(right eye)		*periods (right eye)*
AvgPreblinkL	*Average pupil diameter during pre-blink periods (left eye)*	wMaxPostblinkL	*Wavelet-smoothed pupil diameter maximum during post-blink periods (left eye)*
MaxPreblinkR	*Maximum pupil diameter during pre-blink periods (right eye)*	FBlink	*Frequency of blinks*
MaxPreblinkL	*Maximum pupil diameter during pre-blink periods (left eye)*	FSlowBlink	*Frequency of slow blinks*
AvgPostblinkR	*Average pupil diameter during post-blink periods (right eye)*	FMicroSleep	*Frequency of micro sleeps*
AvgPostblinkL	*Average pupil diameter during post-blink periods (left eye)*	FEyesClosed	*Frequency for any type of blink or voluntary eye closure larger than one second.*
MaxPostblinkR	*Maximum pupil diameter during post-blink periods (right eye)*	AvgAngVel	*Average angular velocity (visual angle)*
MaxAngVel	*Maximum angular velocity (visual angle)*	AvgAngAcc	*Average angular acceleration (visual angle)*
MaxAngAcc	*Maximum angular acceleration (visual angle)*	NFGlissades	*Number of fast glissades*
AvgFGlissades	*Average fast glissades duration*	NSGlissades	*Number of slow glissades*
AvgSGlissades	*Average slow glissades duration*	TGlissades	*Total number of glissades*

Appendix 2. Sequential Forward Floating Selection (SFFS)

The SFFS approach is composed of the following series of steps considering the forward case:

1) Select the best variable that maximizes the criterion,

$$n^{+} = \arg\max_{n \notin N_i} \mathrm{J}(N_i + n);\ N_k + n^{+};\ i = i + 1$$

2) Select the worst variable,

$$n^{-} = \arg\max_{n \notin N_i} \mathrm{J}(N_i - n)$$

3) Check,

$$f(N_i, n^{-}) = \begin{cases} goto\ step\ 1, & \mathrm{J}(N_i - n^{-}) > \mathrm{J}(N_i) \\ goto\ step\ 2, & otherwise \end{cases}$$

where function $\mathrm{J}(.)$ represents the function describing the criterion; N is a vector of selected variables that is initialized as $N = \varnothing$ in the first iteration and only goes through step 1; n is a single variable only, represented as n^{-} when it is the one yielding the worse criterion and n^{+} for the one yielding the best criterion; i is a counter of the number of elements in N.

Appendix 3. Relief-F Algorithm

The Relief-F algorithm assigns a score to each variable as an indicator of its relevance. It gives a higher score to the variables that are better able to discriminate an observation from its neighbours of different classes. Relief-F (Kononenko et al. 1997) was introduced as an extension of a previous Relief algorithm (Kira and Rendell 1992) to improve its capacity to deal with incomplete, noisy and multiclass problems. It starts with selecting a random instance (R) from the set of observations; it searches for another two types of near observations within the neighbour of size *s*: 1) nearest hits (*H*), which are observations from the same class and 2) nearest misses (*M*). The algorithm to compute Relief-F weights is as follows:

$$W[n] := W[n-1] - \sum\nolimits_{j=1}^{k} diff(n, R_i, H_j) \,/\, (t \cdot k) + \sum_{c \neq class(R_i)} \left(\frac{P(C)}{1 - P(class(R_i))} \sum\nolimits_{j=1}^{k} diff(n, R_i, M_j(C))) \,/\, (t \cdot k) \right)$$

$$\forall n \in N; \;\; \forall j \in \{1, 2, \ldots, k\}$$

where W is the weight of each feature and $W_0[n] = 0$, *t* is the total number of observations. $diff$ is a function that takes the values of the random sample instance (R) and nearest hit (H) or miss (M) and calculates its distance. k is a user-defined number of nearest-neighbours and $M(C)$ is the nearest miss from class C. The implementation from MATLAB’s Bioinformatics Toolbox is included in the toolbox.

References

Barreto, A., Zhai, J., & Adjouadi, M. (2007). Non-intrusive physiological monitoring for automated stress detection in human-computer interaction. In Human–Computer Interaction (pp. 29-38). New York: Springer.

Berger, C., Winkels, M., Lischke, A., & Höppner, J. (2012). GazeAlyze: a MATLAB toolbox for the analysis of eye movement data. Behavior research methods, 44(2), 404-419.

Bohme, M., Meyer, A., Martinetz, T., & Barth, E. Remote eye tracking: State of the art and directions for future development. In Proc. of the 2006 Conference on Communication by Gaze Interaction (COGAIN), 2006 (pp. 12-17).

Bradley, M.M., Miccoli, L., Escrig, M.A. and Lang, P.J. (2008). The Pupil as a Measure of Emotional Arousal and Autonomic Activation/ Psychophysiology, 45(4), 602-607.

Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

Chu, W., Ghahramani, Z., Falciani, F., & Wild, D. L. (2005). Biomarker discovery in microarray gene expression data with Gaussian processes. Bioinformatics, 21(16), 3385-3393.

Cordell, H. J. (2009). Detecting gene–gene interactions that underlie human diseases. Nature Reviews Genetics, 10(6), 392-404.

Cornelissen, F. W., Peters, E. M., & Palmer, J. (2002). The Eyelink Toolbox: eye tracking with MATLAB and the Psychophysics Toolbox. Behavior Research Methods, Instruments, & Computers, 34(4), 613-617.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. United Kingdom: Cambridge university press.

Dalmaijer, E. S., Mathôt, S., & Van der Stigchel, S. (2013). PyGaze: An open-source, cross-platform toolbox for minimal-effort programming of eyetracking experiments. Behavior research methods, 1-9.

Diard, J., Rynik, V., & Lorenceau, J. (2013). A Bayesian computational model for online character recognition and disability assessment during cursive eye writing. Frontiers in psychology, 4(843), 1-15.

Dionisio, D. P., Granholm, E., Hillix, W. A., & Perrine, W. F. (2001). Differentiation of deception using pupillary responses as an index of cognitive processing. Psychophysiology, 38(2), 205-211.

Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. Advances in neural information processing systems, 9, 155-161.

Duda, R. O., Hart, P. E., & Stork, D. G. (1999). Pattern classification. New York: John Wiley & Sons.

Efron, B., & Tibshirani, R. J. (1994). An introduction to the bootstrap. New York: Chapman & Hall

Eriksson, M., & Papanikotopoulos, N. (1997). Eye-tracking for detection of driver fatigue. Proc. of IEEE Conference on Intelligent Transportation System, (pp. 314-319). New York: IEEE

Foerster, R. M., & Schneider, W. X. (2013). FuncSim Toolbox for MATLAB: Computation of eye tracking scanpath similarity. Resource document. http://pub.uni-bielefeld.de

Fromberger, P., Jordan, K., Steinkrauss, H., von Herder, J., Witzel, J., Stolpmann, G., et al. (2012). Diagnostic accuracy of eye movements in assessing pedophilia. The journal of sexual medicine, 9(7), 1868-1882.

Frost, S., Kanagasingam, Y., Sohrabi, H., Bourgeat, P., Villemagne, V., C Rowe, C., et al. (2013). Pupil Response Biomarkers for Early Detection and Monitoring of Alzheimer's Disease. Current Alzheimer Research, 10(9), 931-939.

Gabay, S., Pertzov, Y., & Henik, A. (2011). Orienting of attention, pupil size, and the norepinephrine system. Attention, Perception, & Psychophysics, 73(1), 123-129.

Gitelman, D. R. (2002). ILAB: a program for postexperimental eye movement analysis. Behavior Research Methods, Instruments, & Computers, 34(4), 605-612.

Granholm, E., Steinhauer, S.R., 2004. Pupillometric measures of cognitive and emotional processes. International Journal of Psychophysiology, 52, 1-6.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., & Tibshirani, R. (2009). The elements of statistical learning (Vol. 2, Vol. 1). New York: Springer.

Hennessey, C., & Duchowski, A. T. An open source eye-gaze interface: Expanding the adoption of eye-gaze in everyday applications. In Proc. of the 2010 Symposium on Eye-Tracking Research & Applications, 2010 (pp. 81-84). New York: ACM.

Hess, E. H., & Polt, J. M. (1964). Pupil size in relation to mental activity during simple problem-solving. Science, 143(3611), 1190-1192.

Holzman, P. S., Proctor, L. R., & Hughes, D. W. (1973). Eye-tracking patterns in schizophrenia. Science, 181(4095), 179-181.

Jang, Y.-M., Mallipeddi, R., Lee, S., Kwak, H.-W., & Lee, M. (2014). Human intention recognition based on eyeball movement pattern and pupil size variation. Neurocomputing, 128(0), 421-432

Jaques, N., Conati, C., Harley, J. M., & Azevedo, R. Predicting Affect from Gaze Data during Interaction with an Intelligent Tutoring System. In Intelligent Tutoring Systems, 2014 (pp. 29-38): Springer

Kaneko, T., & Tomonaga, M. (2014). Differential reliance of chimpanzees and humans on automatic and deliberate control of motor actions. Cognition, 131(3), 355-366.

Kira, K., & Rendell, L. A. A practical approach to feature selection. In Proceedings of the ninth international workshop on Machine learning, 1992 (pp. 249-256). San Francisco: Morgan Kaufmann Publishers Inc.

Kononenko, I., Šimec, E., & Robnik-Šikonja, M. (1997). Overcoming the myopia of inductive learning algorithms with RELIEFF. Applied Intelligence, 7(1), 39-55.

Law, B., Atkins, M. S., Kirkpatrick, A. E., & Lomax, A. J. Eye gaze patterns differentiate novice and experts in a virtual laparoscopic surgery training environment. In Proceedings of the 2004 symposium on Eye tracking research & applications, 2004 (pp. 41-48). New York: ACM.

Li, W.-C., Chiu, F.-C., Kuo, Y.-s., & Wu, K.-J. (2013). The investigation of visual attention and workload by experts and novices in the cockpit. In Proceeding Engineering Psychology and Cognitive Ergonomics. Applications and Services (pp. 167-176).New York: Springer.

Liang, Y., Reyes, M. L., & Lee, J. D. (2007). Real-time detection of driver cognitive distraction using support vector machines. IEEE Transactions on Intelligent Transportation Systems, 8(2), 340-350.

Mallat, S. (2008). A wavelet tour of signal processing: the sparse way. New York: Academic press.

Marshall, S. P. (2007). Identifying cognitive state from eye metrics. Aviation, space, and environmental medicine, 78(5), 165-175.

Mayer, A. R., Dorflinger, J. M., Rao, S. M., & Seidenberg, M. (2004). Neural networks underlying endogenous and exogenous visual–spatial orienting. Neuroimage, 23(2), 534-541.

Mehrubeoglu, M., Pham, L.M., Le, H.T., Muddu, R., Ryu, D., 2011. Real-time eye tracking using a smart camera. In Proceedings IEEE Workshop Applied Imagery Pattern Recognition (pp. 1-7). New York: IEEE.

McKinney, B. A., White, B. C., Grill, D. E., Li, P. W., Kennedy, R. B., Poland, G. A., et al. (2013). ReliefSeq: A Gene-Wise Adaptive-K Nearest-Neighbor Feature Selection Tool for Finding Gene-Gene Interactions and Main Effects in mRNA-Seq Gene Expression Data. PloS one, 8(12), 1-12.

McLachlan, G. (2004). Discriminant analysis and statistical pattern recognition. New York: John Wiley & Sons.

Metsis, V., Huang, H., Andronesi, O. C., Makedon, F., & Tzika, A. (2012). Heterogeneous data fusion for brain tumor classification. Oncology reports, 28(4), 1413-1416.

Miyaji, M., Kawanaka, H., & Oguri, K. Driver's cognitive distraction detection using physiological features by the adaboost. Proceedings of the IEEE International Conference on Intelligent Transportation Systems, 2009. ITSC'09. 12th International (pp. 1-6). New York: IEEE

Murphy, K. P. (2012). Machine learning: a probabilistic perspective. Cambridge, MA: MIT press.

Otero, S. C., Weekes, B. S., & Hutton, S. B. (2011). Pupil size changes during recognition memory. Psychophysiology, 48(10), 1346-1353.

Plöchl, M., Ossandón, J. P., & König, P. (2012). Combining EEG and eye tracking: identification, characterization, and correction of eye movement artifacts in electroencephalographic data. Frontiers in human neuroscience, 6(278), 1-23.

Pudil, P., Novovičová, J., & Kittler, J. (1994). Floating search methods in feature selection. Pattern recognition letters, 15(11), 1119-1125.

Rasmussen, C. E & Williams C.K.I (2006). Gaussian processes for machine learning. Cambridge, MA: MIT Press.

Recarte, M.A., Nunes, L.M., 2003. Mental workload while driving: effects on visual search, discrimination, and decision making. Journal of experimental psychology: Applied, 9(2), 119-137.

Regis, N., Dehais, F., Tessier, C., & Gagnon, J.-F. (2012). Ocular metrics for detecting attentional tunnelling. Human Factors and Ergonomics Society–Chapter Europe, Resource document. University of Tolouse. http://oatao.univ-toulouse.fr/11690/

Ren, P., Barreto, A., Gao, Y., & Adjouadi, M. (2013). Affective assessment by digital processing of the pupil diameter. IEEE Transactions on Affective Computing, 4(1), 2-14.

Richstone, L., Schwartz, M. J., Seideman, C., Cadeddu, J., Marshall, S., & Kavoussi, L. R. (2010). Eye metrics as an objective assessment of surgical skill. Annals of surgery, 252(1), 177-182.

Saeys, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. Bioinformatics, 23(19), 2507-2517.

Senaratne, R., Hardy, D., Vanderaa, B., & Halgamuge, S. (2007). Driver fatigue detection by fusing multiple cues. In Advances in Neural Networks–ISNN 2007 (pp. 801-809). New York: Springer.

Shahimin, M., Mohammed, Z., Saliman, N., Mohamad-Fadzil, N., Razali, N., Mutalib, H., et al. (2014). The Use of an Infrared Eye Tracker in Evaluating the Reading Performance in a Congenital Nystagmus Patient Fitted with Soft Contact Lens: A Case Report. In Current Trends in Eye Tracking Research (pp. 123-128). New York: Springer.

Stern, J., Boyer, D., Schroeder, D.J., 1994. Blink rate as a measure of fatigue: A review. Human Factors, 36(2), 285-97.

Thiessen, A., Beukelman, D., Ullman, C., & Longenecker, M. (2014). Measurement of the visual attention patterns of people with aphasia: A preliminary investigation of two types of human engagement in photographic images. Augmentative and Alternative Communication, 30(2), 120-129.

Tien, T., Pucher, P. H., Sodergren, M. H., Sriskandarajah, K., Yang, G.-Z., & Darzi, A. (2014). Differences in gaze behaviour of expert and junior surgeons performing open inguinal hernia repair. Surgical endoscopy, 1-9.

Tobii Eye Tracker Tobii Technology. Resource document. http://www.tobii.com

Tsai, Y.-F., Viirre, E., Strychacz, C., Chase, B., & Jung, T.-P. (2007). Task performance and eye activity: predicting behavior relating to cognitive workload. Aviation, space, and environmental medicine, 78(5), 176-185.

Ververidis, D., & Kotropoulos, C. (2006). Fast sequential floating forward selection applied to emotional speech features estimated on DES and SUSAS data collections. Proc. European Conference in Signal Processing (pp. 1-5). Alborg: EURASIP.

Ververidis, D., & Kotropoulos, C. (2008). Fast and accurate sequential floating forward feature selection with the Bayes classifier applied to speech emotion recognition. Signal Processing, 88(12), 2956-2970.

Vickers, J. N. (1993). Toward defining the role of gaze control in complex targeting skills. In B. David, G. Alastair, & C. Karen (Eds.), Visual search 2 (pp. 265-285). New York: Taylor & Francis

Wang, X., & Mitchell, T. (2002). Detecting cognitive states using machine learning. Resource document. http://www.cs.cmu.edu/~xuerui/papers/report.pdf

Wilkinson, K. M., & Mitchell, T. (2014). Eye Tracking Research to Answer Questions about Augmentative and Alternative Communication Assessment and Intervention. Augmentative and Alternative Communication, 30(2), 1-14.

Williams, C. K., & Barber, D. (1998). Bayesian classification with Gaussian processes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(12), 1342-1351.

Ye, K., Feenstra, K. A., Heringa, J., IJzerman, A. P., & Marchiori, E. (2008). Multi-RELIEF: a method to recognize specificity determining residues from multiple sequence alignments using a Machine-Learning approach for feature weighting. Bioinformatics, 24(1), 18-25.

Zhu, J., Zou, H., Rosset, S., & Hastie, T. (2009). Multi-class adaboost. Statistics and its Interface, 2(3), 349-360.