abcd

Quentin Geissmann

Supervised by Giorgio Gilestro

Imperial College London

September 7, 2014

Todo list

QG: conclusion should do more justice 1

Abstract

Electroencephalography (EEG) is a very widespread technique which routinely used as a diagnostic tool for brain dysfunctions and to investigate neurological phenomenons. Because of its relatively non-invasive nature, it is widely used to determine and quantify sleep stages in humans and model mammals such as rodents. Historically, labelling of EEG recordings has been performed visually by trained human experts. This task is extremely tedious an quite subjective. Despite recent efforts to develop a automatic classifier of sleep stages little adoption has occurred, and manual scoring is still the standard.

This study presents a high accuracy classifier of sleep stage that could lead to an efficient a software tool for biologists. Exhaustive feature extraction based on discrete wavelet decomposition was used in combination with random forest classifier and was demonstrated to achieve an overall accuracy as high as 92%. In order to optimise feature extraction and pave the way for a future software implementation, `pyrem`, a `python` package was also developed. `pyrem` is here shown to outperform alternative runtime implementation by several orders of magnitudes.

Contents

1 Introduction **1**

2 Material and Methods **5**

2.1 Data acquisition 5

2.2 Data preprocessing 5

2.3 Feature extraction from time series 6

2.4 Addition of temporal features 7

2.5 Stratified Cross Validation and sampling 8

2.6 Random forests analysis 8

2.7 Performance assessments 9

2.8 Statistical analysis 9

3 Results **11**

3.1 A new efficient `python` package 11

3.2 Twenty variables can generate accurate predictions 12

3.3 Including temporal information improves prediction 14

3.4 Structural differences with ground truth 14

3.5 Attribution of confidence score to predictions 16

4 Discussion **19**

4.1 Software package for feature computation 19

4.2 Exhaustive feature extraction 19

4.3 Random forest classification 21

4.4 Rigorous and comprehensive model evaluation 22

4.5 Quality of the raw data 22

4.6 Overall results 23

Appendices **30**

A `pyrem` documentation **30**

Acronyms

ANN Artificial Neural Network. 3, 20

EEG ElectroEncephaloGram. 1, 2, 5–7, 13, 17, 19, 20, 24

EMG ElectroMyoGram. 1, 2, 5–7, 13, 17, 24

HMM Hidden Markov Model. 3

NREM Non-Rapid Eye Movement. 1, 2, 14, 17, 21

REM Rapid Eye Movement. 1, 2, 8, 14–17, 20, 23

SVM Support Vector Machine. 3, 20

1 Introduction

QG: conclusion should do more justice

Sleep is considered to be a ubiquitous and necessary behaviour amongst animals[1,2]. However, its real physiological functions remain debated. In vertebrate, electrophysiological recordings, in particular, ElectroEncephaloGram (EEG); the recording of the global electrical activity in the brain, but also ElectroMyoGram (EMG), which records muscular activity, have extensively used to study the structure of sleep during the last century[3, 4]. They have the advantage of being non-invasive an relatively high throughput. Today, EEG remains one of the main asset in the study sleep physiology.

Rodents, in particular mice and rats, have proved very successful model for understanding of the mechanisms of sleep in mammals[5]. Classically, three main distinct types of sleep related behaviours: wakefulness, Non-Rapid Eye Movement (NREM) sleep and Rapid Eye Movement (REM) sleep are referred as *vigilance states*. Vigilance states are usually defined on the basis of EEG and EMG (fig. 1). When awake (WAKE), animals tend to have a high muscular activity which translates as a high amplitude voltage changes in the EMG. During wakefulness, EEG is dominated by a relatively low amplitude oscillations of frequency between six and ten hertz often referred as theta waves. In contrast, NREM sleep, also called slow wave sleep, is a period of muscular inactivity (low EMG) dominated by slow oscillations (below 4Hz) of high amplitude named delta waves. The third state, REM sleep, is characterised by a complete lack of muscular activity (atony) and an EEG activity very similar to the awake state. REM sleep is the least prevalent of all three stages, and represents generally 20% of all sleeping time. The prevalence of these three states as well as their temporal organisation are extremely important observations in sleep research

Although definitions of sleep stages appear straightforward, in practice, many cases are ambiguous. For instance, it is difficult to characterise transitions between two states. In addition, there are many sources of variability including how surgery was performed, the type of instrument used and inter-animal variability. The quality of the acquisition can also be made considerably worst by noisy signals or by the presence of artefacts. For these reasons, sleep scoring, *i.e.*

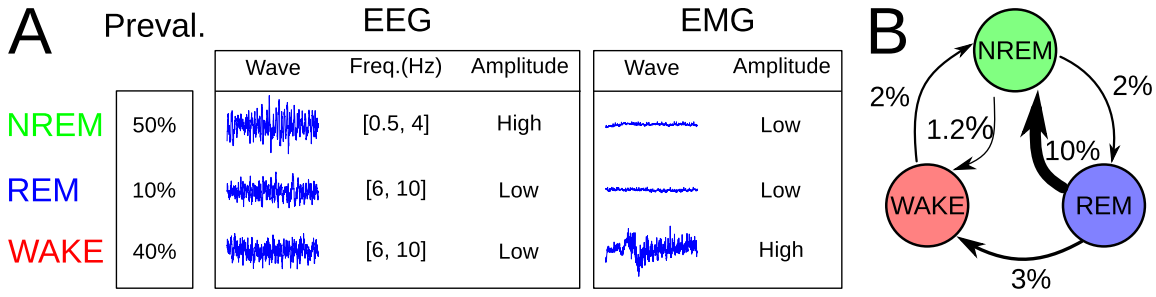

Figure 1: *Structural description of sleep stages.* **A**: Characteristics of the three vigilance states: NREM, REM and wakefulness. Typically, frequency and amplitude of the ElectroEncephaloGram (EEG), and amplitude of the ElectroMyoGram (EMG) are used by experts to infer vigilance state. The frequency ranges and prevalences are approximate values for healthy adult animals in normal conditions. Each wave shows a representative five second epoch with high signal to noise ratio. **B**: Empirical transition probabilities between consecutive five second epochs. The width of the arrow is proportional to transition probability. Probabilities of remaining in each state are implied.

the attribution of discrete vigilance states to electrophysiological time series, is traditionally performed by trained human experts. Such manual annotation is very time consuming; several hours of work have been reported in order to score 24h of recording. This severely limits data throughput, and human subjectivity is likely to introduce systematic bias. Indeed, it is expected that scoring will be performed differently by each expert, making result difficult to reproduce independently. Often, two experts score the same data, in order to ensure satisfying agreement. Although, manual scorers are generally reported as being very consensual with each other[6, 7], it can be argued that experts most likely work in the same laboratory and trained one another, or were trained by the same third person. In this context, agreement between experts does not account for the variability between communities of researchers, and cannot be used to assess reproducibility.

In order to overcome both speed and subjectivity limitations, efforts have long been directed towards automation of sleep scoring[8, 9]. However, little adoption has occurred and very few available implementations, in the form of software that biologists could use, have been developed. Typically, two different approaches to classification have been followed: unsupervised or supervised learning.

Unsupervised learning [10, 11] has the advantage of making no assumption about the nature of the different vigilance states, and how they should be defined. Therefore, this approach can lead to the discovery of truly new states. One issue is that the choice of the variables used for clustering is very critical. Often, variables such as frequency domain variables are in fact chosen

in order to generate clusters that will match human defined clusters. In addition, unsupervised methods may lack robustness[11] in so far as the cannot easily include covariates explaining, for instance, variability between recording equipments.

Another approach is to assume human annotations are, although imperfect, biologically relevant and generally consistent, and therefore to use supervised learning techniques[7, 12–15]. Of course, if human decisions were biased, such a method may suffer from the same bias. However, a vast corpus of experimental work has provided hypothesis about function of these states which tends to validate the actual 'existence' of these discrete vigilance states. Building a classifier that would produce a consensual prediction of vigilance states could be seen as an attempt to formalised and rationalise the definition of such states. This would improve future research without denying decades of sleep neurobiology.

Many supervised learning techniques such as from Support Vector Machines(SVMs)[12], Artificial Neural Networks(ANNs)[13], to Hidden Markov Models(HMMs)[14, 15] have been investigated. In general, the first step is to compute features on consecutive segments of annotated electrophysiological signals know as epochs. Then, the relation between the response variable(annotation) and the independent variables (features) can be modelled. Either epochs are considered to be independent from one another or time-dependent structures are explicitly modelled (*e.g.* using HMMs). Time aware modelling has the advantage of accounting for the interdependence of consecutive epochs (see fig. 1B). However, it generally does not model non-linear relationships between large numbers of predictors as well as classical classifiers.

Recently, promising results were obtained for automatic scoring of human sleep stages by performing an exhaustive feature extraction, including variables resulting from discrete wavelet decomposition[7]. Then, the authors compared several classifiers and found that random forest[16] were, overall, the most accurate predictors.

The study herein bases itself on these promising results by computing an even larger number of features. An important addition was the computation of time-aware features[17, 18] which significantly improved accuracy. Furthermore, rigorous stratified cross-validation[19] procedure and comparisons of sleep structure were performed. These improvement altogether contributed

to achieve a very satisfying overall accuracy of 92%. In order to pave the way to an implementation of an ubiquitous sleep scoring software. `pyrem`, a new `python` package was also build to facilitate efficient feature extraction. This new package is here demonstrated to be significantly more performance than alternative implementation.

2 Material and Methods

2.1 Data acquisition

In this study, 12 male mice (Tg(Gal-cre)KI87Gsat/Mmucd, FVB/N-Crl:CD1(ICR) hybrid strain), between 8 and 12 week old were used. Animal were housed under a 12h light/dark light cycle. Small (diam. = 1mm) holes were drilled in the scull of anaesthetised animals. For the EEG, the reference-ground electrode was placed into the parietal bone (Bregma -1.5, mediolateral (ML) +1.5) and the other electrode was placed into the frontal bone (Bregma +1.5, ML -1.5). For EMG acquisition, three polytetrafluoroethylene-insulated stainless steel electrodes were placed into the neck muscles.

Both signals were recorded by a miniature 'neurologger'[20]. The recording device applies band-pass analogue filtering between 1 and 70Hz and converts both analogue signals to digital time series at a sampling rate of approximately 200.0Hz. All 12 animals were monitored for approximately 24h.

Sleep scoring was performed in a semi-automatic fashion by a trained expert. A first, human assisted, pass was applied to generate preliminary annotations on the basis of logical rules[6]. Then, the expert visually inspected and, when required, corrected the annotations. Annotation were generated for consecutive epochs of approximately 5.0s.

Data acquisition and manual annotation were performed by Dr. Valentina Ferretti and Eleonora Steinberg prior to this project.

2.2 Data preprocessing

EEG and EMG signals were resampled from approximately 200.0Hz to 256.0Hz using conservative sinc interpolation[21]. A sampling frequency of $f_s = 256.0$Hz is convenient since is implies that discrete wavelet decomposition (see subsection 2.3 and fig. 2) will separate frequencies above 4.0Hz from those below 4.0Hz (since $4 = 256.0/2^6$). This frequency is typically uses as a cut-off value between theta and delta waves [22]. In addition, EEG and EMG signals

were standardised ($E[x] = 0, Var[x] = 1$) to account for the variability in baseline amplitude due to acquisition. Vigilance state anotations were resampled at exactly 0.20Hz using nearest neighbour interpolation.

2.3 Feature extraction from time series

A wavelet transform based feature extraction strategy was adopted. In summary, discrete wavelet decomposition was used on both EEG and EMG in order to separate frequency bands (fig. 2).

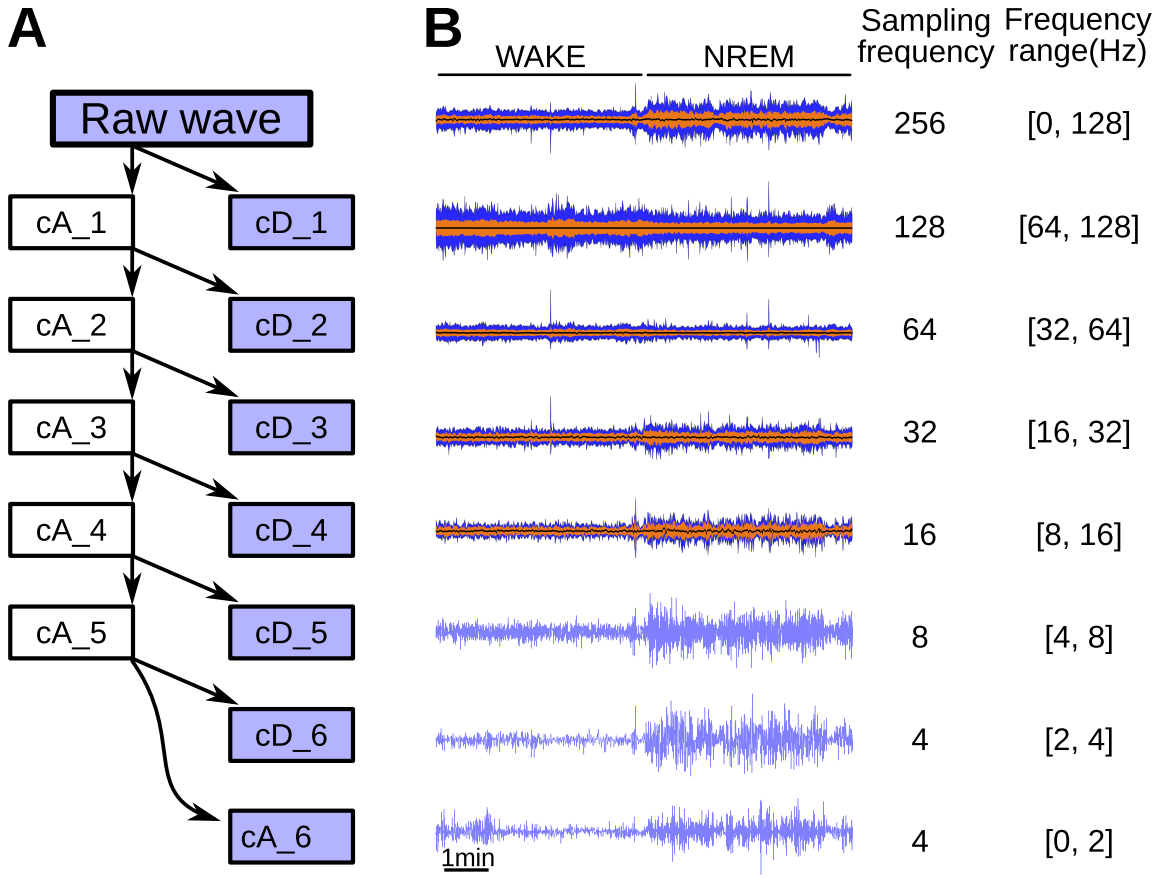

Figure 2: *Discrete wavelet decomposition.* **A**, Through discrete wavelet transform the initial signal is split into a pair of coefficients: cA and cD, which capture the low ($[0, f_s/2]$), and high $[f_s/2, f_s]$ frequency information, respectively. Then, discrete wavelet transform is applied iteratively on subsequent cA coefficients ($cA\_1, ..., cA\_n$), thus generating ($cD\_1, ..., cD\_n$). In this example, decomposition is performed up to the sixth level ($n = 6$). **B**, Ten minutes representing the EEG of a transition between an awake episode and a slow wave sleep(NREM) are shown for the raw wave and for each of the coefficients that are kept for feature extraction (blue rectangles in A). The variation of amplitude in each coefficient corresponds to a concomitant variation of amplitude in its frequency range. In this example, the general increase in power in the raw wave corresponds to an increase of power in the frequency bands below 32Hz, but with a decrease in power above 64Hz. In this figure, the signals with $f_s > 8$ (first five rows) are too dense to display every point. Instead, only local range(blue), inter-quantile range (orange) and median (black line) are represented.

Discrete wavelet decomposition was applied *a priori* to the whole signals in order to avoid edge effects (see [23] for a simple application of discrete wavelet on digital signals). The maximum decomposition levels for the EEG and EMG signals were six and four, respectively. Daubechies wavelet, with four vanishing moments ("db4") was used. As a result, a total of 14 time series was generated for each recording.

For all time series, a list of 16 features (see table 1) was computed in each subsequent non-overlapping five seconds epochs. Thus, resulting in a total of $p = 192$ variables. For each recording ($\simeq 24h$) there were approximately 17000 epochs. Some combinations of variables and signals were however incalculable. These were therefore removed; resulting in an actual number of variables $p = 164$.

Table 1: *List of computed features.* For clarity, the 16 calculated features were classified into four functional families. For all epochs, all features were computed on all wavelet coefficients and raw (EEG and EMG) signals. Therefore, a total of $16 \times (1 + 7 + 1 + 3) = 192$ features were generated for each five seconds of recording. The mathematical detail of the algorithms is provided in the documentation of `pyrem` (see appendix).

Feature family	features
Power	`mean`, `sd`, `skewness`, `kurtosis` `median`, `min`, `max`
Hjorth[24]	`morbidity`, `complexity`
Fractal	`Petrosian`[25] and `Higuchi`[26] fractal dimensions
Entropy	`Fisher information`, `SVD entropy` and `Sample entropy`[27] with $m = 2$, and $r \in \{0.2, 1.0, 1.5\}$

Features were extracted using, `pyrem`, a `python` package developed as part of this project.

2.4 Addition of temporal features

Two strategies, "lag" and "convolution", were used to add temporal information to the variable space[17]. For lag, given a vector $\mathbf{X_t}$ of feature at time t, A new vector of feature was defined as $\mathbf{X'_t}$ as

$$\mathbf{X'_t} = \{\mathbf{X_{t-\tau}}, ..., \mathbf{X_t}, ..., \mathbf{X_{t+\tau}}\} \tag{1}$$

with $\tau \in \mathbb{N}$.

For convolution, the local average $\mathbf{W_t^n}$ of each variable over several rectangular windows of sizes n was added:

$$\mathbf{X'_t} = \{W_t^{n_1}, W_t^{n_2}, ...\} \tag{2}$$

where

$$\mathbf{W_t^n} = \frac{1}{2n+1} \sum_{i=t-n}^{t+n} X_i$$

and n is a set of odd integers representing window sizes ($e.g.\{1, 3, 7, 15\}$).

2.5 Stratified Cross Validation and sampling

Stratified cross-validation was systematically applied to generate vigilance state predictions. That is, in order to generate prediction for a given time series, classifiers were always trained with a subsample of epochs originating from all *other* time series [19].

Quantification of variable importance and accuracy (figs. 3 and 4) was performed in order not to underestimate the relevance of the minority class (REM)[28]. Class unbalancedness was accounted for by fitting predictors on balanced subsamples (750 epochs of each class per tree).

2.6 Random forests analysis

Unless specified otherwise, random forests[16]were trained with balanced samples of 1000 epochs per class. In order to select variables and define new features, forests with 50 classification trees were built. 100 trees were used otherwise. Increasing the sample size and number of trees did not seem to improve performance. The number of variables drawn for each split, $mtry$, was set to the default value for classification problems $\sqrt{p}$. Variable importance was assessed by the average (over all trees) decrease in Gini impurity induced by a given variable. All random forest analysis was implemented in `R` statistical software [29], using the package `randomForest`[30].

To produce an overall summary value of confidence, an entropy based metric c was defined as:

$$c = 1 + \frac{1}{log_2(k)} \sum v_i log_2(v_i) \tag{3}$$

where v_i is the proportion of votes for the class i, and k is the number of classes. This definition has the following properties

$$c \in [0; 1]$$

$$v_1 = v_2 = ... = v_k \rightarrow c = 0$$

$$v_i = 1 \rightarrow c = 1, \forall i$$

2.7 Performance assessments

Comparisons of runtime between `pyrem` and `PyEEG`[31] were performed by measuring the average (between 2 and 100 run, depending on the algorithm) runtime of algorithms on six normally distributed ($\mathcal{N}(\mu = 0, \sigma = 1)$) random time series (*i.e.* white noise) of size n, with n between 256×5 and 256×30. This was repeated five times for each time series. For computing Higuchi fractal dimension[26], k_{max} was set to 2^3. For both approximate entropy and sample entropy[27], the embedding dimension m was set to 2, and the distance threshold, r, to 1.5. For fisher information, the embedding dimension and the delay, τ were set to 3 and 2, respectively. Finally, spectral entropy was computed for the frequency bands bounded by $\{0, 2^0, 2^1, ..., 2^6\}$Hz, with $f_s = 256.0$. Benchmarks were generated using `CPython 2.7.8` on `GNU/Linux` operating stystem with a 3.40GHz intel i7-3770 CPU.

2.8 Statistical analysis

In order to test significance in the difference of performance between `pyrem` and `PyEEG` (table 2), linear models were fitted on the log-transformed runtime for each algorithm. t-tests on the effect of the implementation (*i.e.* whether `pyrem` or `PyEEG` was used) on the intercept at $x = 1280$, were used to compute $p - values$ at $n = 1280$. Significance levels for the differences in scaling were obtained through t-tests on the interaction between n and the implementation.

In order to assess the significance of the effect of the addition of temporal variables on the cross-validation error (fig. 4), repreated t-tests were performed. Bonferronni correction was applied.

In order to determine whether state prevalences were different between predicted and ground truth time series (fig. 5A), beta regression [32] was fitted. Then, interactions between state and method (*i.e.* prediction vs. ground truth) were tested with a z-test.

In order to determine the effect of prediction on the number of episodes of each state (fig. 5B), Poisson generalised linear mixed model, using recordings (*i.e.* animals) as a random effect, was fitted using `lme4`[33]. Then, interaction between state and method were tested with a t-test.

Finall, in order to determine the effect of prediction on the average duration of episodes of each state (fig. 5B), linear mixed model on log-transformed duration, using recordings (*i.e.* animals) as a random effect, was fitted. Then, interaction between state and method were tested with a t-test.

All statistical analysis were performed with `R` software [29].

3 Results

3.1 A new efficient `python` package

Several algorithms to extract features from univariate time series had already been implemented in the `python` package `PyEEG`[31]. Unfortunately, some of them were critically slow, and could therefore not realistically have been used in the present study. Preliminary investigation of `PyEEG` source code revealed that runtimes may be improved mainly by vectorising expressions and pre-allocating of temporary arrays. Therefore, systematic reimplementation of all algorithms in `PyEEG` was undertaken. Very significant improvement in performance were achieved for almost all functions(table 2). Critically, sample and approximate entropies[27] became usable in reasonable time. For instance, 40h of CPU time were originally requiered to compute sample entropy on a all 5 second epochs in a 24h recording. The improved implementation cut this down to 55min.

Table 2: *Performance improvements over `PyEEG`.* In order to improve performance, modifications of the algorithms implemented in `PyEEG` were carried out. This table compares how long, on average, each algorithm would take, for a random sequence of length 1280 (*i.e.* $5s$ at 256Hz). It also represents how many added points would lead to a tenfold runtime increase. For the tested range ($n \in [1280; 7680]$), all algorithms add approximately an exponential time complexity ($10^{O(n)}$, $R^2 > 0.95$, for all). Several mathematical inconsistencies were also discovered and corrected. The rightmost column (†) indicates whether the original implementation was corrected in order to match mathematical definition. Each alteration is mathematically justified in the section `pyrem.univariate` of the `pyrem` documentation (see appendix). **(-)**: indicates a worse performance of `pyrem` over `PyEEG`. Significance levels: ***, $p-value < 10^{-3}$; **, $p-value < 10^{-2}$, see Material and Methods for detail about statistical analysis.

		`PyEEG`		`pyrem`		
algorithm	function	t(ms) for $n = 1280$	n for $\times 10$ increase	t(ms) for $n = 1280$	n for $\times 10$ increase	fix†
Spectral Entropy[31]	`spectral_entropy`	0.309	11459	0.227^{***}	22133^{***}	Yes
Petrosian Fractal Dimension[25]	`pfd`	2.66	8606	2.65	8579	Yes
Fisher Information[31]	`fisher_info`	3.24	8673	0.121^{***}	12427^{***}	No
Singular Value Decomposition entropy[31]	`svd_ent`	3.25	8663	0.113^{***}	11774^{**}	Yes
Hjorth parameters[24]	`hjorth`	5.14	8633	0.088^{***}	36354^{***}	Yes
Higuchi Fractal Dimension[26]	`hfd`	11.7	8833	1.39^{***}	28329^{***}	Yes
Sample Entropy[27]	`samp_ent`	8305	4276	188^{***}	$5483^{***}(-)$	No
Approximate Entropy[27]	`ap_ent`	9970	4288	487^{***}	$3478^{***}(-)$	No

Importantly, several mathematical inconsistencies between the original code and the mathemat-

ical definitions were also noticed. This affected five of the eight reimplemented functions(table 2, rightmost column). Detail of the corrections performed are provided, as notes, in the documentation of the new package (see appendix, section 2.3). Numerical results for the three functions were consistent throughout optimisation.

In order to facilitate feature extraction, several data structures and routines were also implemented in a new `python` package named `pyrem`. Briefly, extensions of `numpy` arrays[34] providing meta-data, sampling frequency, and other attributes were used to represent time series. User friendly indexing with string representing time was also developed. In addition, a container for time series of discrete annotation levels, each linked to a confidence level, was built. Importantly, a container for multiple time series, which supports heterogeneous (between time series) sampling frequencies was implemented. The new package also provides visualization, input/output, and wrappers for resampling and discrete wavelet decomposition. Finally, unittests were implemented to ensure persistence of mathematical and programmatic validity throughout developmental stage. A complete documentation of `pyrem` is provided as an appendix to the report herein.

3.2 Twenty variables can generate accurate predictions

Including temporal information (see next section) wil result in multiplication of the number of variable, rendering computation difficult, and prediction potentially less accurate. Therefore, recursive feature elimination[35] based on Gini variable importance was undertaken. Starting with all 164 variables, random forests were trained, and the number of features was reduced by a factor 1.5 by eliminating the least important variables. For each iteration, the stratified cross-validation error (see material and methods) was computed (fig. 3). Five replicates were performed.

The predictive accuracy globally increases with the number of variables. Interestingly, using only the two most important variable already results in less that 20% average error. However, this increase is very moderate for ($p > 9$) this indicates that dimensionality can be considerably reduced without largely impacting accuracy. For further investigation, $p = 21$ was considered

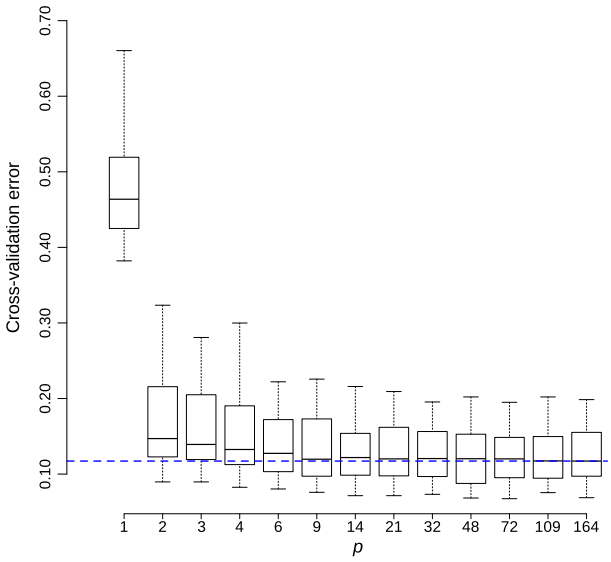

Figure 3: *Recursive variable elimination.* Random forests were recursively fitted and, at each step, only the most important $p/1.5$ variables were kept. Stratified cross-validation (see Material and Methods) was computed at each iteration. This procedure was replicated five time. The dashed blue line ($y = 0.12$) indicates the median cross-validation error when all variable are used.

to be a good compromise between error and computational load. Their relative importances are listed in table 3.

Table 3: *Relative variable importance of the 21 selected features.* Random forest algorithm can produce a value to quantify variable importance. Variable importance corresponds to how much, statistically, a variable contributes to reducing the prediction inacuracy (or, to be more precise, the Gini impurity). Starting from 164 variables, the least important variables were recursively eliminated. This table represents the 21 most important remaining features. Features from both EEG and EMG are important for accurate prediction.

Signal	Wavelet coefficient	Feature family	feature	importance
EMG	cD_1	power	median	222.2
EEG	cD_6	power	mean	161.5
EMG	cD_1	power	sd	144.7
EMG	cD_1	power	mean	136.9
EMG	cD_2	power	median	131.9
EEG	cD_5	hjorth	morbidity	128.3
EEG	raw	entropy	sample entropy ($r = 1.5$)	109.5
EEG	cD_1	power	median	101.5
EMG	cD_1	power	min	81.3
EEG	raw	entropy	sample entropy ($r = 1.0$)	79.3
EEG	raw	entropy	sample entropy ($r = 0.2$)	76.6
EEG	cD_5	hjorth	complexity	72.3
EEG	cD_6	power	sd	71.2
EMG	cD_3	power	mean	62.5
EMG	cD_2	power	mean	61.9
EEG	cD_6	power	median	61.8
EEG	cD_6	power	min	57.3
EMG	cD_4	power	mean	57.0
EEG	raw	hjorth	morbidity	54.9
EMG	cD_3	power	sd	54.4
EEG	raw	fractal	hfd	43.1

3.3 Including temporal information improves prediction

Manual scorer usually use contextual information in order to make a decision concerning a given state. For instance, implicit assumptions are made on the temporal consistency of the states. Therefore, it seems important to account for contextual information when building a predictor. In this study, two different approaches were pursued. Either the local means of features over different ranges were added to the features of each epochs (eq. 2), or the features of neighbouring time points were included (eq. 1).

A comparison of both approaches is proposed in figure 4. Significant improvement was achieved by both methods by including even little temporal information ($\tau = 1$, $n = \{1, 3\}$, $p - value < 5.10^{-3}$). Although both methods improve performance, it appears that the inter-recording variablility is systematically lower using convolution method. For instance, the standard variation in error between all recordings is always lower than 0.03 for convolutions of window size greater than two, whilst it is systematicaly above 0.031 for all the values of τ. The most significant difference with the original set of variables was achieved using $n = \{1, 3, 7, 15, 31\}$ ($p-value < 10^{-6}$). Therefore, this new set of 105 variables was used to train the final predictor. Combining both convolution and lag approaches did not improve prediction any further (data not shown).

3.4 Structural differences with ground truth

After selecting important variables and defining new ones, general and structural performance of the classifier was assessed. For this purpose, random forests were trained on samples accounting for the unbalanced prevalence. Predictions were generated using the same stratified sampling approach. That is, predictions on each time series were made by a classifier for which the training set did not contain any point from this time series.

The global confusion matrix is presented in table 4. The overall accuracy was 0.92. For NREM and wake, both specificity and positive predictive value were above 0.92. However for REM epochs, the specificity is only 0.74, and the false detection rate is 15%.

In order to investigate the structural differences between ground truth and the predicted states,

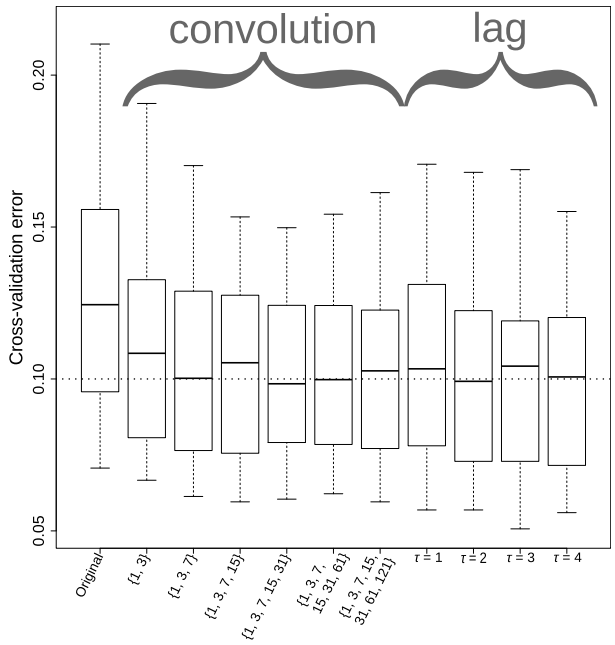

Figure 4: *Integration of temporal information.* In order to improve prediction accuracy, information about the past and future features was added to the original features following two different strategies. Convolution inflates p by adding the mean features of neighbouring points over different window sizes (eq. 2). The numbers in braces represent the window sizes. Alternatively, the features of the τ previous and future neighbours was added to the variables (eq. 1). The dotted line represents a cross-validation error of 10%. All additions of temporal features reduce significantly the cross-validation error compared to the original set of features ($p - value < 5.10^{-3}$, for all, repeated t-tests). Significance threshold, $\alpha = 0.05$, after Bonferroni correction is $\alpha' = 0.05/10 = 5.10^{-3}$.

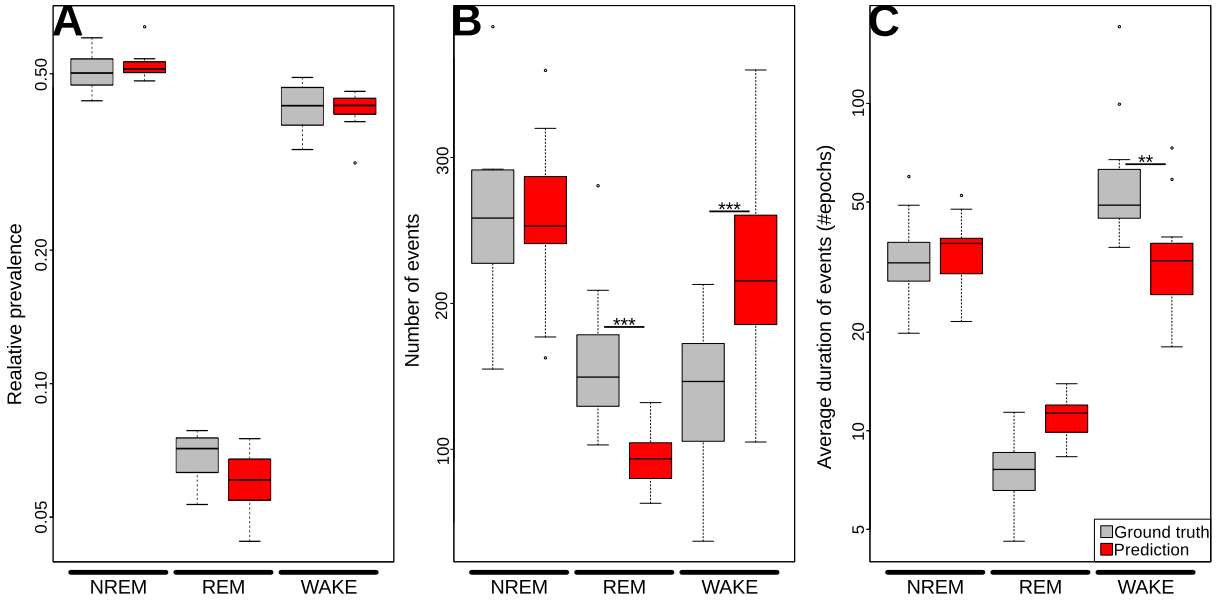

Figure 5: *Structural differences.* Three metrics describing structure of sleep were computed for both ground truth (grey) and predicted (red) time series. **A**, No significant difference in state prevalence was found. **B**, The number of events was significantly over-estimated by the classifier for wake state and under-estimated for REM state ($p - value < 10^{-15}$ for both). **C**, The average duration of wake and REM episodes were under-estimated ($p - value < 10^{-2}$) and marginally over-estimated ($p - value = 0.057$), respectively. Log scale was used for the response variables in A and C. $n = 12$ per combination of factors. Statistical methods are detailed in the Material and Methods section. Significance levels: ***, $p - value < 10^{-3}$; **, $p - value < 10^{-2}$.

Table 4: *Relative confusion matrix.* Agreement between reference scoring (rows) and proposed classifier (column). For clarity, only the percentage of all epochs is shown. The total number of epochs was 202171. *a*, PPV (Positive Predictive Value = 1 - False Detection Rate); *b*, Overall Accuracy.

		Prediction			PPV[a]
		NREM	REM	WAKE	
Ground truth	NREM	47.9	1.36	3.03	0.92
	REM	0.63	5.14	0.25	0.85
	WAKE	2.24	0.42	39.3	0.94
Sensitivity		0.94	0.74	0.92	0.92[b]

three metrics describing physiological properties of sleep were computed (fig. 5).

Prevalence(fig. 5A) of states is a widely used metric to describe sleep patterns. No significant difference was found between the ground truth and predicted prevalences ($p - value > 0.13$ for all, z test on the interaction terms of β regression).

Sleep fragmentation is often assessed by a combination of two variables: number of episodes of a given state and average duration of all episodes per state[36]. The effect of prediction on both variables were therefore studied (fig. 5B and C, respectively).

Large differences were observed between the number of REM sleep and awake episodes between the ground truth and the predicted data ($p - value < 10^{-15}$ for both, t test on the interaction terms of Poisson GLMM).

In addition, significant structural differences were found between methods for the average duration of awake ($p - value < 10^{-3}$) episodes (t test on the interaction terms of linear mixed model).

3.5 Attribution of confidence score to predictions

Since classification may be inaccurate, it would be interesting to associate 'confidence' score to each prediction. A entropy based confidence metric c (eq. 3) was defined for this purpose. In order to validate it, the average cross-validation error was computed for different ranges of confidence (fig6A).

As expected, the probability of misclassification decreases monotonically with c. In addition, error rate seem to tend to zero when the confidence value is one, and, for confidences close to

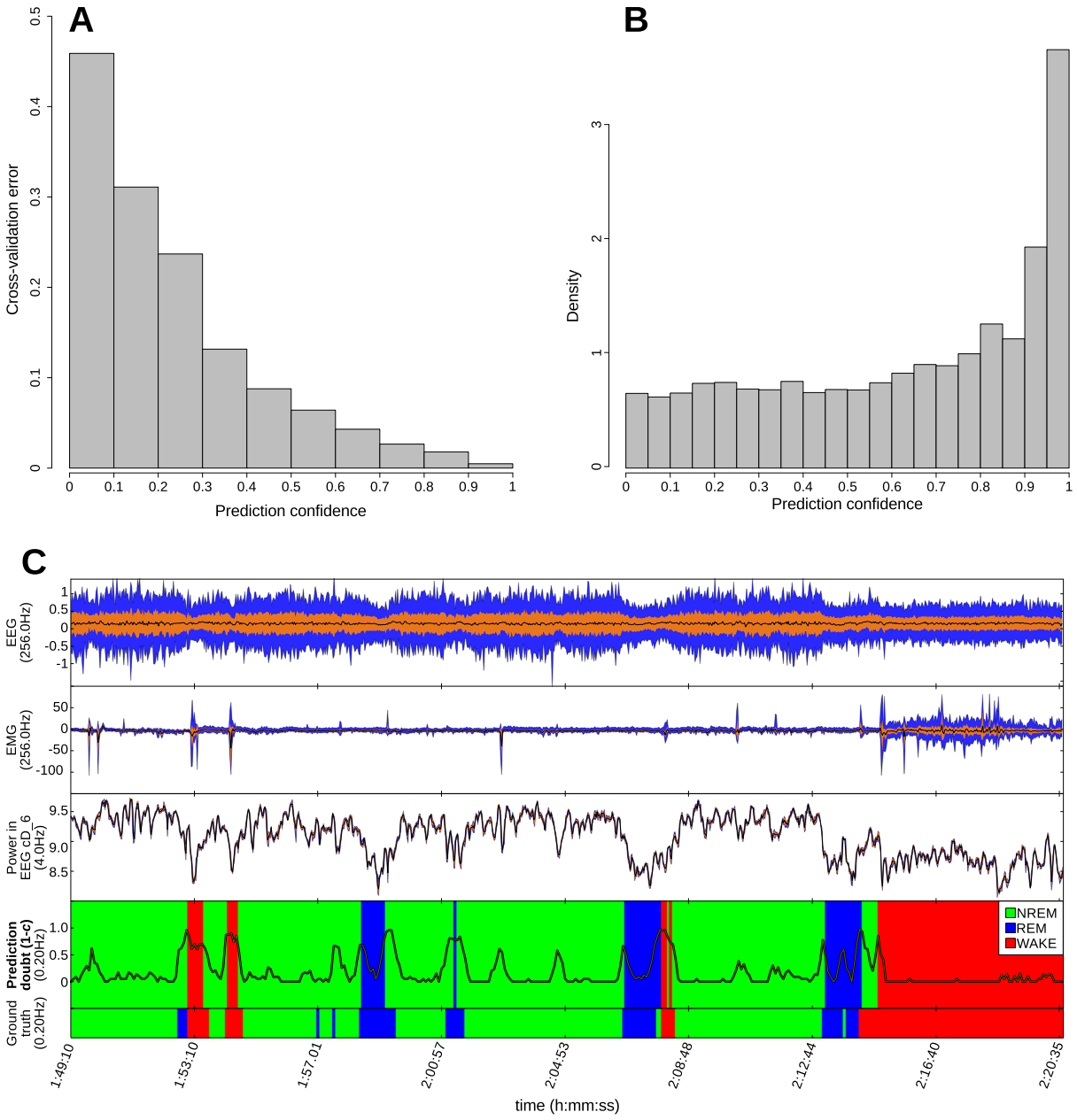

Figure 6: *A posteriori confidence assessment.* **A**, Relation between the confidence value derived from proportions of votes (eq.3) and actual proportion of error. Cross-validation error decreases with empirical confidence value. For low values of confidence, $[0, 0.1]$, the predictor is not very reliable (more that 45% error, whislt chance would be 67%). In contrast, within the highest confidence range, $(0.9, 1.0]$, misclassification are very rare($< 0.5\%$). **B**, Distribution of confidence values for all epochs. The overall median confidence value is at 0.69. **C**, Visualisation of approximately 30 minutes of representative EEG and EMG recording. The variation in the most important EEG variable, mean power in the cD_6 sub-band, is also plotted. Both predicted and ground truth states are colour coded (red: awake, blue: REM, green: NREM). The doubt level $(1 - c)$ associated with predictions is displayed on top of the prediction annotations.

zero, the predictor is very inaccurate. These characteristics indicate that c can be used to as a supporting value for predictions. One application of such a confidence level could be to provide user with an overall quality assessment. In addition, it makes it possible to display confidence

while visually inspecting a recording (fig6C) in order to facilitate resolution of ambiguities.

4 Discussion

4.1 Software package for feature computation

In order to train statistical learning methods to classify vigilance states, it was necessary to compute an exhaustive set of features for all consecutive five second epochs over long (24h) time series. For this purpose, `pyrem`, a new `python` package was developed based on `PyEEG`[31], which already implements several algorithms often used to study EEG. Very significant improvements in performance were achieved for almost all functions implemented in `PyEEG` (table 2). These improvements will considerably speed-up prototyping of feature extraction and may be essential in order to build real time classifiers. In addition, such modifications will make it possible to compute features for a large number of recordings in reasonable time. Further improvements are possible, for instance, sample entropy was tentatively implemented in Julia programming language[37] and performed 25 times faster than `pyrem`'s implementation[1] Interestingly, it appears that the new implementation of sample and absolute entropy does not scale as well as the original implementation. One explanation could be that memory becomes limiting when using vectorised expressions, because large temporary arrays are created. Nevertheless, realistically, neither algorithms would be used for long time series.

Several `PyEEG` functions were also found to be inconsistent with mathematical definitions (see `pyrem` documentation, appendix). This unfortunatly apperas to be a common issue for academic software. The general status of the peer-review process and the reproducibility of programs and algorithms have recently drawn attention (see [38, 39] for discussions about this issue).

4.2 Exhaustive feature extraction

Feature extraction in the present study contrasts with previous work in two respects. First of all, features were exhaustively computed not only on raw signals, but also on all wavelet frequency sub-bands. Then, new variables were created to account for temporal consistency of vigilance state episodes.

[1] Implementation available at https://github.com/qgeissmann/Physiology.jl/blob/master/src/univariate.jl.

Discrete wavelet decomposition is an extremely fast an accurate algorithm to filter a periodic signal into complementary and exclusive frequency sub-bands (fig. 2). Şen et al.[7] obtained very promising results by computing a large number of features on the raw EEG signal and a limited subset of features (*i.e.* mean power and absolute values) in some wavelet coefficients. In contrast, in the present study, all features were computed on all frequency sub-bands. Interestingly, some of the features that are the most important for prediction would not have been discovered otherwise (see table 3).

Many authors have modelled time series of epochs as if each epoch was statistically independent from each other. This assumption makes it straightforward to use classical machine learning techniques such as ANNs, SVMs[12], random forests[16] and others. They have the advantage coping very well with non-linearity, can handle a large number of predictors and have many optimised implementations.

However, working with this assumption generally does not allow to account for temporal consistency of vigilance states. Indeed, prior knowledge of, for instance, the state transition probabilities cannot be modelled. Manual scorers use contextual information to make decisions. For example, if a given epoch has ambiguous features between REM and awake, it is likely to be classified as awake given surrounding epochs are, less ambiguously, awake. For this reason, explicit temporal modelling, using, for instance, Hidden Markov Models has been investigated[14, 15].

In order to benefit from the classical machine learning framework whist including temporal information, it is possible to create, new variables, accounting for the temporal variation[17]. This study demonstrated that addition of temporal context significantly improved predictive accuracy (fig.4). The convolution approach (eq.2) appeared to provide better results. Instead of averaging feature after calculation, it may be advantageous to compute features over epochs of different length in a first place. Thus, the accuracy of local of non additive features, such as median, will be improved. In addition to local mean of feature, other variables, such as local slope and local variance of each feature may improve classification[18].

Although addition of time-dependent variables improved accuracy over a time-unaware model, their use can be seen as controversial. Indeed, including prior information about sleep structure

will cause problems if the aim is to find differences in sleep structure. As an example, let us consider a training set only made of healthy adult wild type animals, and let us assume that NREM episodes are always at least, 5min long. Implicitly, this information becomes a prior. That is, the implicit definition of NREM is that it is uninterrupted. The same classifier is not expected to perform well if used on an animal which, for instance, show frequent interruption of NREM sleep by short awake episodes. Indeed, a 'time-aware' model will need much more evidence to classify correctly a very short waking episode inside sleep (because this never occurred in the training set). Therefore, predictive accuracy alone should not be the ultimate end-goal. Models which can perform well without including too much temporal information ought to be preferred in so far as they are more likely to be generalisable.

4.3 Random forest classification

In this study, random forest classifiers[16] were exclusively used. In addition to their capacity to model non-linearity, they are very efficient at handling very large number of variables. Recently very promising classification of sleep stages in human were generated using this algorithm[7]. A very interesting feature of random forest is their natural ability to generate relative values of importance for the different predictors. These values quantifies how much each variables contributes to the predictive power of the model. This feature is extremely useful because it allows using random forests for variable selection. This can be used to reduce dimensionality of the variable space without losing predictive power (fig.3), but also to study conditional variable importance[40], or, for instance, determine which variables are important to segregate pairs of classes. Whilst random forests are not guaranteed to be the best predictor, they allow fast and in-depth preliminary investigation. Finally, underlying mechanisms of random forest (*i.e.* how variables are combined) is relatively simple to understand in terms of binary logic. This "white box" property is an advantage when trying to provide more rationality to a subjective and implicit human decision.

4.4 Rigorous and comprehensive model evaluation

Previous research, using classical statistical learning framework, have often assessed their classifier through cross-validation. It however often unclear how sampling was performed to generate training and testing sets[7, 41, 42]. Time series of epochs are dense and, in general, the features (and labels) at a given time are very correlated with surrounding features. Therefore, if random sampling of even 50% of all epochs, from all time series, was performed, most points in the training set will have a direct neighbour in the testing set. This almost corresponds to an artificial duplication of a dataset before cross-validation and is likely to fail to detect overfitting. In the preliminary steps of this study, it was observed that almost perfect accuracy could be achieved when performing naive cross-validation (data not shown). Supporting further this idea, such surprisingly high accuracy was not observed when training the model with all the even hours (from start of the experiment) and testing it with all the odd ones. There are several way to reduce overfitting including limiting the maximal number of splits when growing classification trees, or pruning trees. However, it never possible to unsure a model will not overfit *a priori*. Thus it remain necessary to assess the model fairly. In this study, systematic stratified cross-validation was performed [19]. As a result, all predictions made on any 24h time series are generated by models that did not use any point originating from this same time series. This precaution simulate the the behaviour of the predictor with new recordings. Cross-validation was not only used to generate overall value of accuracy, but also, to further assess differences in sleep patterns (fig. 5).

4.5 Quality of the raw data

Vigilance states can be viewed as discrete representation of a phenomena that is, in fact, continuous. In this case, the borders between different states are, by nature, fuzzy and somewhat arbitrary. Therefore, ground truth data cannot be assumed to be be entirely correctly labelled. In particular, transitions between states will be intricately inaccurate. The assessment of prediction doubt (fig. 6, fourth row) illustrate the high uncertainty inherent to transitions.

The ground truth labels used in this study has been generated by a two pass semi-automatic

method. In a first place, an automatic annotation is performed based on a human-defined variable threshold. Then, the expert visually inspect the result and correct ambiguities. The first pass was originally designed to combine, through logical rules, four epochs of five seconds to produce 20s epochs[6]. However, it was simplified in-house in order to produce only five second epochs, ignoring the last step, and has since not been reassessed against manual scoring. It is expected that this simplification increased divergence with manual scorers.

Several studies have used ground-truth data that was manually scored independently by several experts, which often appear to show good mutual agreement. This seem extremely important for several reasons. First of all, it permits to compare inter-human error to the automatic classifier error. Then, it allow to allocate a value of confidence to each annotation. For instance, if, for a given epoch, there is strong disagreement between experts, the confidence will be low. When training a model, this uncertainty can be included, for instance, as a weight.

4.6 Overall results

The predictions of the classifier presented in this research agreed with ground truth for 92% of epochs (table 4). Although the limitation of the ground truth annotation makes it is difficult to put this result into perspective, this score is very promising. In addition, prediction did not result in significant difference in prevalences. However, there were, on average, much less REM episodes in the predicted time series. The duration of REM episodes was also over-estimated by prediction (though this is only marginally significant). Altogether, this indicates that REM state is less fragmented in the predicted data. In contrast, the awake state was more fragmented in the predicted time series. Although statistically significant, these differences in variables characterising sleep structure are never greater than twofold.

It would be very interesting to investigate further the extent to which such classifier could be used to detect alteration in the structure of sleep. One way could be analyse the sleep structure of two groups of animals for which differences were already found, and quantify how much more, or less, difference is found using automatic scoring.

Conclusion

The aim of the study herein was to build a classifier that could accurately predict vigilance states from EEG and EMG data. In a first place, `pyrem`, a new python package was designed to efficiently extract a large number of features from electrophysiological recordings. Then, a random forest approach was used to eliminate irrelevant variables. Importantly, this study shows that prediction accuracy can then be improved by including features derived from restricted local averages. The overall achieved accuracy was as high as 92%, and although some significant structural differences were induced by prediction, the classifier was overall satisfying. In addition, the presented classifier can generate confidence values that can be used to moderate each prediction, and ultimately decide whether to trust them. Before considering implementation of this promising classifier is a ubiquitous software tool, it would be necessary to generalise its results by the inclusion of different sources of data.

Availability

The source code of `pyrem` is available at https://github.com/gilestrolab/pyrem and the package will be released shortly, as an open-source software, in the official python repositories.

References

[1] J. M. Siegel, "Do all animals sleep?," *Trends in Neurosciences*, vol. 31, pp. 208–213, Apr. 2008.

[2] C. Cirelli and G. Tononi, "Is sleep essential?," *PLoS Biol*, vol. 6, p. e216, Aug. 2008.

[3] A. L. Loomis, E. N. Harvey, and G. A. Hobart, "Distribution of disturbance-patterns in the human electroencephalogram with special reference to sleep.," *Journal of Neurophysiology*, 1938.

[4] E. Aserinsky and N. Kleitman, "Regularly occurring periods of eye motility, and concomitant phenomena, during sleep," *Science*, vol. 118, no. 3062, pp. 273–274, 1953.

[5] L. A. Toth and P. Bhargava, "Animal models of sleep disorders," *Comparative Medicine*, vol. 63, pp. 91–104, Apr. 2013.

[6] D. Costa-Miserachs, I. Portell-Corts, M. Torras-Garcia, and I. Morgado-Bernal, "Automated sleep staging in rat with a standard spreadsheet," *Journal of Neuroscience Methods*, vol. 130, pp. 93–101, Nov. 2003.

[7] B. en, M. Peker, A. avuolu, and F. V. elebi, "A comparative study on classification of sleep stage based on EEG signals using feature selection and classification algorithms," *Journal of Medical Systems*, vol. 38, pp. 1–21, Mar. 2014.

[8] G. Chouvet, P. Odet, J.-L. Valatx, and J.-F. Pujol, "An automatic sleep classifier for laboratory rodents," *Waking & Sleeping*, vol. 4, no. 1, pp. 9–31, 1980.

[9] W. Haustein, J. Pilcher, J. Klink, and H. Schulz, "Automatic analysis overcomes limitations of sleep stage scoring," *Electroencephalography and Clinical Neurophysiology*, vol. 64, pp. 364–374, Oct. 1986.

[10] M. Lngkvist, L. Karlsson, and A. Loutfi, "Sleep stage classification using unsupervised feature learning," *Advances in Artificial Neural Systems*, vol. 2012, p. 5, 2012.

[11] G. A. Sunagawa, H. Si, S. Shimba, Y. Urade, and H. R. Ueda, "FASTER: an unsupervised fully automated sleep staging method for mice," *Genes to Cells*, vol. 18, pp. 502–518, June 2013.

[12] S. Crisler, M. J. Morrissey, A. M. Anch, and D. W. Barnett, "Sleep-stage scoring in the rat using a support vector machine," *Journal of Neuroscience Methods*, vol. 168, pp. 524–534, Mar. 2008.

[13] E. M. Ventouras, N.-T. Economou, I. Kritikou, H. Tsekou, T. J. Paparrigopoulos, and P. Y. Ktonas, "Performance evaluation of an artificial neural network automatic spindle detection system," *Conference proceedings: ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, vol. 2012, pp. 4328–4331, 2012.

[14] L. G. Doroshenkov, V. A. Konyshev, and S. V. Selishchev, "Classification of human sleep stages based on EEG processing using hidden markov models," *Biomedical Engineering*, vol. 41, pp. 25–28, Jan. 2007.

[15] S.-T. Pan, C.-E. Kuo, J.-H. Zeng, and S.-F. Liang, "A transition-constrained discrete hidden markov model for automatic sleep staging," *BioMedical Engineering OnLine*, vol. 11, p. 52, Aug. 2012.

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, Oct. 2001.

[17] T. G. Dietterich, "Machine learning for sequential data: A review," in *Structural, Syntactic, and Statistical Pattern Recognition* (T. Caelli, A. Amin, R. P. W. Duin, D. d. Ridder, and M. Kamel, eds.), no. 2396 in Lecture Notes in Computer Science, pp. 15–30, Springer Berlin Heidelberg, Jan. 2002.

[18] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, Aug. 2013.

[19] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures,"

Proc. VLDB Endow., vol. 1, pp. 1542–1552, Aug. 2008.

[20] A. L. Vyssotski, A. N. Serkov, P. M. Itskov, G. Dell'Omo, A. V. Latanov, D. P. Wolfer, and H.-P. Lipp, "Miniature neurologgers for flying pigeons: Multichannel EEG and action and field potentials in combination with GPS recording," *Journal of Neurophysiology*, vol. 95, pp. 1263–1273, Feb. 2006.

[21] W. Putnam and J. Smith, "Design of fractional delay filters using convex optimization," in *Applications of Signal Processing to Audio and Acoustics, 1997. 1997 IEEE ASSP Workshop on*, pp. 4–pp, IEEE, 1997.

[22] V. V. Vyazovskiy and A. Delogu, "NREM and REM sleep: Complementary roles in recovery after wakefulness," *The Neuroscientist*, vol. 20, pp. 203–219, June 2014.

[23] S. Prabhakar, A. R. Mohanty, and A. S. Sekhar, "Application of discrete wavelet transform for detection of ball bearing race faults," *Tribology International*, vol. 35, pp. 793–800, Dec. 2002.

[24] B. Hjorth, "EEG analysis based on time domain properties," *Electroencephalography and Clinical Neurophysiology*, vol. 29, pp. 306–310, Sept. 1970.

[25] A. Petrosian, "Kolmogorov complexity of finite sequences and recognition of different preictal EEG patterns," in *, Proceedings of the Eighth IEEE Symposium on Computer-Based Medical Systems, 1995*, pp. 212–217, June 1995.

[26] T. Higuchi, "Approach to an irregular time series on the basis of the fractal theory," *Physica D: Nonlinear Phenomena*, vol. 31, pp. 277–283, June 1988.

[27] J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," *American Journal of Physiology - Heart and Circulatory Physiology*, vol. 278, pp. H2039–H2049, June 2000.

[28] A.-L. Boulesteix, S. Janitza, J. Kruppa, and I. R. Knig, "Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinfor-

matics," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 6, pp. 493–507, 2012.

[29] R Core Team, *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2014.

[30] A. Liaw and M. Wiener, "Classification and regression by randomForest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[31] F. S. Bao, X. Liu, and C. Zhang, "PyEEG: An open source python module for EEG/MEG feature extraction," *Computational Intelligence and Neuroscience*, vol. 2011, p. e406391, Mar. 2011.

[32] F. Cribari-Neto and A. Zeileis, "Beta regression in r," 2009.

[33] D. Bates, M. Maechler, B. Bolker, and S. Walker, *lme4: Linear mixed-effects models using Eigen and S4*. 2014. R package version 1.1-6.

[34] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, pp. 22–30, Mar. 2011.

[35] B. H. Menze, M. B. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht, "A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data," *BMC Bioinformatics*, vol. 10, pp. 1–16, Dec. 2009.

[36] D. S. J. Pang, C. J. Robledo, D. R. Carr, T. C. Gent, A. L. Vyssotski, A. Caley, A. Y. Zecharia, W. Wisden, S. G. Brickley, and N. P. Franks, "An unexpected role for TASK-3 potassium channels in network oscillations with implications for sleep mechanisms and anesthetic action," *Proceedings of the National Academy of Sciences*, vol. 106, pp. 17546–17551, Oct. 2009.

[37] J. Bezanson, S. Karpinski, V. B. Shah, and A. Edelman, "Julia: A fast dynamic language for technical computing," *arXiv:1209.5145 [cs]*, Sept. 2012. arXiv: 1209.5145.

[38] A. Morin, J. Urban, P. D. Adams, I. Foster, A. Sali, D. Baker, and P. Sliz, "Shining light into black boxes," *Science*, vol. 336, pp. 159–160, Apr. 2012. Cited by 0031.

[39] T. Crick, B. A. Hall, and S. Ishtiaq, ""can i implement your algorithm?": A model for reproducible research software," *arXiv:1407.5981 [cs]*, July 2014. arXiv: 1407.5981.

[40] C. Strobl, A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis, "Conditional variable importance for random forests," *BMC Bioinformatics*, vol. 9, p. 307, July 2008.

[41] F. Ebrahimi, M. Mikaeili, E. Estrada, and H. Nazeran, "Automatic sleep stage classification based on EEG signals by using neural networks and wavelet packet coefficients," in *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2008. EMBS 2008*, pp. 1151–1154, Aug. 2008.

[42] F. Chapotot and G. Becq, "Automated sleepwake staging combining robust feature extraction, artificial neural network classification, and flexible decision rules," *International Journal of Adaptive Control and Signal Processing*, vol. 24, pp. 409–423, May 2010.

Appendices

A `pyrem` documentation