HHS Public Access

Author manuscript

Comput Biol Med. Author manuscript; available in PMC 2018 October 01.

Published in final edited form as:

Comput Biol Med. 2017 October 01; 89: 441–453. doi:10.1016/j.compbiomed.2017.08.030.

Identifying Sleep Spindles with Multichannel EEG and Classification Optimization

Ning Mei[1], **Michael Grossberg**[2], **Kenneth Ng**[1], **Karen T. Navarro**[1], and **Timothy M. Ellmore**[1,*]

[1]Department of Psychology, The City College of the City University of New York

[2]Department of Computer Science, The City College of the City University of New York

Abstract

Researchers classify critical neural events during sleep called spindles that are related to memory consolidation using the method of scalp electroencephalography (EEG). Manual classification is time consuming and is susceptible to low inter-rater agreement. This could be improved using an automated approach. This study presents an optimized filter based and thresholding (FBT) model to set up a baseline for comparison to evaluate machine learning models using naïve features, such as raw signals, peak frequency, and dominant power. The FBT model allows us to formally define sleep spindles using signal processing but may miss examples most human scorers would agree are spindles. Machine learning methods in theory should be able to approach performance of human raters but they require a large quantity of scored data, proper feature representation, intensive feature engineering, and model selection. We evaluate both the FBT model and machine learning models with naïve features. We show that the machine learning models derived from the FBT model improve classification performance. An automated approach designed for the current data was applied to the DREAMS dataset [1]. With one of the expert's annotation as a gold standard, our pipeline yields an excellent sensitivity that is close to a second expert's scores and with the advantage that it can classify spindles based on multiple channels if more channels are available. More importantly, our pipeline could be modified as a guide to aid manual annotation of sleep spindles based on multiple channels quickly (6–10 seconds for processing a 40-minute EEG recording), making spindle detection faster and more objective.

Keywords

thresholding; machine learning; optimization; sleep spindle; memory consolidation

*Corresponding Author: Timothy M. Ellmore, Ph.D., The City College of New York, 160 Convent Avenue, Department of Psychology, NAC 7/120, New York, NY 10031, Phone: 212 650-5714, Fax: 212 650-5659, tellmore@ccny.cuny.edu.

Conflict of Interest Statement

None declared.

Introduction

Background

The functional role of sleep in mammals remains a matter of debate [2–6]. One of the theories is that occurrences of particular neural events during sleep reflect the processes associated with memory consolidation [7, 8]. It has become a challenge to identify these neural events by simply viewing the data because it is time consuming and prone to different interpretations by different viewers, especially in high definition neural recordings, which contain thousands of data points in just a few seconds of data. One of the neural recording techniques is to record and monitor using scalp electroencephalography (EEG).

Macro and micro structures are typically found in segmented EEG recordings. Macro-structured neural events refer to segments that are usually 20 to 30 seconds long and represent different sleep stages, or levels of sleep compared to the awake condition [9–11]. On the other hand, micro-structured neural events refer to local and short segments, such as sleep spindles. Sleep spindles typically occur during sleep stage 2, and they are believed to be generated from the thalamus [6, 12]. Based on the dominant frequency of a segment around a spindle, each is classified as a slow spindle (9–10 Hz [1314]; 10 – 12 Hz [13, 15, 16]) or a fast spindle (13 – 15 Hz [15, 17]; 12 – 14 Hz [16]), which are believed to occur during different phases of slow oscillations (< 1 Hz) [18]. Measuring sleep spindles and analyzing their relationship to behavior and cognition may provide insight into how these neural events influence memory performance, as well as provide diagnostic measurements for various sleep disorders. It is not completely understood how the brain integrates past information to generate new memories. Thus, the recording of sleep spindles provides common and quantifiable measurements of sleep so that we can connect sleep spindles with memory and describe how the brain processes information during sleep.

Related Work

Given that identifying these neural events may provide a powerful tool to study the relationship between sleep and memory, it is critical that we have a standard to define these events [45, 46]. Unfortunately, the definition of spindle features varies across studies. This complicates the aim to classify spindles automatically [19–23]. Studies that incorporate machine learning algorithms to classify these neural events usually make classifications based on a single EEG channel (i.e. Cz) and long period (> 7 hours) of recordings. Among all the automated algorithms, filtering based and thresholding (FBT) approaches have shown some promising results for classifying sleep stages, spindles, and k-complexes [24, 25]. Methods like template-based filtering and continuous wavelet transforms (CWTs) [12], Support Vector Machine classifiers (SVM) [26], decision-tree classifiers [27], and artificial neural networks (ANN) [28] have also been investigated. However, there are few studies that classify spindles and other neural events (k-complexes) simultaneously using a unified framework [29–33]. Visually, a typical spindle (11–16 Hz) has a unique symmetrical shape along the temporal axis, looking like a football, while other neural events, such as k-complexes, usually have an asymmetrical shape. This difference limits regular approaches that rely on an explicitly characterizing both events using signal processing. While distinguishing different patterns is easy, it remains a challenge to recognize the different

patterns with the same system. The current study focused mainly on classifying sleep spindles and investigates how the current results might guide us to classify many other neural events, such as sleep stages and k-complexes.

Motivation

The studies mentioned above provide evidence that sophisticated machine learning algorithms perform better in classifying sleep spindles against “non-spindles” in segments of neural recordings. However, it is not useful for small datasets or practical use. First, machine learning algorithms, especially multi-layer neural networks, usually require a large amount of data (> 1000 training samples) and features besides raw signals are extracted to improve classification performance, but neither is a common approach in many clinical evaluations of sleep. As we mentioned above, features for defining a typical spindle varies from study to study, and features in time-frequency space are usually extracted to add to the feature list so that traditional single layer machine learning models learn better about the patterns between spindles and non spindles. Second, machine learning models take a structured segment of the data (namely an epoch) and return probabilities of whether this segment of data contains spindles or not. Applying a machine learning model is challenging for practical use in localizing spindles for several reasons.

The first reason is that it is difficult to localize and recognize neural events. Neural events like spindles can occur at any moment of a recording with varying duration (~ 0.5 – 2 seconds), and this makes it difficult to define a segmenting window to sample representative training data for machine learning models [36]. It is difficult to construct sampling windows to sample segments containing a full cycle of spindle. With small windows, we might capture part of a spindle, while with large windows we might capture too much irrelevant signal around the spindle. To localize spindles, a model must take the varying duration of spindles into account and return the locations (time) and the durations (length) of the spindles. Such a goal could be achieved by using flexible kernels within a machine learning model and is usually easier to address in recurrent neural networks with long-short term memory (LSTM) neurons [34].

The second reason is that the sample size of spindles could be small while the sample size of non-spindles could be large, which are not optimal for machine learning models [6]. Researchers usually only are able to sample about 50 to 200 spindles in a 30-minute short nap [7], regardless of the duration of the spindles. The total sample size of spindles is usually a small fraction of the total recording. However, a sufficient machine learning training takes more than 1000 training samples, and a LSTM neural network would take more than 5000 training samples. It is common in short nap periods to sample imbalanced sample sizes of spindles and non-spindles (e.g., 5 spindles and 95 non-spindles). Assuming a machine learning model is flexible to take the varying duration of spindle and non-spindle samples into account, the model could report at least 95% accuracy by claiming all the samples are “non-spindles”, but this is not we want to see in practical settings. Therefore, the imbalanced sample size of the spindle and non-spindle classes makes preparing training data a difficult problem in sleep studies [35].

The third reason is that the classification of a spindle is usually made based on signals of a single channel. Classifying spindles based on a single channel could misrepresent the global characteristics of spindles, which we might capture better through a multi-channel approach. Studies have shown that spindles can be recorded across multiple channels [7]. The FBT approach includes particular spindle and non-spindle samples based on global signal patterns across multiple channels (n>2). Making classifications based on multiple channels, we could identify spindles that consistently occur in several regions of the brain.

Objectives

The objectives of the current study include implementing the FBT approaches [19] to classify spindles using a short period of high definition EEG recordings. The FBT approach is designed to classify spindles quickly with flexible parameters that capture temporal and spectral variations of the EEG representations of spindles (e.g., frequency, duration, amplitude, etc.) and serves as a classification bench mark for further investigation of machine learning models. Furthermore, this study aims to optimize feature parameters that are used to speed up and aid the sampling of enough data to train a more accurate fully automated process. With enough training data, we hope to eventually define spindles probabilistically by intuitive features. Thus, we applied our algorithms to the publically available DREAMS project data [1] with few human inputs to classify spindles based on a single EEG channel. An additional objective is to present a state of the art multi-channel FBT approach which encodes current characterization of spindles with a flexible range of features. To make a fair comparison among the FBT approach, machine learning models, and experts' scores, a cross-validation criterion that is described in the Ray et al study [37] is used. We applied this algorithm to our data with six channels of interest to examine how machine learning models perform better than the FBT approach.

Novelty and Outline of the Present Study

There are two novel aspects in the current study. The first is that we propose a nested model of FBT and machine learning, using a fast processing FBT model to guide machine learning in model selection. The second is that we propose a nested machine learning model derived from the FBT model that can perform well by using simple signal information, such as raw signal values, peak frequencies, and corresponding power density values sampled across multiple recording channels. The paper is structured first by outlining the preprocessing of the EEG data. Then the implementation, optimization, and cross-validation is detailed. The development of the machine learning pipeline is then detailed followed lastly by a comparison of the FBT and machine learning models.

Methods

Data Acquisition

A total of 64-channels of EEG data, including 2 EOG electrodes, were continuously recorded at 1 kHz sampling with an antiCHamp active electrode system (Brain Products, GmbH) while subjects napped on a bed inside a sound-attenuated testing booth (IAC industries). Experiments were carried out in accordance with the The Code of Ethics of the World Medical Association (Declaration of Helsinki). Each subject provided written

informed consent and completed the study procedures according to a protocol approved by the Institutional Review Board of the City College of New York. Subjects were compensated $15 per hour for participation. Before EEG recordings began, the impedance of all electrodes was optimized to be < 25 kOhms by application of electrolytic gel between the scalp and the electrodes tips. The reference electrode during recording was T9 (left mastoid) and two additional flat electrodes were placed around the eye for electrooculography (EOG) recording. Data used in the current study are freely available (https://osf.io/chav7/) and included 28 healthy subjects recorded on two separate days. Data from 25 subjects were included (6 female, mean age 25.5 +/− 7.03 standard deviation, range 18 – 42 years old). Nine recordings (Both days of subject 7, 26, 27, day 1 of subject 5 and 12, day 2 of subject 20) were excluded because they failed to reach stage 2 sleep or because their data was too noisy for processing. Thus, in total 22 subjects (41 recordings) were included in the following analyses. We focused on six electrodes of interest in applying the pipeline on data collected for the present study. These included the 10/20 international standard electrodes F3, F4, C3, C4, O1 and O2, Fig 1.). In the Schabus et al. [7] study, recording locations Fp1, Fp2, C3, C4, O1, and O2 were used to detect spindles in both hemispheres of the frontal, central, and occipital areas of the brain. Due to the high density 64 channel montage used in the present study, Fp1 and Fp2 were replaced by F3 and F4 to reduce contributions from artifacts.

Experimental Design

These data are part of a larger study examining the effect of memory load on the generation of sleep spindles and some brief details are provided here about the experimental design [49]. Subjects completed memory tasks before and after the nap EEG recordings. Over the course of two days, subjects came in at the same time each day, between the hours of 10 am to 5 pm. Each session lasted two hours for a total experimental participation time of four hours. Before each recording session, subjects were instructed to sleep an hour and a half later than their usual sleep time so that they would be tired at the start of the experiment and therefore more likely to fall asleep. On each experimental day, subjects performed either a low or high load Sternberg scene working memory task with load order randomized between the two testing sessions separated by either one day or one week. They then completed a recognition task before they took a short nap in the sound attenuated chamber during continuous EEG recording. After the nap, subjects performed another recognition task.

Preprocessing

Standard preprocessing of each EEG recording included low-pass filtering [50], artifact correction, down sampling. The preprocessing code is available at https://osf.io/zxr8m/ and was implemented within an automatic pipeline using the MNE software [39, 40]. After down sampling each EEG recording from 1 kHz to 500 Hz, signals were re-referenced to a common average. Then we applied a bandpass filter between 0.1 to 200 Hz with a notch filter to remove line noise and harmonics at 60, 120, and 180 Hz. The filter employed was a zero-phase finite impulse response (FIR) filter with transition bandwidth of approximately 5 Hz. Filtering was done in preparation for the ICA artifact correction process. Artifact correction was performed based on the MNE-python Independent Component Analysis (ICA) procedures [39, 40]. All the hyper-parameters for ICA processing were fixed except

peak to peak amplitude change, the peak to peak amplitude change that was used for rejecting bad segments of data. During the ICA artifact correction, eye blink and muscle movement artifact components were identified automatically without human inputs. These components were removed in the ICA space and projected back to the signal space after removal. A bandpass filter between 0.1 and 50 Hz was applied to the corrected signal. The bandpass filters used were zero-phase and the method was overlap-add FIR filtering. The filter length was chosen automatically based on the size of the transition region, which was 6.6 times the reciprocal of the shortest transition band for the Hamming window. Otherwise default MNE-python filtering parameters were used. Processed data were saved for later used. No other human input was involved in the preprocessing stage. All processing was done on a computer with Window 7 Enterprise, Intel® Core™ i7-3770 CPU @ 3.40 GHz, 7.68 usable RAM, and 64-bit operating system. The configuration of the system and python libraries are summarized in Table 1.

Filter Based and Thresholding Model

First, a bandpass filter was applied to the EEG signals with range 11 to 16 Hz, which is the full frequency range of the spindle band and includes both slow (11 – 13 Hz) and fast spindles (13.5 – 16 Hz). Root-Mean-Squares (RMSs) of individual selected channels were computed by convolving a Gaussian moving window (Eqs. 1 and 2).

$$\mathbf{W}[\mathbf{n}]=\mathbf{e}\frac{-\mathbf{n}^{\mathbf{2}}}{\mathbf{2}\boldsymbol{\sigma}^{\mathbf{2}}} \quad (1)$$

W denotes the Gaussian window function, and **n** denotes the sample size of the window. The standard deviation of the function (**σ**) is defined by the sample size of the window, which was set to **n**/0.68.

$$RMS[n]=\sqrt{\frac{\sum_{k=0}^{n}S[k]\cdot W[n-k]}{sample\,size}}*C \quad (2)$$

RMS[s] denotes the $\mathbf{n}^{th}$ sample of the computed ***Root-mean-square (RMS)***. *S[k]* denotes the k^{th} sample of the signal, and *W[n-k]* denotes the $(n\text{-}k)^{th}$ sample of the Gaussian window kernel. Convolution was applied during computation of the ***RMS*** using *Python.numpy.convolve* and kept the same sample size as the target signal using a Gaussian window kernel defined in Eq. 1. For the current study, the length of the kernel was 500 samples.

After computing RMS of individual selected channels, the harmonic mean of all these RMSs was saved for later analysis [53]. In the next step, low and high boundaries for classifying spindles were computed according to Eq. 3.

$$\text{Lower boundary}=\mu_{\tau}(\text{RMS})+\mathbf{T}^{\mathbf{l}}*\sigma_{\tau}(\text{RMS}) \quad (3.1)$$

$$\text{Higher boundary} = \mu_\tau(\text{RMS}) + \text{T}^{\text{h}} * \sigma_\tau(\text{RMS}) \quad (3.2)$$

μ_τ denotes trimmed mean, and $sigma;_\tau$ denotes trimmed standard deviation. T^l is the lower threshold value we define in the model and T^h is the upper threshold value we define in the model. Eq 3.1 computes a lower boundary of the ***RMS*** values of interest, and **3.2** computes a higher boundary of the ***RMS*** values of interest. Trimmed mean (μ_τ) and trimmed standard deviation (σ_τ) takes account 95% of the data to avoid influence of outliers (high amplitude fluctuation) [51].

If all values of a segment are between lower boundary and higher boundary and the duration of the segment is between 0.5 to 2 seconds, a possible spindle location will be marked at the peak of the segment. This was done on individual channels and/or the mean RMS (Eq. 3.1–2). To determine that a spindle occurred at a given time in the mean RMS, the spindle criteria were required to be met in a minimum of 3 individual channels with a 1 second variation. Additionally, sleep stage information was added to the finalized spindle classification in order to reduce the false alarm rate. Spindles marked in the first 300 and last 100 seconds are excluded because the duration of recordings differed across the two groups of subjects used in this report. An example of spindle segments classified by the filter based and thresholding model is shown in Supplemental Fig 1.

Optimization of the FBT model

Optimization of both the lower and upper threshold values in the FBT model was done using a grid search paradigm with respect to the Areas Under the Curve (AUC), where the curve is the receiver operating characteristic (ROC).

As we mentioned before, the training samples of the spindles and non-spindles would always be imbalanced. A skewed sample size of two classes would bias the classification towards the majority of the samples, which include the non-spindle class. Accuracy measures would be the least optimal metric in an imbalanced classification practice. Therefore, we choose AUC because it is a more sensitive and non-parametric criterion-free measure of classification performance [38]. AUC takes account of false positive and true positive classifications and returns a metric value between 0 and 1. An AUC of 0.5 or lower indicates a bad classification, while an AUC that is close to 1.0 indicates an excellent classification.

A grid search means testing a list of values of interest to determine the best value from the list of values. We fixed the other parameters in the thresholding pipeline (bandpass filter between 11 to 16 Hz, duration between 0.5 to 2 seconds, temporal toleration of 1 second, number of channels for decision making, etc.) and we classified spindles by iterating pairs of lower threshold (0.1 to 1) and upper threshold (2 to 3.5). For computational convenience, the AUC score for each of the 41 recordings were computed for each pair of possible pair of the lower and upper thresholds in the grid search.

With one particular pair of lower and upper thresholds, the FBT model would provide predicted information of the sleep spindle of one EEG recording, such as onsets and

durations of the spindles. Implementing Ray et al. [37] method, we segmented the original length of the EEG data by sliding a 3-second non-overlapping Hamming window. Using the onsets and durations, we predicted labels for each segmented signal to determine which segments held predicted spindles or not. With *Python.Scit-kit learn.model_selection.k_folds* [41], we fitted our data with the predicted labels using *Python.Scit-kit learn.linear_regression.logisticregressionCV* [41] by 5-fold cross-validation so that we could determine the regularization value. The features matrix included three categories: 1) local thresholding values (Eq. 4), 2) the frequency at the peak power spectral density, and 3) the peak power spectral density. Local thresholding values were used to capture similar thresholding patterns as the FBT model, representing the normalized distance between the most typical spindle signal and current processing signal.

$$\frac{\sum(\mathbf{RMS}>\mathrm{l})+\sum(\mathbf{RMS}<\mathbf{h})}{\sum(\mathbf{RMS}<\mathbf{h})-\sum(\mathbf{RMS}<\mathrm{l})} \quad (4)$$

l and ***h*** are values of the lower and upper boundaries of the RMS criteria computed locally at each segment (Eq. 2). The numerator is the sum of samples that are in between the lower and upper boundaries. The denominator is the difference of samples that is less than the upper boundary and less than the lower boundary. When the denominator equals zero, it raises a divide by zero condition, at which point we do not divide by the denominator. Instead, only the computed value of the numerator would be included in the feature matrix.

It is important to note that the features extracted from six individual channels were converted to a number of features (6 threshold features, 6 frequency features, and 6 power spectral features) by the number of segments (varying over subjects) matrix. The feature matrix was standardized before being fed to a logistic regression machine learning classifier (Eq. 5). The label fed to the classifier with the feature matrix was made based on the spindles found by the prior steps in the FBT model. In other words, no information other than the signals and sleep stages were put into the FBT model.

$$\mathbf{F_{ij}}=\frac{M_{ij}-\mu_i}{\sigma_i} \quad (5)$$

The normalized feature matrix standardizes features by removing the mean and scaling it to unit variance. Centering and scaling happens independently on each feature by computing the relevant statistics on the sample features in the training set. *i* represents the i^{th} row of the feature matrix and *j* represents j^{th} column of the feature matrix. Each value in the matrix, M of the i^{th} row and j^{th} column, subtracts the mean and standard deviation of its corresponding row. The standardizing process was implemented by *Python.Scit-kit learn.preprocessing.StamdardScaler* [41].

By doing this, we took the advantage of the decision function embedded in the *Python.Scit-kit learning.linear_model.logisticregressionCV* [41] to compute the probabilistic AUCs, and generated a probabilistic ROC curve to represent the cross-validation results.

In the "***Cross-validation of FBT Model***", we cross validated the optimization results using AUC as the metrics. The principle of optimizing both lower and upper thresholds is to maximize the average of between-subject AUC scores over the training/validation set.

Cross-validation of the FBT Model

The cross validation results of the FBT model was measured using ROC AUC scores and confusion matrices were also used. Using the Ray et al [37] method, we segmented the origin length of the whole EEG signal recording by a 3-second non-overlapping Hamming window. There were two reasons we implemented this method. The first was that a typical spindle could vary from 0.5 to 2 seconds so it was too convoluted to find the local matching rate between a manual marked and automatically marked spindle. The second was that the Ray et al [37] method produced fixed length windows to segment the data and this benefited us to prepare the data for machine learning. The FBT model provided us a list of onsets and durations of classified spindles for one particular pair of lower and upper thresholds. The information could be transformed to be predicted labels of each segmented signal. Based on the same principle, we could transform manually score spindle annotations to be true labels of each segmented signal. With each segmented signal, we labeled it "0" if this window did not overlap a designated spindle or "1" if this window overlapped a designated spindle.

According to Ray et al [37], we determined a "true positive" (TP) if a segment overlapped an expert scored spindle and an automated marked spindle; a "false negative" (FN) if a segment overlapped an expert scored spindle but not an automated marked spindle; a "false positive" (FP) if a segment overlapped an automated marked spindle but not an expert scored spindle; a "true negative" (TN) if a segment did not overlap any marked spindles. To better represent individual cross-validation results and compare between-subject performance, confusion matrices containing TN, FN, FP, and TP (from top to bottom and from left to right) were normalized, and TN, FN, FP, and TP became TN rate, FN rate, FP rate, and TP rate. After normalization, the TN rate and FN rate summed to 1, and the TP rate and FP rate summed to 1. Sensitivity and specificity were computed (Eq. 6) and summarized with other measures (i.e. AUC) in Supplemental Fig 2. TP rate is equivalent to sensitivity and TN rate is equivalent to specificity.

$$\mathbf{sensitivity}=\frac{\mathbf{TP}}{\mathbf{TP}+\mathbf{FN}}=\mathbf{true\,positive\,rate} \quad (6.1)$$

$$\mathbf{specificity}=\frac{\mathbf{TN}}{\mathbf{TN}+\mathbf{FP}}=\mathbf{true\,negative\,rate} \quad (6.2)$$

In order to evaluate the optimization results, we used a standard 10-fold cross validation (CV) [54]. There were 41 EEG recordings that were used in the evaluation. This yielded a partition into 9 groups of 4 recordings and 1 group of 5 recordings. The CV consisted of 10 different folds, each fold with 9 groups in the fitting set, and 1 group in the evaluation group. For each of the 10 folds, we took the lower and upper threshold pair with the best average AUC score from the recordings in the fitting set. The evaluation error was the average AUC score of the evaluation set resulting from using the lower and upper threshold pair that maximizes the AUC score on the fitting set.

Machine Learning Pipeline

We segmented the data with a fixed length Hamming window (3 seconds) and labeled each depending on whether a segment overlapped any spindles based on the manually marked spindles. This provided us with labeled training data for supervised machine learning model selection. The training features contained three categories (same as in the probabilistic step of the FBT model): raw signal values (6 channels x 3 seconds x sampling rate), peak frequencies (6 channel x 1), and power spectral density at the peak frequencies (6 channels x 1). Thus, the number of features was 9012 with a sampling rate of 500 Hz. To better represent our data and balance the sample sizes of the spindle and non-spindle classes, we fed the model selection algorithm mainly the FP (classified by FBT but not by experts) and FN (classified by experts but not by FBT) cases under the optimized FBT model. If in both cases the sample size of spindle and non-spindle classes were not balanced, we added either TP (true spindles classified by FBT and experts) or TN (true non-spindles classified by FBT and experts) cases. It is important to note that the TP and FN cases were labeled the "spindles" class, while the TN and FP cases were labeled the "non-spindles" class. The TP, TN, FP, and FN cases were determined by the optimized FBT model using 0.4 and 3.5 as the lower and upper thresholds. We searched the machine learning model selection using Tree-based Pipeline Optimization Tool (TPOT [42]) on between-subject training data grouped by all subject samples. *TPOT* implements a generic algorithm to search over a broad range of supervised classification algorithms that follows the *Python Scit-kit learn* library [41], including preprocessors, transformers, feature selection techniques, estimators, and their hyperparameters, without any domain knowledge of human inputs of the data. The finalized machine learning pipeline was determined by AUC as the measure metric. After the model selection using *TPOT* [42], we implemented the finalized machine learning pipeline on individual subjects' data. Within each individual data, a 5-fold cross-validation was implemented. A total of five AUCs were computed and we chose the interpolated average of the 5 ROC curves to represent the cross-validation results. The finalized machine learning model predicted class probabilities of instances of features. To compensate for the imbalanced sample sizes between the spindle and non-spindle classes, we defined the ratio between the spindle sample size and the total sample size because the default threshold was 0.5 [52]. For example, if the model predicted that an instance of features had 0.5 probability to be a spindle and the ratio was 0.2 (20 spindles and 100 non-spindles), then the instance was classified as a spindle because the predicted probability exceeded the new threshold.

Model Comparison: FBT Model and Machine Learning Pipeline

We computed AUCs for the FBT model and the finalized machine learning pipeline to compare the cross-validation results at the individual level. By doing so, we might lose the localization of individual spindles, but it gave us better resolution about how the FBT model and the machine learning model performed to provide probabilistic estimates of spindle occurrence. Measure metrics such as AUC, confusion matrix, sensitivity and specificity were used for the model comparison. A pseudo-cross-validation method was implemented for the FBT model within each individual data. Within each individual data, same set of parameters were applied (optimized lower threshold, optimized upper threshold, bandpass filter between 11 to 16 Hz, duration between 0.5 to 2 seconds, temporal toleration of 1 second, number of channels for decision making, etc.), and onsets and durations of local spindles were marked. The predicted annotations were transformed to predicted labels of segmented signals using 3-second Hamming non-overlapping window, same as in the Machine learning model. Manually scored annotations were also transformed to true labels of segmented signals. Predicted labels and true labels were split to major set (80%) and minor set (20%). The cross-validation measure was computed on the major set. Splitting labels were repeated 5 times for each individual data. The comparison results were summarized in Fig. 6.

Manual Sleep Spindle Annotations

The EDF browser (http://www.teuniz.net/edfbrowser/) was used to manually annotate sleep stages and spindles. We defined spindles that had dominant power at 11 – 16 Hz with a duration varying between 0.5 – 2 seconds. Only those that occurred across several channels ($n > 3$) with minimal temporal variation (< 1 second) were marked. Channels of interest were F3, F4, C3, C4, O1, and O2 (Fig 2). These channels were chosen because their spatial representation covered most of the brain from frontal to occipital areas [7]. The filter used was a Bandpass Butterworth in the 2nd order. Each dataset was marked based on every 30 second interval. Within each interval, a sleep stage and spindle locations were marked (https://osf.io/h68a3/). Decisions about locations of spindles were made based on visible spindle features occurring in more than three of the channels. Spindle markers were places at the beginning of the spindles. No information about durations of individual spindles was marked. Thus, all spindles were set to 2 seconds by default, with each beginning at 0.5 seconds prior to where the spindle was marked (Fig 2). Example 1 and 2 are spindles marked in a dataset collected for the present study, and example 3 is a spindle marked in the DREAMS dataset. Preprocessed raw and filtered signals are shown. The duration of a spindle is shaded in red.

DREAMS Project Dataset

We applied our pipelines (FBT, machine learning, and cross-validation) to the publically available DREAMS project data [1] to investigate how the FBT model set a benchmark for deriving a machine learning pipeline. The DREAMS data were acquired in a sleep study and each recording contained only one channel (C3 or Cz). The data were given independently to two experts for manually marking sleep spindles. We applied our pipelines to find the best thresholds for the FBT model, construct training data to search for the best machine learning pipeline, and cross validate both the FBT model and finalized machine learning pipeline, as

well as manual annotations from the second expert and scores provided by Devuyst et al [1]. In other words, we considered the spindle annotations marked by the first expert as gold standard. Since each EEG recording in the DREAMS project data contained one channel, we determined spindles only based on one global RMS with the feature criteria we defined in "***Filter Based and Thresholding Pipeline***" section. The FBT model optimization and machine learning model selection were applied to all eight recordings, but the model comparison process was only applied to six recordings because the second expert only scored these recordings. It is important to note that we kept the 0.5 – 2 second duration assumption for the DREAMS datasets although the data contained spindle duration between 0.61 to 1.89 seconds (M = 0.81, SD = 0.27).

Permutation Test

After we classified spindles using the FBT approach, a randomized permutation test (https://osf.io/jnfk6/, permutation test.py) was used to compare average spindle counts for the low and high load working memory tasks. The randomized permutation test was chosen over a parametric test because of the small sample size (N = 30, 15 pairs) as such a small sample size might violate the normal distribution assumption. First, we computed the difference between two sets of classified spindles (set 1 low load and set 2 high load for each subject). Then we randomly shuffled the two sets of values so that some of the values in set 1 were in set 2 and some of the values in set 2 were in set 1. We computed the difference between the new set 1 and new set 2. After doing this 500 times, we computed the quartiles of the distribution of the differences computed in the randomization process to determine whether the value of the quartile exceeded an *a priori* threshold (p = 0.05). We repeated this process 500 times to compute the confidence interval of the percentage quartile. The same computation process was implemented to compare cross-validation performance using data collected in the present study and also on data from the DREAMS study. Results reported are the mean and standard deviation of the posterior p values.

Results

Detected Sleep Spindles and Spindle Densities

Some examples of sleep spindles detected by the FBT model are shown in Fig. 3. Examples 1 and 2 are spindles detected by the FBT model in our data, while example 3 is a spindle detected by the FBT model in the DREAMS data. The average duration of the auto-detected spindles was 1.12 seconds (SD = 0.36), where the minimum was 0.5 seconds and maximum was 1.99 seconds. Manually marked spindles are shown in Fig. 4 and a comparison of manually marked and automated classified spindles are shown in Fig. 5 in terms of spindle densities. Spindle densities were computed and visualized with *Python Seaborn* library [44]. The kernel density was 20 samples and the first 300 and last 100 seconds were excluded when manually annotating spindles. The best lower and higher thresholds for the FBT model were 0.4 and 3.5 based on AUC and accuracy. The reason we show spindle density is the FBT model took global patterns to detect spindles and spindle density represents variations of spindle count in terms of temporal dynamics [47].

According to the manually marked scores, spindles occurred around 1000 to 1500 seconds (Fig. 4). Annotators classified 2.81 less spindles on days that subjects performed the high load task (M = 67.37, SD = 35.03) than on days that subjects performed the low load task (M = 70.18, SD = 39.51); however, the difference is not statistically significant (randomized permutation test, p values = 0.58 +/− 0.02, M +/− SD). The FBT model classified 14.25 more spindles on days that subjects performed the high load task (M = 105.93, SD = 63.89) than on days that subjects performed the low load task (M = 91.68, SD = 63.81), a difference that is not statistically significant (randomized permutation test, p values = 0.272 +/− 0.204, M +/− SD).

Spindle densities by the FBT model visually matched manually marked scores for long recordings better than for short recordings. In the short recordings, the FBT model classified fewer spindles than the human annotators. However, there was no significant difference between the numbers of spindles detected by the FBT model in the long and short recordings (permutation test, p value = 0.245 +/− 0.018, M +/− SD). There was significant evidence to suggest that the machine learning model performed better for the long recordings than the short recordings in terms of AUC (permutation test, p values = 10^{-4} +/− 10^{-4}, M +/− SD). One of the reasons that machine learning model performed better for the long recordings may be that machine learning models generally perform better with more data.

Machine Learning Pipeline

Based on the training data prepared by the FBT model, the finalized machine learning pipeline used in the present study included a feature selection and an estimate model determined by the *TPOT* library [42]. The feature selection union used a nested voting classifier (VC) and a decision tree classifier (DTC) and the estimate model was gradient boosting classifier (GBC). The VC is a soft maximum classifier for unfitted estimators and the DTC is a classifier with auto split and auto maximum stop point in choosing feature samples. The GBC allows for the optimization of arbitrary differentiable loss functions. The loss function used in the current study was logistic regression for classification with probabilistic outputs. Learning rate was chosen to be 0.24 and the number of gradient boosting was 500 stages to avoid overfitting and improve performance. Also, the maximum of feature selected was set to be 0.24 of all possible features. VC, DTC and GBC were all provided in the *Python Scit-kit learn* library [43]. Hyperparameters mentioned above were specified and the rest were left as default.

The average between-subject AUC score for the FBT model across the 10-fold was 0.58 +/− 0.055 (M +/− SD), TP rate was 0.33 +/− 0.17 (M +/− SD), and FN rate was 0.16 +/− 0.12 (M +/− SD). As for the average within-subject cross validation results, the machine learning pipeline (M = 0.65, SD = 0.112) was significantly better in classifying spindle segments than the FBT model (M = 0.59, SD = 0.051), $t_{(40)} = 9.12$, $p < 0.05$ (Fig 6). Thirteen of 41 recordings showed that results from the thresholding pipeline and the machine learning pipeline were matched. Particularly in Fig 6(a) we show the average of the 5-fold cross-validation result of subject 29 at day 1 for the machine learning pipeline and of subject 25 at day 2 for the FBT model because these two subjects' data represent the median performance in corresponding models. The ROC curves of the FBT model were generated by a softmax

decision function implementing logistic regression in order to provide probabilistic interpretation of the classifications. We observed that the FBT model generated less variability across subjects while the machine learning model generated more between-subject variability (Fig 6a and b). This might be due to the fact that the FBT model implemented a pseudo “train-test-split” cross-validation, and the cross-validation was applied to 80% of the classified spindle segments, leaving the rest 20% untouched. Cross-validation was applied to 20% of the available data for the machine learning pipeline, which produced more variability in randomly selecting the testing set of the data due to a small data sample size. The between-subject variations were also reflected by the confusion matrices shown in Fig 6(c). The machine learning pipeline had worse TN rate (upper left cell) and TP rate (lower right cell) than the FBT model. This could be due to how we randomly selected one of the 5 folds of cross-validation results. Detailed reports of within-subject AUC, TP rate, FP rate, TN rate, FN rate, sensitivity, and specificity are summarized in Supplemental Figure 2.

DREAMS Project Data

The best lower and higher threshold values for the FBT model were 0.48 and 3.48. The finalized machine learning pipeline determined by *TPOT* contained an extra trees classifier (ETC). The ETC implemented a meta estimator that fit a range of randomized decision trees on various sub-samples of the training dataset and used averaging to improve the classification performance and control over-fitting. Entropy loss was chosen to be the loss function, measuring information gain through the model training iterations. A total of 500 tree estimators and 0.32 of feature selection rate were set to control for the model training. Results of applying our pipelines (FBT, model selection, machine learning and cross-validation) to the DREAMS project data were similar (Fig 7(a)). Machine learning pipeline (M = 0.67, SD = 0.03) marginally outperformed the FBT model (M = 0.59, SD = 0.03) (permutation test, p values = 0.052 +/− 10^{-3}, M +/− SD), and it was not significantly different from the results of the second expert (M =0. 73, SD = 0.04) (permutation, p values = 0.190 +/− 10^{-3}, M +/− SD). Results of Devuyst et al. (2011) thresholding approach (M = 0.77, SD = 0.02) outperformed the machine learning pipeline (permutation, p values = 0.014 +/− 10^{-3}, M +/− SD), but did not significantly outperform the second expert (permutation, p values = 0.23 +/− 0.02, M +/− SD). Particularly, in Fig 7(b), we showed the cross-validation results of all datasets (preprocessed data, http://www.tcts.fpms.ac.be/~devuyst/Databases/DatabaseSpindles/) among the thresholding pipeline, machine learning pipeline, the second expert, and the Devuyst et al. [1] results, and the median sample dataset within each method were highlighted in black. In Fig 7(c), we showed the normalized confusion matrices, annotating true negative rate, false negative rate, false positive rate, and true positive rate with mean +/− standard deviation. The FBT model had the lowest performance due to the low TP rate, meaning that the FBT model was not sensitive to detect sleep spindle segments. The machine learning pipeline performed similar to the second expert in terms of TP rate and TN rate, suggesting the machine learning pipeline derived from the FBT model improve the detecting sensitivity and achieved the goal of approximating human performance.

Discussion

We have shown that automated pipelines with minimal human input were able to take flexible variables to provide a classification bench mark and sample suitable training data to determine a best one-layer machine learning pipeline using naïve features. Both between-subject and within-subject cross-validation results were reported for the FBT model, and within-subject cross-validation results were reported for the selected machine learning model. The actual decision making in the FBT model relies on passing the signal RMSs through ranges of features of a typical spindle defined by experts, and the probabilistic decisions were made by a softmax function implementing a logistic regression classifier. With our long and short recordings, we implemented a multi-channel decision making criterion to the FBT model, and the results showed that we could optimize the FBT model significantly beyond chance level. The FBT model provided a better representation of "a typical sleep spindle" because the pre-defined feature parameters in the FBT model were more generative while capturing the critical statistics of the set of sleep spindles. With the knowledge gained from the FBT (i.e., the data statistics about the spindles) we implemented model selection and cross-validation of machine learning pipelines and significantly improved the classification performance with less flexible viewing windows (fixed 3-second). Results of implementing these pipelines on our data provided evidence that the FBT model and the machine learning pipeline derived from the FBT model were able to classify sleep spindles among multiple EEG channels and provided probabilistic statistics of the classification. Results of implementing our pipelines to the DREAMS project data [1] supported that the FBT model and the machine learning pipeline derived from the FBT model captured perspectives of an expert annotator without intensive feature engineering and provided probabilistic statistics of the target classification. In general, the automated pipelines classified fewer spindles than expert annotators because the automated pipelines took global signal patterns and were restricted by annotated sleep stages (accepted spindle instances only in non-REM sleep stage 2 signals), while expert annotators took local information of the segmented signals.

Annotators classified 2.81 fewer spindles on days that subjects performed the high load task than on days that subjects performed the low load task. The FBT model classified 14.25 more spindles on days that subjects performed the high load memory task than on days that subjects performed the low load memory task. In the Purcel et al. study [51], the average spindle density (spindle counts divided by total nap time) was 1.18, while it is 4.43 (SD = 2.62) in the present study by manual annotating and 2.80 (SD = 1.45) by the FBT model, both of which are higher. One reason densities in the present study are higher than in the Purcel et al. study may be because we excluded subjects who had fewer than 5 spindles.

Four main concerns for implementing one-layer machine learning models to classify sleep spindles are 1) small sample size, 2) loss of information for localization of spindles, 3) imbalanced sampling, and 4) representing a general knowledge about the spindles. Most of the machine learning models require a large data set (N > 1000) and they take segmented EEG signals to determine the probability that a segment contains/overlaps a sleep spindle. It is difficult to record sleep activity long enough to meet the sample size criterion with a few subjects. On the other hand, segmenting continuous recordings might lead to loss of

localization information, while for sleep studies, the locations of sleep spindles are as important as the counts of the spindles.

In practice, with highly variable individual differences, the number of samples in the class “spindle” is often less than the number of samples in class “non-spindles” (about 1:15 in our data, and about 1:10 in the DREAMS data). Additionally, the definition of a “typical spindle” varies from study to study, as well as within the same study. Instead of flexibly defining a spindle, we usually need to feed machine learning models data that represent “spindle” and “non-spindle” discretely. An advantage of implementing a FBT approach is that it would provide a better understanding of the most important features, such as onsets, durations, and tempo-spectral information. Such an approach is equivalent to a “pre-trained” machine learning model, and the features are fed to the machine learning models in forms of feature vector representation, which reduces the computational difficulty of model selection in the machine learning approach.

In the current study, the FBT model implemented the multi-channel criterion used in Begmann et al. [19] study, taking account of the oscillation pattern among multiple channels, so that we can determine the most consistent spindles. By consistent, we aim to classify spindles that occur in more than half of the channels of interest. Because the FBT model takes global signal patterns, the FBT model would be easy to implement in clinical settings. With a cross-validation module, the FBT (M = 41.60 sec, SD = 16.34 sec) performed significantly faster than the best naïve machine learning pipeline (M = 115.16 sec, SD = 82.29 sec) for our data, $t_{(40)} = -5.18$, $p < 0.001$. It is noted that some of our data are more than 60 minutes long and some of the data are 30 minutes long with sampling rates of 1k Hz. We down sampled our data to 500 Hz to increase computing speed. For data that are less complex, it should be faster to compute.

Another advantage of implementing the FBT model is that feature parameters in the model are very flexible and users could define all of them based on their particular scientific needs. For example, with our preliminary test, the best window size to compute the RMS is the sampling rate of the data, and in our case, it was the down sampled frequency rate of 500 Hz. However, it does not mean that this is the universally optimal value. Parameters in our pipelines were optimized via cross-validation, and we propose an exhaustive grid search paradigm to optimize most of the parameters. It would be easy for a clinical study to classify spindles with their feature parameters, feeding the FBT model raw EEG data and getting spindle annotation data frames in a short time. If a user wants to optimize the results, it will be very easy to implement the cross-validation pipeline to the FBT model with expert scored spindle annotations. By cross-validation, the optimized FBT model would capture an expert’s scoring statistics and help perform classifications on novel data automatically. Since the optimization takes grid search paradigms, the optimization process would take a long time to run. For our data, it took three days to process all possible pairs of lower and higher threshold values for all our data to have a summary containing cross-validation results. And we chose the best pair threshold values based on the cross-validation results using AUC and accuracy as the metrics.

Additionally, our pipelines, including the FBT model, the machine learning pipeline, and the cross-validation pipeline, require minimal human input. We argue that this is good for replications because our pipelines take objective criteria, process raw signals, and provide interpretable annotation data frames containing onsets and durations of sleep spindles. The FBT model takes two inputs: the signals and the sleep stages, and produce the onsets and durations of classified sleep spindles. Many machine learning models mentioned in the introduction either do not provide this information or provide this information within a certain length of local windows, which is not flexible enough to view sleep spindles as independent Poisson events. Our machine learning pipeline performed similarly to one-layer machine learning models. It did not require intensive feature engineering to optimize the performance.

In the machine learning pipeline, we adapted a changing threshold to the probabilistic results of the classifying predictions. The threshold reflected the proportion of the spindle class in the total training samples. This was an adaptive decision-making criterion because we were working on highly imbalance classification problems [52]. We implemented the adaptive thresholding in the decision-making step instead of tuning the hyper parameters of the model because this was simpler and more intuitive in terms of interpreting the predicted results. Since the FBT model nested a decision-making function implementing logistic regression classifier to provide probabilistic predictions, it would be reasonable to implement the adaptive thresholding in the step and theoretically would improve the classification performance. The prediction that the performance would improve is that the adaptive thresholding takes account of the trade-off between specificity and sensitivity and accepts more spindle instances, which means sacrificing specificity to gain sensitivity (Supplemental Figure 3).

There are several future directions that could improve our models. The FBT model takes signal information and sleep stage as inputs and returns onsets, durations, and amplitude information as outputs. Implementing the changing thresholding method to its probabilistic decision-making step, we could feed the adjusted probabilities to the identified sleep spindles prior to this step, reducing the false negatives in classifying sleep spindles. The feedback loop could go through several iterations. As for the machine learning perspective, the current study focused on the model selection using only the naïve features that captured multi-channel statistics (i.e. signal, frequency, power). A feature engineering [48] could be performed to improve classification performance while using standard one-layer machine learning models (i.e. linear models, support vector machines, and ensemble models). Furthermore, in order to understand how statistical representations of the signals inform automated approaches in classifying sleep spindles, deep neural networks could be used to learn those statistics by implementing reconstruction network frameworks.

Conclusion

The objective of the current study was to design a FBT pipeline to process raw EEG recordings of sleep and return localized sleep spindle information. The process was completed on multiple channels quickly and provided statistical representations for machine learning model selection. By implementing a decision function on the detected spindles,

particular segments of the signals were sampled to form feature vectors that represented well knowledge of both spindles and non-spindles. With minimal human input, the pipelines are easy to implement and the results are intuitive because the goal of optimizing our pipelines is to learn the global patterns of the EEG signals through multiple channels. A future direction to improve this approach is to extract better feature representations via graphic models and to emulate the human annotator better. On the clinical side, future work should investigate extending the set of neural events and distinguishing different types of sleep spindles (i.e., slow and fast spindles), which can be automatically detected with models mentioned in the present study. Both of these approaches could make spindle detection more objective thereby improving the efficiency and reproducibility of both basic research and clinical studies.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

The authors would like to thank Forrest Armstrong, Miriam San Lucas, Ben Fernandez, Chelsea Reichert, and Erin Hunt for help with data collection.

Funding

This work was supported by PSC-CUNY award TRADB #67546-00 45. Research reported in this publication was supported by the National Institute of General Medical Sciences of the National Institutes of Health under Award Number SC2GM109346. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

1. Devuyst, Stéphanie, et al. Automatic sleep spindles detection—overview and development of a standard proposal assessment method. Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE; IEEE; 2011.
2. Maquet, Pierre. The role of sleep in learning and memory. Science. 2001; 294(5544):1048–1052. [PubMed: 11691982]
3. Rasch, Björn, Born, Jan. About sleep's role in memory. Physiological reviews. 2013; 93(2):681–766. [PubMed: 23589831]
4. Stickgold, Robert, Walker, Matthew P. Sleep-dependent memory triage: evolving generalization through selective processing. Nature neuroscience. 2013; 16(2):139–145. [PubMed: 23354387]
5. Hoel, Erik P., et al. Synaptic refinement during development and its effect on slow-wave activity: a computational study. Journal of neurophysiology. 2016; 115(4):2199–2213. [PubMed: 26843602]
6. Wallant, Dorothée Coppieters'T., Maquet, Pierre, Phillips, Christophe. Sleep spindles as an electrographic element: description and automatic detection methods. Neural Plasticity. 2016; 2016
7. Schabus, Manuel, et al. Sleep spindles and their significance for declarative memory consolidation. Sleep. 2004; 27(8):1479–1485. [PubMed: 15683137]
8. Fogel, Stuart M., Smith, Carlyle T. The function of the sleep spindle: a physiological index of intelligence and a mechanism for sleep-dependent memory consolidation. Neuroscience & Biobehavioral Reviews. 2011; 35(5):1154–1165. [PubMed: 21167865]
9. Liu, Yawu, et al. Analysis of regional MRI volumes and thicknesses as predictors of conversion from mild cognitive impairment to Alzheimer's disease. Neurobiology of aging. 2010; 31(8):1375–1385. [PubMed: 20447732]
10. Muller, Lyle, et al. The stimulus-evoked population response in visual cortex of awake monkey is a propagating wave. Nature communications. 2014; 5

11. Ebrahimi, Farideh, et al. Automatic sleep stage classification based on EEG signals by using neural networks and wavelet packet coefficients. Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE; IEEE; 2008.
12. Tsanas, Athanasios, Clifford, Gari D. Stage-independent, single lead EEG sleep spindle detection using the continuous wavelet transform and local weighted smoothing. Frontiers in human neuroscience. 2015; 9:181. [PubMed: 25926784]
13. Andrillon, Thomas, et al. Sleep spindles in humans: insights from intracranial EEG and unit recordings. Journal of Neuroscience. 2011; 31(49):17821–17834. [PubMed: 22159098]
14. Mölle, Matthias, et al. Grouping of spindle activity during slow oscillations in human non-rapid eye movement sleep. Journal of Neuroscience. 2002; 22(24):10941–10947. [PubMed: 12486189]
15. De Gennaro, Luigi, Ferrara, Michele. Sleep spindles: an overview. 2003:423–440.
16. Terrier G, Gottesmann CL. Study of cortical spindles during sleep in the rat. Brain research bulletin. 1978; 3(6):701–706. [PubMed: 162576]
17. Astori, Simone, Wimmer, Ralf D., Lüthi, Anita. Manipulating sleep spindles–expanding views on sleep, memory, and disease. Trends in neurosciences. 2013; 36(12):738–748. [PubMed: 24210901]
18. Mölle, Matthias, et al. Fast and slow spindles during the sleep slow oscillation: disparate coalescence and engagement in memory processing. Sleep. 2011; 34(10):1411. [PubMed: 21966073]
19. Bergmann, Til O., et al. Sleep spindle-related reactivation of category-specific cortical regions after learning face-scene associations. Neuroimage. 2012; 59(3):2733–2742. [PubMed: 22037418]
20. Schimicek P, et al. Automatic sleep-spindle detection procedure: aspects of reliability and validity. Clinical EEG and neuroscience. 1994; 25(1):26–29.
21. Perumalsamy, Venkatakrishnan, Sankaranarayanan, Sangeetha, Rajamony, Sukanesh. Sleep spindles detection from human sleep EEG signals using autoregressive (AR) model: a surrogate data approach. Journal of Biomedical Science and Engineering. 2009; 2(05):294.
22. Palliyali, Abdul J., Ahmed, Mohammad N., Ahmed, Beena. Using a quadratic parameter sinusoid model to characterize the structure of EEG sleep spindles. Frontiers in human neuroscience. 2015; 9
23. Subasi, Abdulhamit, et al. Wavelet neural network classification of EEG signals by using AR model with MLE preprocessing. Neural Networks. 2005; 18(7):985–997. [PubMed: 15921885]
24. Huupponen E, et al. Optimization of sigma amplitude threshold in sleep spindle detection. Journal of sleep research. 2000; 9(4):327–334. [PubMed: 11386202]
25. Devuyst, Stéphanie, et al. Automatic K-complexes detection in sleep EEG recordings using likelihood thresholds. Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE; IEEE; 2010.
26. Acir, Nurettin, Güzeliş, Cüneyt. Automatic spike detection in EEG by a two-stage procedure based on support vector machines. Computers in Biology and Medicine. 2004; 34(7):561–575. [PubMed: 15369708]
27. Duman, Fazil, et al. Efficient sleep spindle detection algorithm with decision tree. Expert Systems with Applications. 2009; 36(6):9980–9985.
28. Güneş, Salih, et al. Sleep spindles recognition system based on time and frequency domain features. Expert Systems with Applications. 2011; 38(3):2455–2461.
29. Jobert M, et al. Topographical analysis of sleep spindle activity. Neuropsychobiology. 1992; 26(4): 210–217. [PubMed: 1299797]
30. Koley, Bijoy Laxmi, Dey, Debangshu. Detection of characteristic waves of sleep EEG by continuous wavelet transform. Computing and Communication Systems (NCCCS), 2012 National Conference on; IEEE; 2012.
31. Jaleel, Abdul, et al. Pilot validation of a mimicking algorithm for detection of sleep spindles and K-complexes. World Congress on Medical Physics and Biomedical Engineering; May 26–31, 2012; Beijing, China. Springer Berlin Heidelberg; 2013.
32. Camilleri, Tracey A., Camilleri, Kenneth P., Fabri, Simon G. Automatic detection of spindles and K-complexes in sleep EEG using switching multiple models. Biomedical Signal Processing and Control. 2014; 10:117–127.

33. Parekh, Ankit, et al. Detection of K-complexes and sleep spindles (DETOKS) using sparse optimization. Journal of neuroscience methods. 2015; 251:37–46. [PubMed: 25956566]
34. Chow, Tommy WS., Fang, Yong. A recurrent neural-network-based real-time learning control strategy applying to nonlinear systems with unknown dynamics. IEEE transactions on industrial electronics. 1998; 45(1):151–161.
35. O'Reilly, Christian, Nielsen, Tore. Automatic sleep spindle detection: benchmarking with fine temporal resolution using open science tools. Frontiers in human neuroscience. 2015; 9:353. [PubMed: 26157375]
36. Abraham, Alexandre, et al. Machine learning for neuroimaging with scikit-learn. 2014 arXiv preprint arXiv:1412.3919.
37. Ray, Laura B., et al. Validating an automated sleep spindle detection algorithm using an individualized approach. Journal of sleep research. 2010; 19(2):374–378. [PubMed: 20149067]
38. Powers, David Martin. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. 2011
39. Gramfort, Alexandre, et al. MNE software for processing MEG and EEG data. Neuroimage. 2014; 86:446–460. [PubMed: 24161808]
40. Gramfort, Alexandre, et al. MEG and EEG data analysis with MNE-Python. Frontiers in neuroscience. 2013; 7:267. [PubMed: 24431986]
41. Buitinck, Lars, et al. API design for machine learning software: experiences from the scikit-learn project. 2013 arXiv preprint arXiv:1309.0238.
42. Braun, M., et al. Proceedings, chapter Comparison of Multi-objective Evolutionary Optimization in Smart Building Scenarios; Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016; Porto, Portugal. March 30–April 1, 2016; Springer International Publishing; 2016.
43. Olson, Randal S., Urbanowicz, Ryan J., Andrews, Peter C., Lavender, Nicole A., Kidd, La Creis, Moore, Jason H. Automating biomedical data science through tree-based pipeline optimization. Applications of Evolutionary Computation. 2016:123–137.
44. Waskom, M., et al. seaborn: v0.5.0 (November 2014) (ZENODO). 2014.
45. Purcell SM, Manoach DS, Demanuele C, Cade BE, Mariani S, Cox R, Panagiotaropoulou G, Saxena R, Pan JQ, Smoller JW, Redline S. Characterizing sleep spindles in 11,630 individuals from the National Sleep Research Resource. Nature communications. 2017 Jun 26.8:15930.
46. Coppieters't Wallant D, Maquet P, Phillips C. Sleep spindles as an electrographic element: description and automatic detection methods. Neural Plast Neural Plast. 2016; 2016:e6783812.
47. Gaillard J-M, Blois R. Spindle density in sleep of normal subjects. Sleep. 1981; 4(4):385–391. [PubMed: 7313391]
48. SCHAABOVÁ, H., et al. Application of Artificial Neural Networks for Analyses of EEG Record with Semi-Automated Etalons Extraction: A Pilot Study. Communications in Computer and Information Science, Edition No. 1. sv. 629; Engineering Applications of Neural Networks. Engineering Applications of Neural Networks (EANN) 2016; Aberdeen. 2016-09-02/2016-09-05; Cham: Springer International Publishing; 2016. p. 94-107.
49. Ellmore TM, Feng A, Ng K, Dewan L, Root JC. The Effects of Changing Attention and Context in an Awake Offline Processing Period on Visual Long-Term Memory. Frontiers in psychology. 2015:6. [PubMed: 25759672]
50. Rabiner, Lawrence R., Gold, Bernard. Theory and application of digital signal processing. Vol. 1975. Englewood Cliffs, NJ: Prentice-Hall, Inc; 1975. p. 777
51. Purcell SM, et al. Characterizing sleep spindles in 11,630 individuals from the National Sleep Research Resource. Nature communications. 2017; 8:15930.
52. Chen JJ, et al. Decision threshold adjustment in class prediction. SAR and QSAR in Environmental Research. 2006; 17(3):337–352. [PubMed: 16815772]
53. Harding G, et al. The electroencephalogram in three cases of periodic psychosis. Electroencephalography and clinical neurophysiology. 1966; 21(1):59–66. [PubMed: 4165365]
54. Stone, Mervyn. Asymptotics for and against cross-validation. Biometrika. 1977:29–35.

Highlights

- We developed an automated pipeline for sleep spindle classification.
- Tested on multichannel EEG data collected during naps.
- Spindle identification is quick; option to guide sophisticated machine learning model selection.
- Pipeline is open source and easy to implement in practical settings for spindle classification.
- Pipeline speeds manual spindle annotation process for larger EEG datasets.

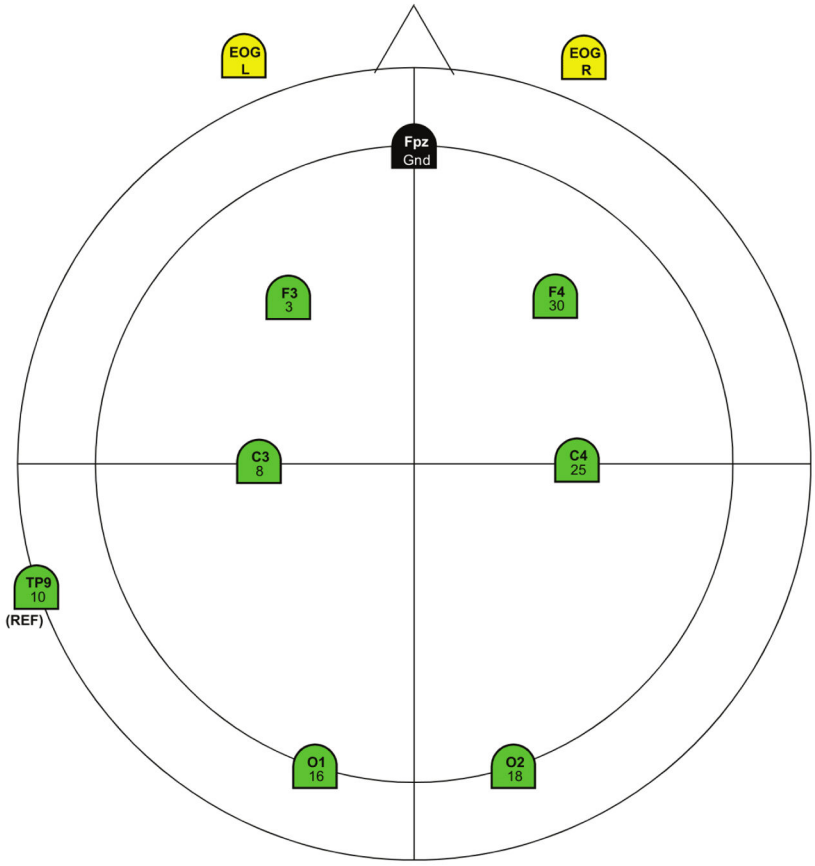

Figure 1. Spatial locations for electrodes of interest (10/20 international standard F3, F4, C3, C4, O1, and O2), EOG, and the reference electrode

Channel Fpz was used as the ground channel and channel TP9 was used as the reference channel during recording. We used a vertical EOG approach in our design. The “EOG L” electrode was placed under the left eye on the maxilla while the “EOG R” electrode was placed above the right eyebrow on the frontal bone. Both electrodes on either side were in line with the middle of the eyes. Recorded data was re-referenced to the average of the data during the preprocessing phase. Because the FBT model detects sleep spindles based on fluctuations relative to the individual channels after bandpass filtering, an asymmetrical amplitude distribution would not influence spindle detection.

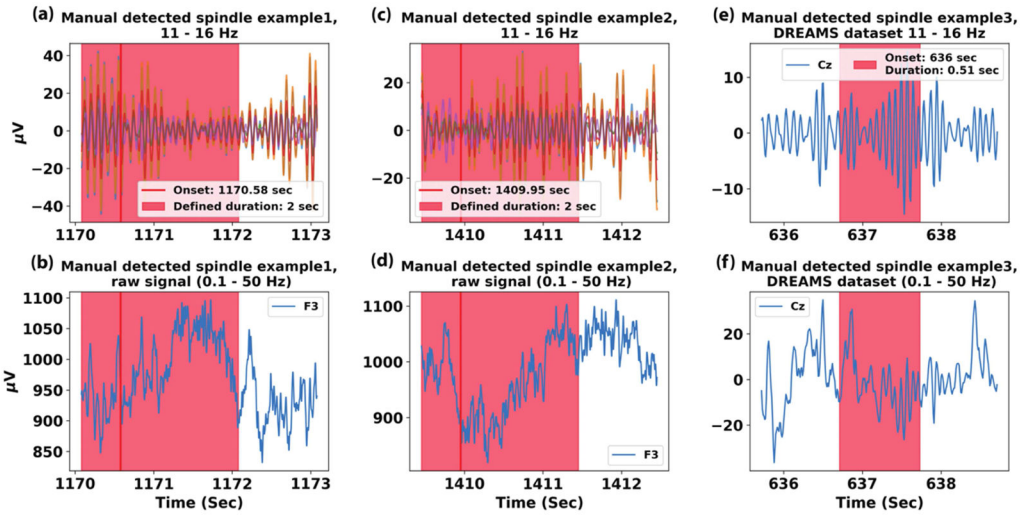

Figure 2. Three examples of manually marked sleep spindles

Each column represents one example of a manually annotated sleep spindle. The last column contains an example spindle signal from the DREAMS data. The first row contains preprocessed spindles without a bandpass filter at 11 to 16 Hz. Since the duration of spindles was not marked by the annotators in our data, these spindles were set to 2 seconds by default with each beginning at 0.5 seconds (start of the red shaded span) prior to the marked onset (red line). Duration information was provided in the DREAMS data and we specify the onset plus the duration on the x ticks.

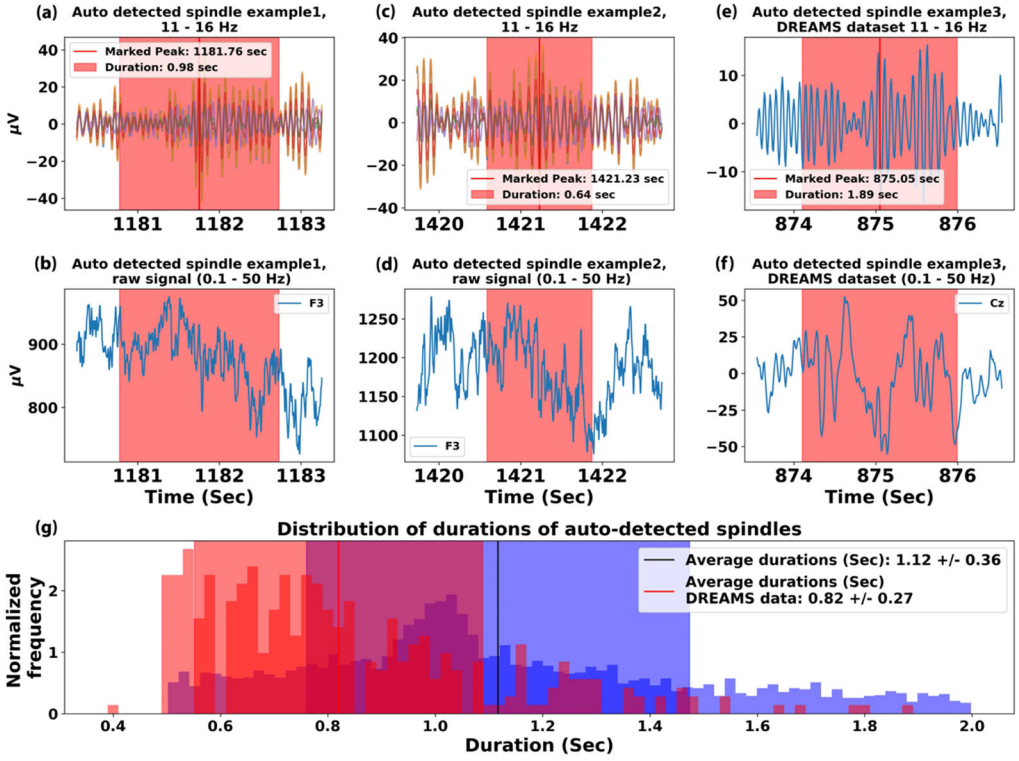

Figure 3. Three examples of sleep spindles detected by the FBT model

Example 1 (**a**, **b**) and 2 (**c**, **d**) are spindles of the dataset of the current study marked by the FBT model, and example 3 (**e**, **f**) is a spindle of the DREAMS dataset marked by the FBT model. Preprocessed raw and filtered signals are shown. The red shaded areas represent the durations of the spindle. The red lines are the peaks of the filtered spindle signals marked by the FBT model. (**g**) shows the distribution of the durations of the auto detected spindles in both our data and the DREAMS dataset. Average duration is shown as mean +/− standard deviation.

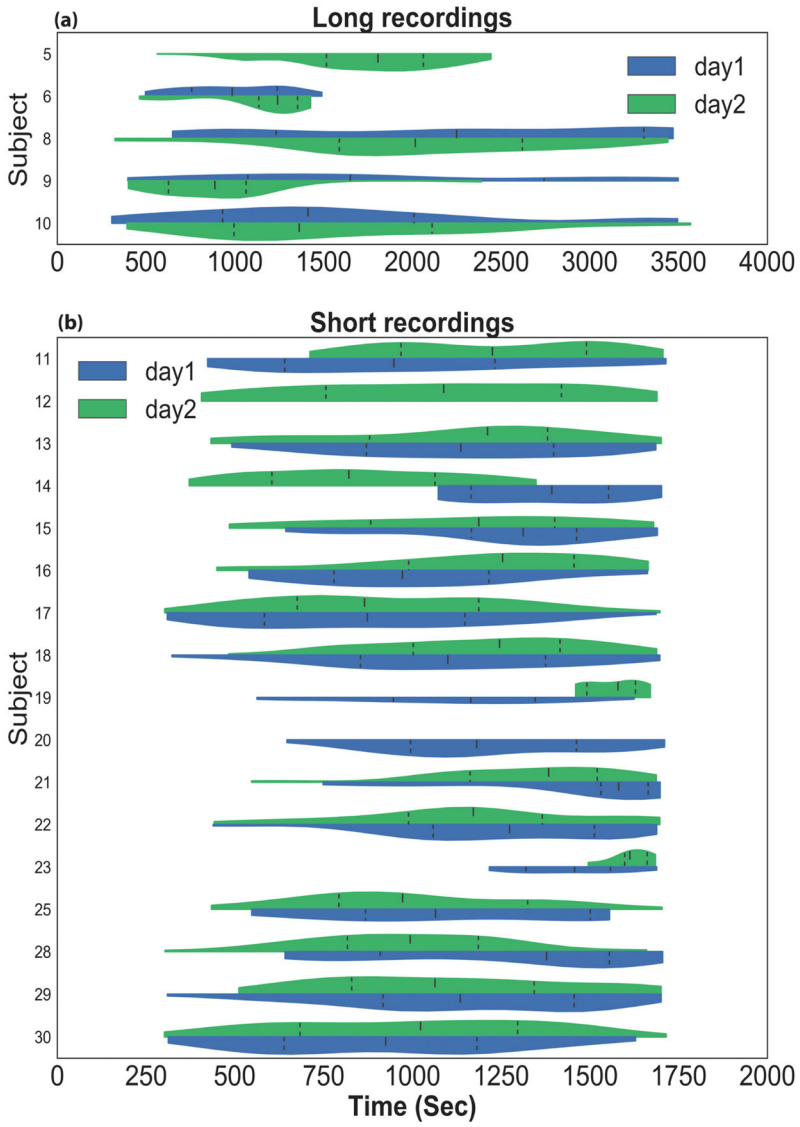

Figure 4. Manual Annotation of Spindles Shown as Spindle Density

Onsets of sleep spindles were marked manually by 2 experts, but no information about the duration of each spindle was annotated. One of the experts annotated data for subjects 5 through 10, which were long recordings that lasted more than 60 minutes (4000 seconds). The other expert annotated data for subjects 11 through 30, which were short recordings that lasted for 30 minutes (2000 seconds). Both days of subject 7 and day 1 of subject 5 were excluded from analysis due to excessive noise. Both days of subject 24, 26, 27 and day 2 of subject 20 were excluded because subjects did not reach to stage 2 sleep during the recordings on those days. The figure shows the temporal density of the annotated sleep spindles. Spindles marked in the first 300 seconds and the last 100 seconds were excluded to avoid false positive raised by artificial patterns. Spindle densities were computed by *Python.Seaborn.violinplot* using a kernel size of 20 samples. Both ends of the density distribution were cut by the first and the last occurrence of a spindle event. The three dashed

lines in one half of a violin plot represent 25%, 50%, and 75% quartile of the density distribution.

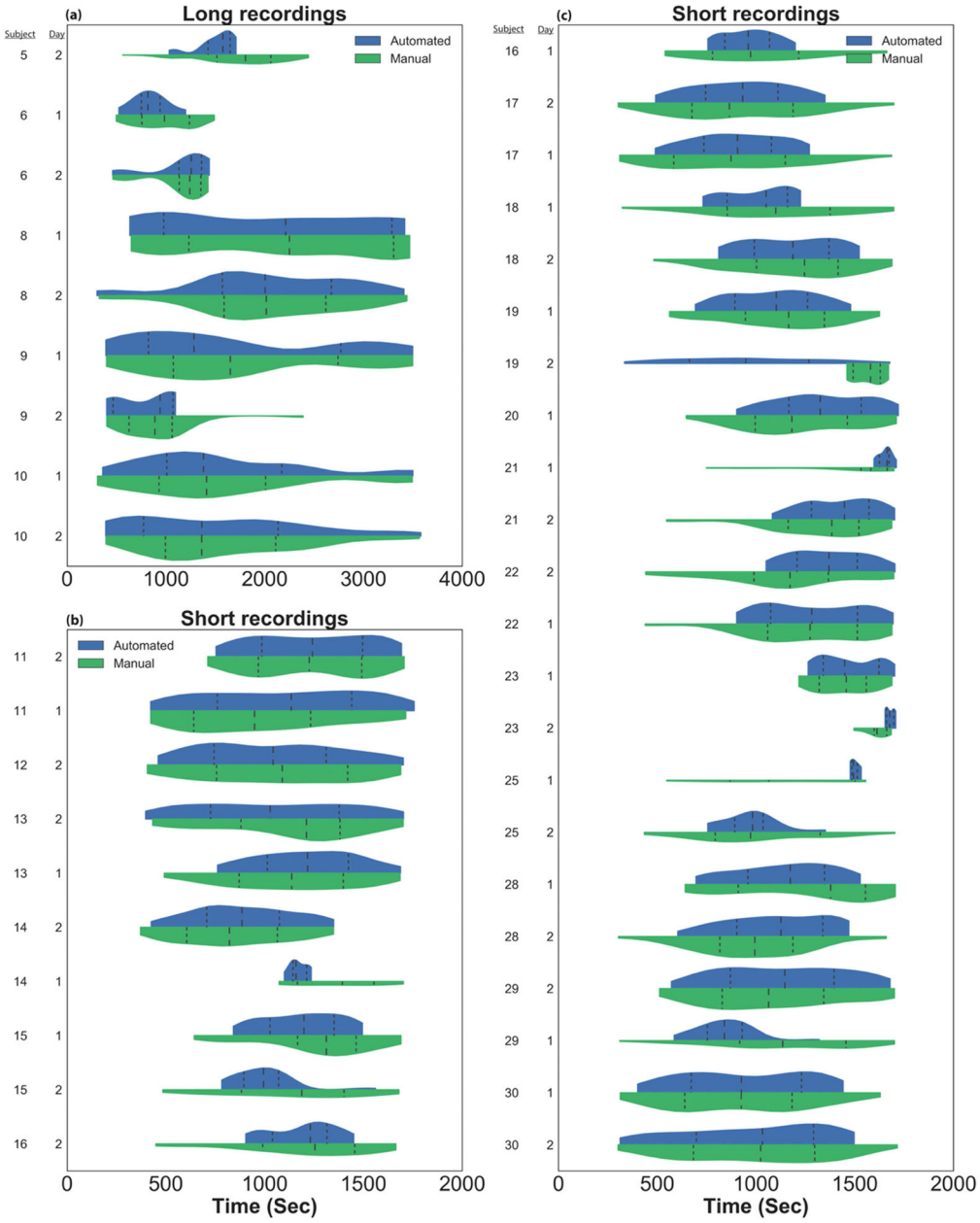

Figure 5. Spindle densities for automated and manual classification

Automated classifications were made by the optimized FBT pipeline. The best lower threshold is 0.4 and higher threshold is 3.5. There was no significant evidence to support that the FBT model performed better for the long recordings than the short recordings (permutation test, p values = 0.245 +/− 0.018, M +/− SD). On the other hand, there was significant evidence to suggest that the machine learning model performed better for the long recordings than the short recordings (permutation test, p values = 10^{-4} +/−10^{-4}, M +/− SD). Spindle density was computed by *Python.Seaborn.violinplot* with a kernel size of 20 samples. In general, the FBT pipeline classified fewer spindles than human annotators.

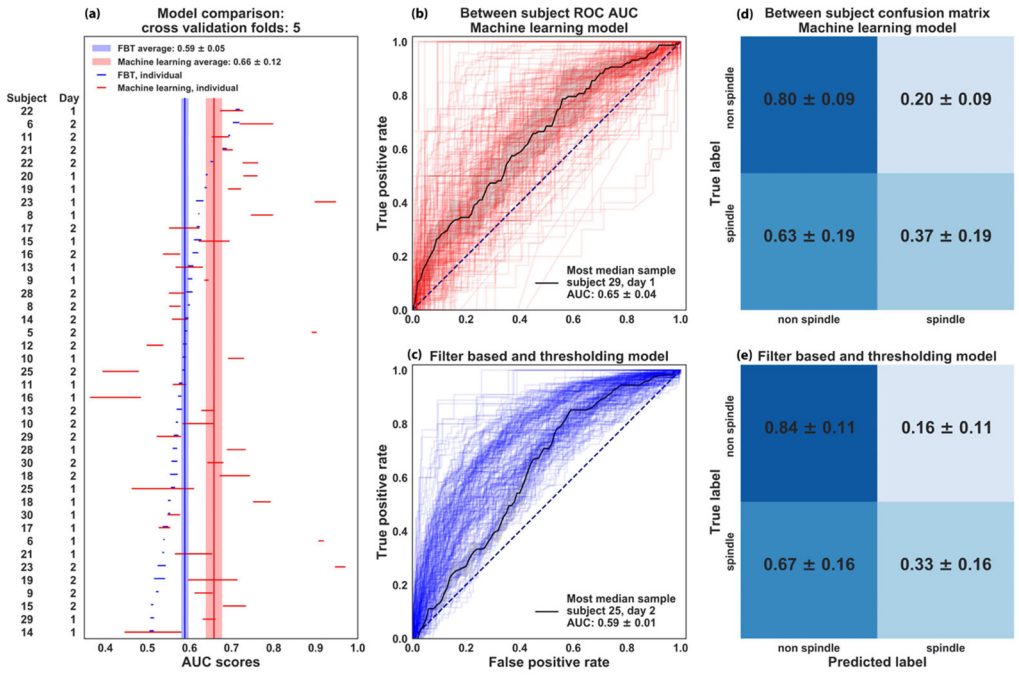

Figure 6. Validation Results

A comparison is shown between the best machine learning pipeline (nested voting classifier and decision tree classifier as feature selection component and a gradient boost classifier as an estimator, using naïve features) and the FBT model at individual level. **(a)** shows the cross-validation results of applying the machine learning pipeline and the FBT model to each subject, with standard errors across subjects shaded in corresponding colors (red for machine learning and blue for FBT). The machine learning pipeline determined by the *TPOT* library increased the within-subject AUC from 0.59 to 0.67, which was statistically significant using a two-sample t test. Variations within each individual captured the randomized selection of a training subset from all possible samples. While the FBT model contained strict criteria and was less flexible than the machine learning pipeline in terms of probabilistic decision functions, the AUCs of the machine learning pipeline showed higher variations than the FBT model within each individual. **(b)** and **(c)** show all the cross-validation ROC curves, and the median sample subject is shown by average ROC and corresponding standard error. The difference between **(b)** and **(c)** also reflects the difference of variances between two models. **(d)** and **(e)** show the normalized confusion matrix across subjects. Two models have similar TP rate and TN rate on average.

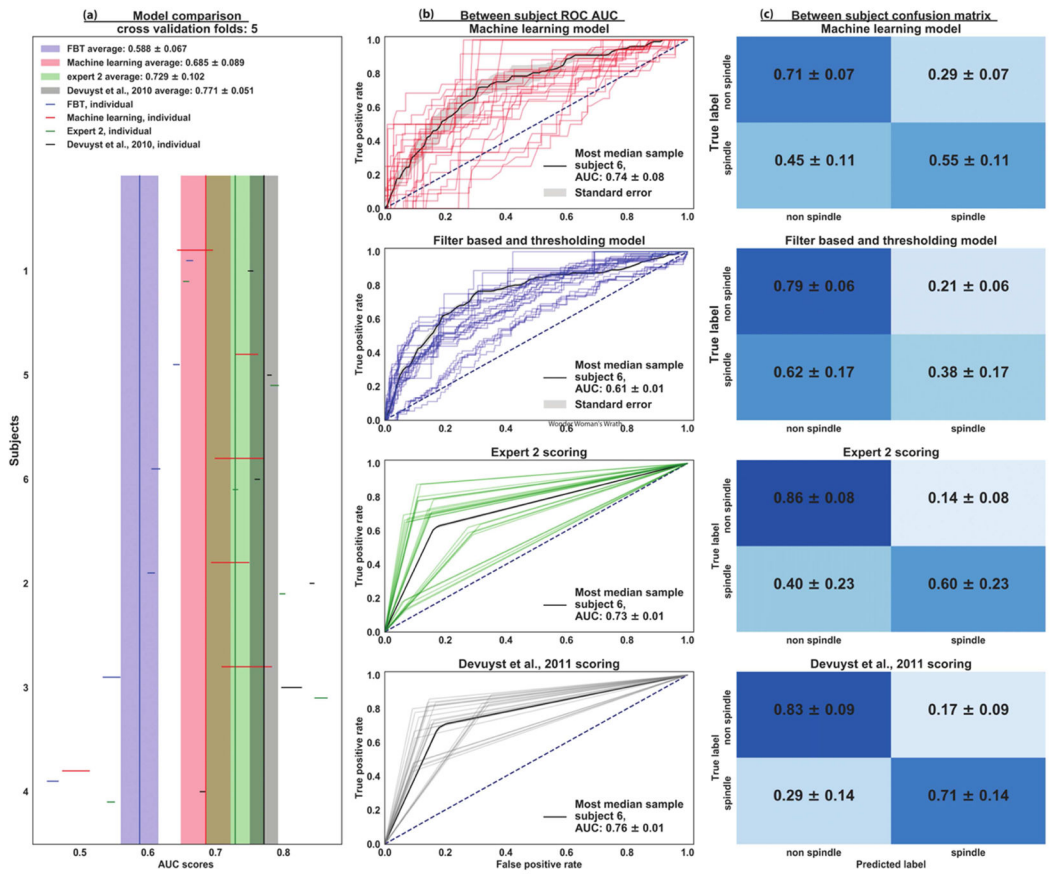

Figure 7. Applying the pipeline to the DREAMS project data

A comparison of cross-validation performance among the FBT model, the machine learning pipeline, the second expert who annotated spindles independently from the first expert, and Deyuyst et al. [1]. In **(a)** between subject standard errors are shaded in corresponding colors (purple for FBT, red for machine learning, green for the second expert, and grey for Deyuyst et al.). The FBT model set a benchmark and all other methods outperformed the FBT model. In **(b)** we show the ROC curves of all the cross-validation results. For the machine learning pipeline and the FBT model, probabilistic predictions were generated by the decision functions. Thus, the ROC curves show a step-wise trend. For the second expert and Deyuyst et al. no probabilistic estimate was made because data was derived from time stamps. The median cross-validation individuals are highlighted to represent the model performance estimate. In **(c)** from top to bottom and from left to right, the cells in the confusion matrices are TN rate, FN rate, FP rate, and TP rate. The confusion matrices suggested the machine learning pipeline and the second expert highly agreed on this dataset, but the second expert has higher TP rate and TN rate. The FBT model suffers from a high false negative rate but has a high TN rate. Devuyst et al. has both a high TP and TN rate.

Table 1

Configuration of the system and versions of python libraries used in the following method section.

Property	Description	Property	Description
Compiler	MSC v.1900 64 bit	System	Windows
Release	7	Machine	AMD 64
Processor	Intel64 6-M-58-9	CPU cores	8
CPython	3.5.3	IPython	6.0.0
Numpy	1.12.1	Pandas	0.20.1
MNE	0.14	Seaborn	0.7.1
Scipy	0.19.0	Matplotlib	2.0.1
TPOT	0.8		