# Automatic Sleep Staging Using State Machine-controlled Decision Trees

Syed Anas Imtiaz and Esther Rodriguez-Villegas

*Abstract*— **Automatic sleep staging from a reduced number of channels is desirable to save time, reduce costs and make sleep monitoring more accessible by providing home-based polysomnography. This paper introduces a novel algorithm for automatic scoring of sleep stages using a combination of small decision trees driven by a state machine. The algorithm uses two channels of EEG for feature extraction and has a state machine that selects a suitable decision tree for classification based on the prevailing sleep stage. Its performance has been evaluated using the complete dataset of 61 recordings from PhysioNet Sleep EDF Expanded database achieving an overall accuracy of 82% and 79% on training and test sets respectively. The algorithm has been developed with a very small number of decision tree nodes that are active at any given time making it suitable for use in resource-constrained wearable systems.**

## I. Introduction

Sleep studies involve monitoring and analysis of physiological signals from brain (EEG), eyes (EOG) and muscle movement (EMG) followed by their classification in to one of the five stages of sleep. These stages, based on AASM classification [1] are, Wake, N1, N2, N3 and REM. Analysis of these overnight recordings and their classification is a tedious task [2] making their automation highly desirable. This would not only save time and costs associated with sleep testing but also make the tests more accessible to a larger population. Further, this also helps to reduce inter-rater disagreement [3] as well as subjective error in human scoring and improve consistency between different tests. These motivations lead to the research and development of algorithms for automatic scoring of sleep stages.

Automatic sleep scoring is an increasingly active area of research with many algorithms already published in literature. A typical algorithm involves using some kind of signal processing to extract representative features followed by a classifier to assign one of the sleep stages based on these features. Some of the algorithms proposed recently include the use of support vector machines [4]–[6], hidden Markov models [7] and frequency coupling with linear discriminant analysis classification [8]. Other methods have also used artificial neural networks [2], [9], [10] and decision trees [11], [12] for classification with a variety of time, frequency, entropy [13], [14] and wavelet [10] based features.

All sleep scoring algorithms in literature report their classification performance by evaluating their method on sleep data either recorded by the researchers or using signals available from public sleep databases. It is generally difficult to compare the performance of algorithms that have been evaluated using different databases. However, the use of public databases helps in comparison since that is easily accessible for all researchers.

Until recently, the most popular sleep database has been the PhysioNet Sleep EDF database [15] which comprises of 8 overnight sleep recordings. A superset of this database, the Sleep EDF Expanded database [16], is now available with 61 sleep recordings. Of the various sleep scoring algorithms mentioned above, only four have reported their results using the new database.

In this paper, we present a novel algorithm for classification of sleep stages using a combination of small decision trees contextually driven by a state machine. This approach is inspired by the combination of state machine and decision trees used in artificial intelligence for game development [17] and results in an overall shorter worst case path for individual trees that are designed based on the current state. We evaluate the algorithm's performance using the entire PhysioNet Sleep EDF Expanded database. Section II describes the database in more detail and discusses the features extracted followed by Section III which explains the proposed sleep staging algorithm. The performance of this algorithm is evaluated in Section IV followed by a discussion on future development and improvements in Section V.

## II. Material and Methods

### A. Database

Recordings from the PhysioNet Sleep EDF Expanded database were used for the training and testing of the algorithm proposed in this paper. This database consists of 61 recordings of which 22 are approximately 8 hour overnight recordings while the other 39 are recordings over a 24-hour period including night sleep. For all recordings in the database, two channels of EEG signals (Fpz-Cz and Pz-Oz) were available, together with EOG and submental EMG. The hypnograms in this database were scored using R&K classification with epoch size of 30 seconds. The relevant sleep data from all recordings were taken as those between the *lights off* and *lights on* times where available, or 15 minutes before the first and 15 minutes after the last scored sleep epoch. The hypnograms were also converted to AASM classification using the recommendations in [18] and only the two EEG channels were used to extract features for the algorithm. From the recordings (as listed on the PhysioNet webpage [16]), starting with the first one, every alternate recording was selected to be part of the training set (making

S. A. Imtiaz and E. Rodriguez-Villegas are with the Circuits and Systems Group, Electrical and Electronic Engineering Department, Imperial College London, United Kingdom. Email: ({anas.imtiaz,e.rodriguez}@imperial.ac.uk).

The research leading to these results has received funding from the European Research Council under the European Community's 7th Framework Programme (FP7/2007-2013) / ERC grant agreement no. 239749.

it a total of 31 recordings) while the remaining 30 recordings were taken as part of the test set.

### B. Features

At the initial stages of development, 66 features were extracted from the two EEG channels to study their effectiveness. To compute the features, each 30-second epoch was divided in to 2-second sub-epochs. The value of any feature for the epoch was then calculated by taking the mean within its fifteen sub-epochs. The set of initial features include absolute and relative powers in the following frequency bands: delta (0.5-4Hz), delta1 (0.5-2Hz), delta2 (2-4Hz), sigma (11-16Hz), beta (16-30Hz), alpha (8-13Hz), alpha1 (8-10Hz), alpha2 (10-13Hz), theta (4-8Hz), gamma (30-40Hz); spectral ratio of powers in these bands; spectral edge frequencies at 95% (SEF95) and 50% (SEF50) in several bands. The difference between SEF95 and SEF50 (known as *SEFd*) in 8-16 Hz frequency band, shown to be highly useful for detection of REM sleep [19] was also included in this set. Further, line length of the signal in 11-16 Hz range was also added since it is useful for sleep spindle detection [20], thereby helpful for scoring N2 and N3 stages.

From this set, the features with redundancies and low discriminatory power were removed using sequential feature selection. As a result, 30 features remained that were used in the algorithm developed in this work. The list of these features is shown in Table I.

**TABLE I:** *List of discriminative features used for the sleep staging algorithm*

| Channel | Features |
|---|---|
| Fpz-Cz | *sigma/beta, beta/delta, delta/alpha, beta/alpha, SEFd(8-16Hz), SEFd(0.5-8Hz), SEF95(0.5-30Hz), SEF50(0.5-8Hz), line length (11-16Hz), rel. delta2, rel. beta, rel. gamma, abs. delta, abs. delta1, abs. delta2, abs. alpha2* |
| Pz-Oz | *sigma/beta, beta/delta, theta/alpha, beta/alpha, SEF95(0.5-30Hz), SEF50(0.5-8Hz), rel. beta, rel. gamma, rel. alpha, rel. theta, abs. delta, abs. delta1, abs. alpha1* |

*rel - relative power; abs - absolute power.*

## III. Sleep Staging Algorithm

The algorithm is designed in such a way that the state machine starts with a pre-defined initial state and must satisfy two levels of checks to transition in to another state. The first level is the *core test* which is a *one-versus-all* decision tree with a maximum of seven nodes in total (four nodes in the longest path). It checks to determine whether the epoch being analysed is of the same sleep stage as previous or not. In other words it checks whether the current state of the machine needs to change. If the core test determines that the current epoch may potentially be of a different sleep stage then a series of *peripheral tests* are applied, otherwise the state machine remains unchanged. These peripheral tests are very small *one-versus-one* decision trees with a maximum of two levels and three decision nodes (two nodes in the longest path). Since there are only five possible sleep states including the current state, there can always be a maximum of four peripheral tests required. The order of these peripheral tests are important and determined during the training stage. If one of these tests is passed, the sleep stage corresponding to that test is assigned to the current epoch, no further peripheral tests are executed and the state machine transitions to the new state. If, however, the peripheral tests also fail to assign a different sleep stage to the current epoch, the state of the machine remains unchanged and the previous stage is assigned to the epoch. The pseudocode of the complete algorithm is shown in Listing 1.

**Listing 1:** *Pseudocode of the sleep staging algorithm*

**Initial Condition:** $current\_state$ is $Wake$

**if** $current\_state = Wake$ **then**
  **if** $CoreTest(Wake, Others) = Wake$ **then**
    $current\_state = Wake$
  **else**
    **if** $PeriTest(Wake, N2) = N2$ **then**
      $current\_state = N2$
    **else if** $PeriTest(Wake, N1) = N1$ **then**
      $current\_state = N1$
    **else if** $PeriTest(Wake, N3) = N3$ **then**
      $current\_state = N3$
    **else if** $PeriTest(Wake, REM) = REM$ **then**
      $current\_state = REM$
    **else**
      $current\_state = Wake$
    **end if**
  **end if**
**else if** $current\_state = N1$ **then**
  **if** $CoreTest(N1, Others) = N1$ **then**
    $current\_state = N1$
  **else**
    **if** $PeriTest(Wake, N1) = N1$ **then**
      $current\_state = N1$
    **else if** $PeriTest(N1, N2) = N2$ **then**
      $current\_state = N2$
    **else if** $PeriTest(N1, N3) = N3$ **then**
      $current\_state = N3$
    **else if** $PeriTest(N1, REM) = REM$ **then**
      $current\_state = REM$
    **else**
      $current\_state = N1$
    **end if**
  **end if**
**else if** $current\_state = N2$ **then**
  **if** $CoreTest(N2, Others) = N2$ **then**
    $current\_state = N2$
  **else**
    **if** $PeriTest(N2, N3) = N3$ **then**
      $current\_state = N3$
    **else if** $PeriTest(N1, N2) = N1$ **then**
      $current\_state = N1$
    **else if** $PeriTest(N2, REM) = REM$ **then**
      $current\_state = REM$
    **else if** $PeriTest(Wake, N2) = Wake$ **then**
      $current\_state = Wake$
    **else**
      $current\_state = N2$
    **end if**
  **end if**

```
else if current_state = N3 then
    if CoreTest(N3, Others) = N3 then
        current_state = N3
    else
        if PeriTest(N1, N3) = N1 then
            if PeriTest(Wake, N1) = N1 then
                current_state = N1
            else
                current_state = Wake
            end if
        else if PeriTest(N2, N3) = N2 then
            current_state = N2
        else if PeriTest(N3, REM) = REM then
            current_state = REM
        else if PeriTest(Wake, N3) = Wake then
            current_state = Wake
        else
            current_state = N3
        end if
    end if
else if current_state = REM then
    if CoreTest(REM, Others) = REM then
        current_state = REM
    else
        if PeriTest(Wake, REM) = Wake then
            if PeriTest(Wake, N2) = Wake then
                current_state = Wake
            else
                current_state = N2
            end if
        else if PeriTest(N2, REM) = N2 then
            current_state = N2
        else if PeriTest(N1, REM) = N1 then
            current_state = N1
        else if PeriTest(N3, REM) = N3 then
            current_state = N3
        else
            current_state = REM
        end if
    end if
end if
```

The algorithm starts initially with the state machine in the Wake state. For an incoming new epoch, the *CoreTest(Wake,Others)* determines whether a state change is required. If yes, then a series of four peripheral tests are used to determine the new state of the machine. For the Wake state, the peripheral checks are: Wake vs N1, Wake vs N2, Wake vs N3 and Wake vs REM. If one of these determine the epoch to be other than Wake then the epoch is assigned that sleep stage and the machine transitions to that state, otherwise the state remains unchanged. If the state of the machine is changed, then based on the newly assigned state, the next epoch will be classified by starting at a different core test following a similar pattern of peripheral tests.

Two exception are made to an otherwise symmetrical structure of the algorithm. First, if the state machine is currently in N3 state and its *PeriTest(N1,N3)* determines the next state to be N1, then a further *PeriTest(Wake,N1)* is used to filter out possible false N1 classifications from this peripheral test. Second, during the REM state, if *PeriTest(Wake,REM)* determines the next state as Wake, another *PeriTest(Wake,N2)* is used to reduce false Wake classifications. These two additional peripheral tests were found to be useful in improving the classification accuracy during the training stage.

## IV. Results

The performance of the proposed algorithm was evaluated using the sensitivity and selectivity metrics defined in [18]. The training set was first used to determine the optimum core and peripheral decision trees and their internal order of evaluation that would result in the highest f-score. Of the 29499 epochs in this set, the best performance of the algorithm resulted in 24255 epochs being correctly classified giving an overall accuracy of 82.22%. The detection performance for all stages (except N1) showed a sensitivity of more than 80% and is shown in Table II.

**TABLE II:** *Algorithm performance using the Training data set*

| | | ALGORITHM | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **W** | **N1** | **N2** | **N3** | **R** | **Sen(%)** | **Sel(%)** |
| **REFERENCE** | **W** | 3290 | 209 | 162 | 26 | 217 | 84.3 | 84.6 |
| | **N1** | 339 | 676 | 519 | 10 | 724 | 29.8 | 68.0 |
| | **N2** | 74 | 54 | 11672 | 592 | 792 | 88.5 | 85.8 |
| | **N3** | 14 | 13 | 759 | 3633 | 21 | 81.8 | 85.1 |
| | **R** | 174 | 42 | 495 | 8 | 4984 | 87.4 | 74.0 |

The test dataset consisted of 29817 epochs in total of which 23512 were correctly classified by the algorithm with an overall accuracy of 78.85%. This is, expectedly, slightly lower than the accuracy obtained using the training set. The results for each sleep stage are shown in Table III. Comparing the sensitivity of each sleep stage with that obtained using the training set, it can be seen that the accuracies for stages N2, N3 and REM are very similar. However, there is a noticeable reduction in the sensitivity for Wake stage and a slight reduction in N1 accuracy as well.

**TABLE III:** *Algorithm performance using the Test data set*

| | | ALGORITHM | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **W** | **N1** | **N2** | **N3** | **R** | **Sen(%)** | **Sel(%)** |
| **REFERENCE** | **W** | 2134 | 304 | 175 | 24 | 321 | 72.1 | 72.1 |
| | **N1** | 441 | 558 | 720 | 16 | 803 | 22.0 | 45.3 |
| | **N2** | 126 | 242 | 12241 | 545 | 732 | 88.2 | 84.0 |
| | **N3** | 32 | 13 | 812 | 3418 | 21 | 79.6 | 85.2 |
| | **R** | 227 | 115 | 629 | 7 | 5161 | 84.1 | 73.3 |

## V. Discussion & Conclusion

A novel algorithm for automatic sleep scoring is presented in this paper using a state machine that is contextually driven by small decision trees to determine the next sleep stage. Its performance has been evaluated using sleep recordings from PhysioNet Sleep EDF Expanded database and is the first algorithm to utilise all the recordings from this database. The results in Section IV show that the algorithm classified N2, N3 and REM stages with a high accuracy. However, the sensitivity for Wake stage in test set fell to about 72% while N1 stage showed poor detection in both test and training subjects. From Table III, it can be seen that most N1 epochs are misclassified as Wake, REM and N2 while substantial

number of Wake epochs are also falsely classified as N1 and REM. This is not unexpected since certain spectral similarities between these three stages are well documented [21], [22].

Only four other methods have used the Sleep EDF Expanded database for performance evaluation. Of these, Yaghouby et al. [7] obtained similar results to this method but used only a subset of the complete database ($ST$ subjects). Sanders et al. [8] also used only the $ST$ subjects and reported an overall accuracy of 75%, which is lower than that obtained in this work. Rodriguez-Sotelo et al. [14] used both $SC$ and $ST$ subjects. For $SC$ subjects, they reported a maximum accuracy of 80% for individual test subjects separately. However, this dropped to 51% when these subjects were combined. They also used $ST$ subjects for validation separately which resulted in a lower accuracy. Finally, Aboalayon et al. [4] also used a subset of this database however their method only discriminated between Wake and N1, and not all the stages of sleep.

The number of epochs classified by each decision tree was also looked at in detail. Of the 29817 test epochs, 84.8% were classified by the core decision trees. This shows that these core tests bear most of the classification load. The peripheral trees are responsible for state transitions and become involved mostly when the sleep stages are at the boundary between two stages. Consequently, even though the peripheral trees are required fewer times their misclassification represents a higher cost since the state machine could potentially be transitioned to a wrong state leading to further misclassifications. At present, all the binary peripheral decision trees are designed with equal misclassification cost for either stage. For example, the peripheral decision tree corresponding to N1 vs N2 classification is the same whether the current state is N1 or N2. Initial work involving training of trees with different misclassification cost based on the current state has shown promising results with improved classification accuracy. This will be pursued as future research work to improve the current algorithm.

The core decision trees are constrained to have a maximum of four nodes while the peripheral trees have a maximum of two nodes in their longest path. Although, this limits the maximum accuracy that can be achieved, it was done to realise an algorithm with smaller processing requirements making it suitable for being used in a wearable environment where limited processing resources are available.

Overall the results in this paper suggest that the approach of combining state machines and decision trees in the context of sleep staging can be highly useful for the classification of sleep. This approach allows for the use of multiple small decision trees that get activated depending on the current sleep stage. It also results in better usage of processor resources on which the algorithm is run. This is because only a subset of features are computed each time depending on the current sleep stage. Further, since the starting decision trees change based on the current state, not all of them are required at all times. This saves several nodes of comparison that would have been required in an approach using conventional decision trees alone. Although the algorithm showed a good overall performance, sensitivity in N1 stage was found to be lacking. Nevertheless, we believe that the approach presented in this paper will be highly useful for designers of automatic sleep scoring systems and can be further improved with the use of more discriminative features and better designed decision trees.

## REFERENCES


[1] C. Iber, S. Ancoli-Israel, A. Chesson, and S. Quan, Eds., *The AASM manual for the scoring of sleep and associated events: rules, terminology and technical specifications*. Westchester, IL: American Academy of Sleep Medicine, 2007.

[2] M. Ronzhina *et al.*, "Sleep scoring using artificial neural networks," *Sleep Med. Rev.*, vol. 16, no. 3, pp. 251–63, 2012.

[3] H. Danker-Hopfe *et al.*, "Interrater reliability for sleep scoring according to the rechtschaffen & kales and the new aasm standard," *J. Sleep Res.*, vol. 18, no. 1, pp. 74–84, 2009.

[4] K. Aboalayon, H. Ocbagabir, and M. Faezipour, "Efficient sleep stage classification based on eeg signals," in *IEEE LISAT*, New York, May 2014.

[5] T. Lajnef *et al.*, "Learning machines and sleeping brains: Automatic sleep stage classification using decision-tree multi-class support vector machines," *J. Neurosci. Methods*, no. 0, 2015.

[6] T. Sousa *et al.*, "A two-step automatic sleep stage classification method with dubious range detection," *Comput. Biol. Med.*, vol. 59, no. 1, pp. 42–53, 2015.

[7] F. Yaghouby, P. Modur, and S. Sunderam, "Naive scoring of human sleep based on a hidden markov model of the electroencephalogram," in *IEEE EMBC*, Chicago, August 2014.

[8] T. Sanders, M. McCurry, and M. Clements, "Sleep stage classification with cross frequency coupling," in *IEEE EMBC*, Chicago, August 2014.

[9] S. Charbonnier, L. Zoubek, S. Lesecq, and F. Chapotot, "Self-evaluated automatic classifier as a decision-support tool for sleep/wake staging," *Comput. Biol. Med.*, vol. 41, no. 6, pp. 380–9, 2011.

[10] 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 *IEEE EMBC*, Vancouver, August 2008.

[11] S.-F. Liang, C.-E. Kuo, Y.-H. Hu, and Y.-S. Cheng, "A rule-based automatic sleep staging method," *J. Neurosci. Methods*, vol. 205, no. 1, pp. 169–76, 2012.

[12] M. Hanaoka, M. Kobayashi, and H. Yamazaki, "Automated sleep stage scoring by decision tree learning," in *IEEE EMBC*, Chicago, July 2000.

[13] S.-F. Liang *et al.*, "Automatic stage scoring of single-channel sleep eeg by using multiscale entropy and autoregressive models," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 6, pp. 1649–1657, 2012.

[14] J. L. Rodriguez-Sotelo *et al.*, "Automatic sleep stages classification using eeg entropy features and unsupervised pattern analysis techniques," *Entropy*, vol. 16, no. 12, pp. 6573–6589, 2014.

[15] PhysioNet. (2013) Sleep-edf database. [Online]. Available: http://www.physionet.org/physiobank/database/sleep-edf/.

[16] PhysioNet. (2014) Sleep-edf database [expanded]. [Online]. Available: http://www.physionet.org/physiobank/database/sleep-edfx/.

[17] I. Millington and J. Funge, Eds., *Artificial Intelligence for Games*. CRC Press, 2009.

[18] S. Imtiaz and E. Rodriguez-Villegas, "Recommendations for performance assessment of automatic sleep staging algorithms," in *IEEE EMBC*, Chicago, August 2014.

[19] S. Imtiaz and E. Rodriguez-Villegas, "A low computational cost algorithm for rem sleep detection using single channel eeg," *Ann. Biomed. Eng.*, vol. 42, no. 11, pp. 2344–2359, 2014.

[20] S. Imtiaz and E. Rodriguez-Villegas, "Evaluating the use of line length for automatic sleep spindle detection," in *IEEE EMBC*, Chicago, August 2014.

[21] M. Corsi-Cabrera, Z. Munoz-Torres, Y. delRio-Portilla, and M. A. Guevara, "Power and coherent oscillations distinguish rem sleep, stage 1 and wakefulness," *Int. J. Psychophysiol.*, vol. 60, no. 1, pp. 59–66, 2006.

[22] R. Bódizs, M. Sverteczki, and E. Mészáros, "Wakefulness-sleep transition: Emerging electroencephalographic similarities with the rapid eye movement phase," *Brain Res. Bull.*, vol. 76, no. 1, pp. 85–89, 2008.