Detecting Sleep States from Accelerometer Data using Deep Learning

Alvási fázisok érzékelése gyorsulásmérő adatok alapján mélytanulás használatával

Ádám M. Biró
biro.adam@edu.bme.hu

Bálint Szalay
balint.szalay@edu.bme.hu

Zsombor Szenyán
zsomborszenyan@edu.bme.hu

Budapest University of Technology and Economics
Budapest, Hungary

NEM VÉGLEGES VÁLTOZAT!

Abstract

Monitoring sleep patterns is essential when researching sleep and sleep-related problems. Automatically analyzing wrist-worn accelerometer data could emerge as a low-cost and practical method for logging sleep periods when conducting large-scale studies. However, traditional rule-based methods have limited accuracy in detecting sleep onset and wakeup events. In this paper we explore ways how deep learning, a powerful machine learning technique could be used to detect sleep states more precisely. We demonstrate two possible approaches to solve this problem and compare the results.

Az alvási minták monitorozása elengedhetetlen fontosságú az alvással kapcsolatos kutatások során. A csuklón hordható gyorsulásmérő adatok automatikus elemzése egy olcsón és egyszerű megoldás lehet az alvási ciklusok rögzítésére nagy mintájú kutatások során. Azonban a hagyományos szabályalapú módszerek nem tudják megfelelő pontossággal meghatározni az elalvás és felébredés időpontját. Ebben a dokumentumban megvizsgálunk lehetséges módszereket arra, hogy a mélytanulást, a gépi tanulás egyik formáját használjuk fel az alvási állapot pontosabb meghatározására. Megmutatunk két lehetséges megközelítést, és összevetjük az eredményeket.

1 Introduction

Fulfilling sleep plays a crucial role in maintaining physical and mental well–being. Yet a great number of people suffer from problems which negatively affect their sleep patterns. Inadequate quality and quantity of sleep is associated with various health problems, including depression and cardiovascular diseases.

Preprint. Under review.

Monitoring sleep patterns is essential when researchers conduct studies on sleep. However, detecting the onset and end of sleep phases is a challenging task. Invasive sleep monitoring methods, such as polysomnography (PSG) are expensive and unsuitable for large–scale studies.

In recent years, wearable devices equipped with accelerometers have emerged as a possible tool for non–invasive sleep monitoring. Traditional techniques which use human–engineered features achieve moderate success in recognizing sleep onset and wakeup events from accelerometer data due to the nuanced transition between sleep and awake states.

A Kaggle competition was hosted by the Child Mind Institute [3] with the goal of finding new approaches to sleep state detection from accelerometer data utilizing the latest advancements in the field of data science.

In this paper we explore the ways how deep learning, a promising machine learning method could be used to solve the task, and also document our models which we created as an entry to the Kaggle competition. Our work also serves as a homework project for the course Deep Learning in Practice at Budapest University of Technology and Economics.[1]

2 Dataset

For our model, we used the dataset provided in the Kaggle competition [3]. The training dataset contained about 270 recordings of wrist-worn accelerometer data, annotated with two event types: onset, the beginning of sleep and wake up, the end of a sleep phase.

The accelerometer recordings are time series data which contain two features: the *z-angle* and *ENMO*. The z-angle is a metric derived from individual accelerometer components that is commonly used in sleep detection, and refers to the angle of the arm relative to the vertical axis of the body. ENMO is the Euclidean Norm Minus One of all accelerometer signals, with negative values rounded to zero. The time series have a resolution of five seconds intervals between data points.

The data was labeled by sleep experts. The following policy was used to label sleep events according to the Kaggle competition description [2]:

> "While sleep logbooks remain the gold-standard, when working with accelerometer data we refer to sleep as the longest single period of inactivity while the watch is being worn. For this data, we have guided raters with several concrete instructions:
>
> - A single sleep period must be at least 30 minutes in length A single sleep period can be interrupted by bouts of activity that do not exceed 30 consecutive minutes
> - No sleep windows can be detected unless the watch is deemed to be worn for the duration (elaborated upon, below) The longest sleep window during the night is the only one which is recorded
> - If no valid sleep window is identifiable, neither an onset nor a wakeup event is recorded for that night.
> - Sleep events do not need to straddle the day-line, and therefore there is no hard rule defining how many may occur within a given period. However, no more than one window should be assigned per night. For example, it is valid for an individual to have a sleep window from 01h00-06h00 and 19h00-23h30 in the same calendar day, though assigned to consecutive nights
> - There are roughly as many nights recorded for a series as there are 24-hour periods in that series."

Figure 2. shows an example plot from the input data, a portion of a time series labeled with sleep states.

[1] Deep Learning in Practice with Python and LUA. `http://smartlab.tmit.bme.hu/oktatas-deep-learning`

[2] `https://www.kaggle.com/competitions/child-mind-institute-detect-sleep-states/data`

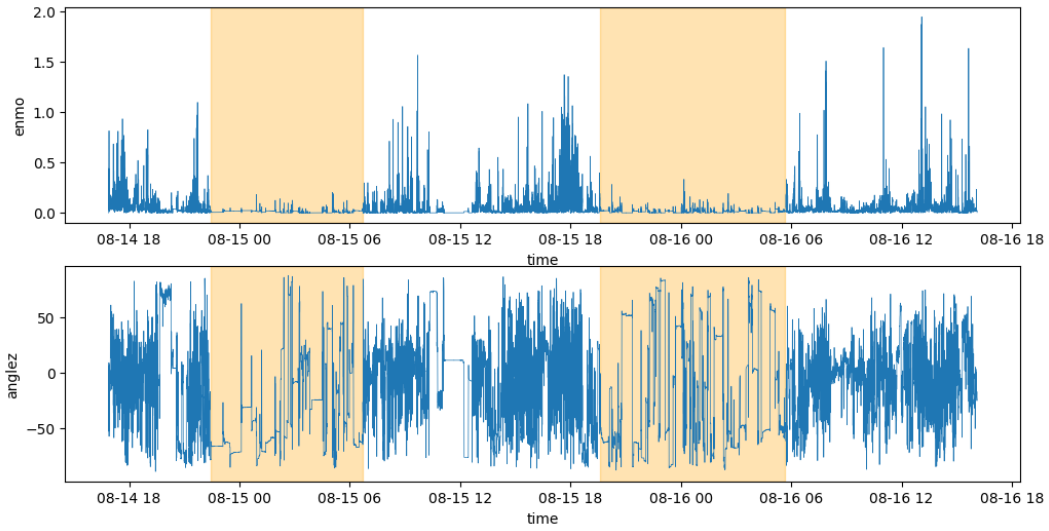

Figure 1: A roughly two–day period of a time series data, annotated with sleep states. The upper diagram shows the ENMO signal and the bottom diagram shows the Z–angle signal. The highlighted intervals are labeled as sleep periods.

3 Exploring possible methods

We began our work by exploring existing work related to the subject. Loh et al. [5] and Malafeev et al. [6] showed that deep neural networks can be utilized for classification of sleep stages based on brainwave signal time series, such as polysomnography (PSG) recordings. Roberts et al. [7] showed that motion sensor data from consumer-grade wearables could possibly be used for sleep stage classification, as there is correlation between motion sensor data and PSG signal data. This led us to believe that deep learning could be successfully used on accelerometer data to detect sleep events. However, the previous works focused on classifying the sleep stage based on an fixed-length window of time series sequences, rather than providing a method to accurately find the boundaries of these stages in an series of arbitrary length.

Typically, the output of a neural network is of a fixed-length. But in our case, the number of sleep onset and wakeup is variable, so we had to design a solution that can produce variable-length output. A straightforward approach would be to divide the time series into smaller segments, as there is at most one sleep period recorded per night. However, this is problematic because there is no hard definition on the boundaries of a night, and the length of a sleep period can vary significantly.

We tried two different approaches to solve this problem: a change point detection based approach and a momentary sleep state classification approach. Below we present the basic principle of these two approaches.

4 Approach 1: Change point detection (CPD)

4.1 Overview

Change points are points in a time series that mark the boundaries of certain states. In our problem, sleep and awake phases can be viewed as states and the change points represent onset and wake up events. Various methods exist to detect change points using classical machine learning. See Aminikhanghahi et al. [1] for a survey of possible methods. Some works have examined the possibility of using deep neural networks for change point detection, notably Li et al. [4] and Ebrahimzadeh et al. [2]. We used an approach similar to those that are outlined in these works.

Firstly, we trained a classifier deep neural network to recognize if there is a change point in a given segment of the sensor data series. The input is a fixed-length segment of the sensor data, which we call a *frame*. The output is a one-hot encoded representation of either one of the following three

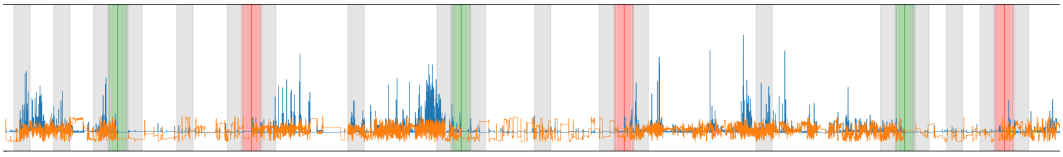

Figure 2: A segment of a training sensor series marked with the sampled frames. Grey, green and red frames show sampled frames labeled as *no event*, *onset*, and *wake up* respectively.

classes: *no event*, *onset*, *wake up*. Then we used a sliding window technique to detect onset and wake up events during the prediction. We employed simple algorithmic methods to merge events that were detected multiple times and filter out sleep periods that are too short or should not be recorded according to the policy described in section 2.

We further refer to this approach as *CPD*.

4.2 Data preprocessing

We used two features as an input from the time series: ENMO and Z-angle. We standardize the two signals to have zero mean and one standard deviation. Then we sampled frames around ground-truth onset events, wake up events and in between events with the appropriate labels. Figure 2. shows a segment of a training series marked with sampled and labeled frames. Figure 3. shows example frames that were assigned to each classes. The frame length was set to 60 minutes. Longer frame lengths improved the classification performance, but made the localization of events less accurate as many frames overlapped with one event.

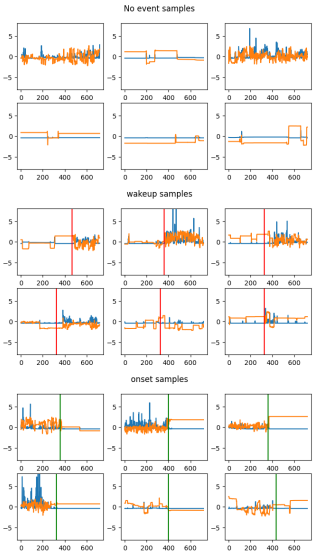

Figure 3: Training samples in each class extracted from the training data.

4.3 Model architecture

We used a multilayered artificial neural network with one-dimensional convolutional layers. Convolutional layers are ideal for extracting features in an input which is indifferent to translation. The model consists of three convolutional layers with additional subsampling layers, and three fully-connected layers. The output layer has softmax activation because the output is an one-hot encoded representation of the predicted class. We used regularization to counter overfitting. The detailed architecture of the model can be seen on figure 4.3. We also tried a recurrent architecture, but there was no improvement in the classification performance.

4.4 Training

For the training we used the Adam optimization algorithm and categorical crossentropy as a loss function. The training dataset contained around 25.000 samples in each class. The best achieved validation loss was 0.18 and with 94% classification accuracy.

4.5 Prediction

We used the following algorithm to make predictions using the trained classifier model:

1. Slide a frame-sized window with a stride of 3 minutes. Feed the frame at each window position to the classifier network. If the predicted probability of an onset or wake up event exceeds a certain threshold, record the event accordingly at the center of the window. We got the best results with 0.97 as a threshold.

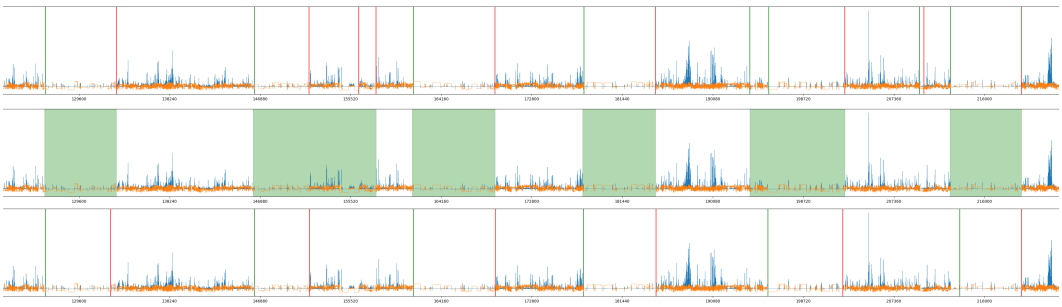

Figure 5: An example of result of the prediction algorithm. The uppermost plot shows the combined predicted events, the middle plot shows the fitted and filtered sleep periods. The bottom plot shows the ground truth events. Red lines represent onset events, green lines represent wake up events. The ENMO and Z-angle signals are colored orange and blue respectively.

2. Combine multiple events of the same type into one event if they are close to each other, which means they probably represent the same ground-truth event. The time step of the new combined event is the average of the combined events.
3. Determine the sleep periods based on the predicted events with by choosing the longest possible sleep periods. This effectively means sleep periods start when there is a predicted onset before a wakeup event and ends when there is a wakeup event before an onset event.
4. Merge sleep periods which have a less than 30 minute long gap between them. This is useful because according to the instructions used in recording the ground-truth events, sleep periods can be interrupted with less than 30-minute long bouts of activity (see section 2).
5. Filter out sleep periods that are shorter than 30 minutes (which are not recorded according to the instructions).
6. Assign each predicted sleep period to a night based on which *midnight point* it its midpoint is the closest to. Midnight point is a designated time of the day, which we set to 3AM.
7. Take only the longest sleep period for each night. This is needed because the rest are not recorded according to the instructions. Furthermore this helps to filter out inactive periods during the day which are not sleeping states.

See figure 5. for a visual example on the algorithm.

5 Approach 2: Momentary state classification (MSC)

5.1 Overview

The other method is based on predicting the asleep/awake state in each time step based on a fixed-length segment of sensor data. In each case, the input was a window of 200 time steps, with the mixed training data capturing sequences where the subject gets up, falls asleep, sleeps or is awake. The target output was also a 200 length sequence, with the values 0 if the person was awake and 1 if the person was asleep at each step.

Next, for each step in the series, we predicted whether the person was sleeping or not at that moment. Following this, we calculated a moving average for the predictions. Finally, we filtered out the points corresponding to sleep/wake and wake/sleep transitions, as these were precisely the events we were seeking. We then searched for the longest sleep event for each day. If it is greater than 30 minutes, it is a suitable sleep period.

We further refer to this approach as *MSC*.

5.2 Data preprocessing

5.3 Model architecture

In the following sentences, we will describe our model. The model input is a Bidirectional LSTM with 200 units. The input to this layer is (200, 3), where 200 refers to the length of a window, and 3 refers to the number of features. We want the full sequence to be returned, so we set the return_sequences flag to true. The next layer is the same Bidirectional LSTM, we return the sequences, as mentioned before. The output layer is a dense layer with an input shape of (200, 400), where 200 refers to the length of the window, and 400 refers to the output of the Bidirectional LSTM. The output dimension of this layer is (200, 1), assigning a probability to every step of the window. If the probability is higher, the person is considered to be sleeping; if it is lower, then they are awake. The model architecture can be seen on figure 5.3.

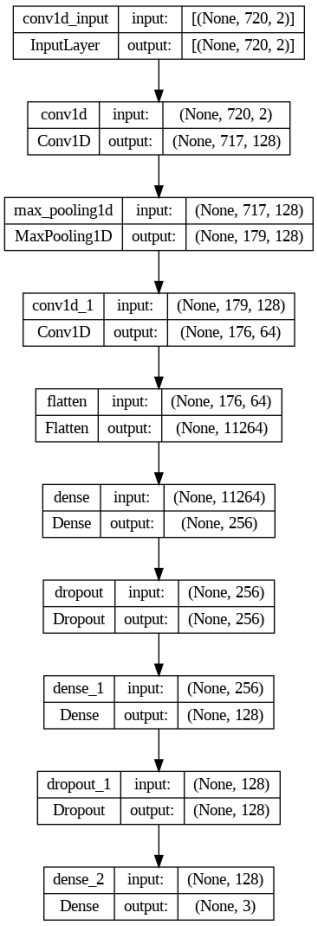

Figure 4: An overview of the neural network of the CPD approach.

5.4 Training

5.5 Prediction

6 Evaluation

For the evaluation of the two approaches, we used the event prediction average precision score metric provided by Kaggle. A description from Kaggle:

> Event Detection Average Precision, an AUCPR metric for event detection in time series and video.
>
> This metric is similar to IOU-threshold average precision metrics commonly used in object detection. For events occuring in time series, we replace the IOU threshold with a time tolerance.
>
> Submissions are evaluated on the average precision of detected events, averaged over timestamp error tolerance thresholds, averaged over event classes.
>
> Detections are matched to ground-truth events within error tolerances, with ambiguities resolved in order of decreasing confidence.

We evaluated the models on 30 randomly selected series which were not used in the training.

6.1 Results

1st approach (CPD) best score: 0.23
2nd approach (momentary classification) best score: 0.33

7 Observations

The CPD based model is capable of succesfully recognizing almost all ground-truth onset and wake up events, and achieves moderate precision in the localization of the events, which significantly detoriates if the frame size is increased. The greatest limiting factor of the model is that it produces many false-positive predictions, which alter the boundaries of the corresponded sleep periods. It is prone to detecting false positive events partly due to the fact that it only sees a short context of a change point. As a result, this model labels all periods with little variation in sensor data as sleep periods, including intervals where the accelerometer was removed or the subject was resting but was

not asleep, which can easily be distinguished from sleep periods when the whole interval is seen. Increasing the frame size and thus giving greater context might allow the model to better differentiate between these cases, but it ruins the localization accuracy as noted before. Perhaps it would be worth considering to combine the change point classification method with a localization regression method.

References

[1] Samaneh Aminikhanghahi and Diane J. Cook. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, 2017.

[2] Zahra Ebrahimzadeh, Min Zheng, Selcuk Karakas, and Samantha Kleinberg. Deep learning for multi-scale changepoint detection in multivariate time series. *CoRR*, abs/1905.06913, 2019.

[3] Nathalia Esper, Maggie Demkin, Ryan Hoolbrok, Yuki Kotani, Larissa Hunt, Andrew Leroux, Vincent van Hees, Vadim Zipunnikov, Kathleen Merikangas, Michael Milham, Alexandre Franco, and Gregory Kiar. Child Mind Institute - Detect Sleep States, 2023.

[4] Jie Li, Paul Fearnhead, Piotr Fryzlewicz, and Tengyao Wang. Automatic change-point detection in time series via deep learning. *arXiv preprint arXiv:2211.03860*, 2022.

[5] Hui Wen Loh, Chui Ping Ooi, Jahmunah Vicnesh, Shu Lih Oh, Oliver Faust, Arkadiusz Gertych, and U. Rajendra Acharya. Automated detection of sleep stages using deep learning techniques: A systematic review of the last decade (2010–2020). *Applied Sciences*, 10(24), 2020.

[6] Alexander Malafeev, Dmitry Laptev, Stefan Bauer, Ximena Omlin, Aleksandra Wierzbicka, Adam Wichniak, Wojciech Jernajczyk, Robert Riener, Joachim Buhmann, and Peter Achermann. Automatic Human Sleep Stage Scoring Using Deep Neural Networks. *Frontiers in Neuroscience*, 12, 2018.

[7] Daniel M Roberts, Margeaux M Schade, Gina M Mathew, Daniel Gartenberg, and Orfeu M Buxton. Detecting sleep using heart rate and motion data from multisensor consumer-grade wearables, relative to wrist actigraphy and polysomnography. *Sleep*, 43(7):zsaa045, 03 2020.

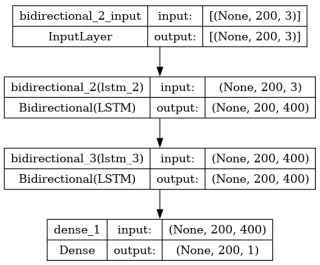

Figure 6: An overview of the neural network of the momentary classification approach.