Sleep Stages Classification using deep neural networks

- Project Proposal -

Gladys ROCH
CentraleSupélec
Gladys.roch@supelec.fr

Abstract

This project proposes to implement a supervised deep learning solution to classify 5 sleep stages: Wake, N1, N2, N3 and Rem (Rapid Eye Mouvement). The classification is be based on electroencephalograms (eeg), pulse and movement recordings. The neural network considered is a Fully Convolutional Network (FCN).

Foreword

My Machine Learning class final project is a Kaggle competition proposed by the company Dreem 2. In this paper, I propose to work on finding a deep learning solution to this challenge.

1. Introduction

Sleep progresses in cycles that involve multiple sleep stages: wake, light sleep (N1, N2), deep sleep (N3), rem sleep (rapid eye movement).

Monitoring sleep stage is beneficial for diagnosing sleep disorders. The gold standard to monitor sleep stages relies on a polysomnography study conducted in a hospital or a sleep lab. Different physiological signals are recorded such as electroencephalogram (eeg), electrocardiogram (ecg) etc. Sleep stage scoring is then performed visually by an expert on epochs of 30 seconds of signals recording. The resulting graph is called a hypnogram. He provides a compact description the night.

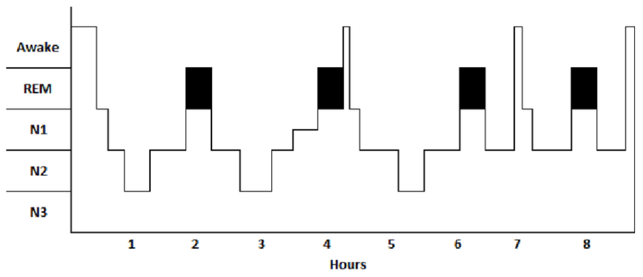

Figure 1. Hypnogram of scored human sleep staging

Dreem 2 is a company that specialize in at-home sleep monitoring. They created a headband that measures brain activity and other biosignals (e.g. pulse, eyes movement) throughout the night. These recordings are then used to classify the sleep stages of the user.

The objective is to develop a deep neural network able to classify five sleep stages (wake, rem, N1, N2, N3) from the biosignals recorded by the Deem 2 headband.

1.1. Datasets

1.1.1 Source and composition

The dataset used is propriety of Dreem 2. It consists of windows of 30 seconds of raw, labelled data. The raw data includes 7 eegs channels in frontal and occipital position, 1 pulse oximeter infrared channel, and 3 accelerometers channels (x, y and z).

1.1.2 Dimensions and size

The dataset contains 24688 samples.

The 7 eggs are recordings of 30sec with a sample frequency of 50Hz (1500 data points). The pulse signals have a sampling frequency of 10Hz (300 data points). The accelerometers have also 300 points.

1.2. Evaluation of classification

The evaluation metric will be the one used for the initial Kaggle competition: the Mean F1-Score.

The F1 score, commonly used in information retrieval, measures accuracy using the statistics precision *p* and recall *r*. Precision is the ratio of true positives (*tp*) to all predicted positives (*tp* + *fp*). Recall is the ratio of true positives to all actual positives (*tp* + *fn*). The F1 score is given by:

$$F1 = 2\frac{p \cdot r}{p + r} \text{ where } p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}$$

The F1 metric weights recall and precision equally, and a good retrieval algorithm will maximize both precision and recall simultaneously. Thus, moderately good performance on both will be favored over extremely good performance on one and poor performance on the other.

F1 score is computed for each of the 5 classes independently. The 5 obtained sub-scores are then average proportionally to each class frequency.

2. Implementation

2.1 Choice of the network

Following their success in image recognition, Convolutional Neural Network (CNN) have become a point of focus in the field of time series classification [1]. One advantage of CNNs is that they don't require feature engineering. Feature engineering have a strong impact on the performances of a classification model, badly chosen features generally result in very poor performances.

Moreover, Hassan Ismail Fawaz et al. [1] also conclude that Fully Convolutional Neural Networks (FCNs) and Residual Networks (ResNets) show the best performance among the end-to-end models (no feature engineering). I will therefore focus on building a FCN.

2.2 Separable Convolution

If needs be, Depthwise Separable Convolutions will be tested to reduce the computational complexity of the network.

2.3 Weights Initialisation

Kaiming He et al. [2] proposed a new initialization strategy that enable deep networks using ReLU-like activation functions to converge much earlier. This strategy is in particular better than the "Xavier" initialization method that uses a properly scaled uniform distribution.

The Kaiming method, that I will try to implement is as follow:

- Populate a tensor with numbers randomly chosen from a standard normal distribution.
- Multiply each randomly chosen number by √2/√n where n is the number of incoming connections coming into a given layer
- Bias tensors are initialized to zero.

2.4 Activation Function

As we have 5 sleep stages to classify, the final layer's activation function can be either a sigmoid or a SoftMax. The main difference between the two functions is that the output probabilities not necessarily sum-up to 1 with the sigmoid function, whereas they do which the SoftMax. As a result, which the sigmoid, a sample can be in 2 classes with the same probability [3].

Hence if I use more than one channel (e.g. eeg1 and pulse) to classify the sleep stages, I can use the sigmoid function and then use a vote classifier. If I use only one channel, I will have to use the SoftMax in order to have mutually exclusive results.

References

[1] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, Pierre-Alain Muller. Deep learning for

time series classification: a review. Data Mining and Knowledge Discovery, Springer, 2019, 33 (4), pp.917-963. 10.1007/s10618-019-00619-1 . hal-02365025v2

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. arXiv:1502.01852v1 [cs.CV] 6 Feb 2015

[3] Rachel Draelos. Multi-label vs. Multi-class Classification: Sigmoid vs. Softmax, : May 26, 2019. Glass Box.