Deep Learning Challenge

Álvaro Meléndez 2047635 u203902
Francesco Lauria 2041074 u819639
Miftahul Ridwan 2040778 u989303

Codalab Account: FAR_Challenge

Chosen Model & Accuracy

Convolutional 2D: accuracy on validation set 80%, accuracy on test set 67.05%

Procedure and Tested Algorithm

Firstly, we discussed possible algorithms for multiclass classification, such as Decision Tree, Convolutional Neural Network, and Recurrent Neural Network. We decided to develop the Decision Tree algorithm since we learned that Wake stage and REM stage have overlapping frequencies, and thought that we will be able achieve reasonable accuracy by performing feature engineering (taking statistical summary out of raw signal data) as discussed in [1]. We followed [2] and evaluated the algorithm using Gini Index and Information Entropy. We found out that, using the minimum, maximum, mean, standard deviation, kurtosis, and skewness values, the highest accuracy of this algorithm is 47.5% obtained using the Gini index method.

When we performed Exploratory Data Analysis, It has also brought to our attention that Spectrogram data are essentially an image. Since it comes from 2 different sensors, we then focused on developing Convolutional 2D algorithms. Performance of these algorithms will be discussed in Performance Discussion. Moreover, we also explored the well-known VGG16 algorithm. The premises for using VGG16 were understanding spectrograms as images, so the experiments were performed on this network with minimum alteration to its original structure. This architecture is one of the best CNN's in image classification [1], and its success resides in its deep structure. Same padding was added to the last MaxPooling2D layer . The parameters for training were the standard ones for image classification problems [3] which have been proven to be good in classification challenges, being this 128 batches, *Adam* optimizer and categorical cross-entropy. It roughly took around 30 epochs to start overfitting. It is still a 2D convolutional network, which, in the end, will be the one that delivers the best performance.

Lastly, we tried to combine Convolutional 2D with RNN since RNN excel in capturing patterns in sequential data. Details of our tested algorithms are reported on Appendix 1. The model with highest accuracy is highlighted in red.

Performance Discussion

The main structure of this model came from [4]. However, the group created a deeper Network, with different kernel size, units and the use of Leaky ReLU instead of Relu as activation function. These changes were done in order to improve the performance of the model. Thus, different layers have been added or removed as well as the number of the kernel size has been changed until the best accuracy was reached. In addition to that, a small percentage of dropout was added to each layer, likewise an early stopping with patience 10 was added during training. These latter changes were done in order to avoid overfitting [5],[6]. The final architecture can be found in the appendix 2.

Overall, this model reached the best test accuracy among all the models that the group developed, that is, 67% in the test set. Although we implemented two strategies to avoid overfitting and have better generalization, namely: drop out and early stopping. This model could not pass the accuracy of the best line. According to what we have studied, there could be possible reasons for that, for example that the data for training our model were not enough or they are not representative of the test set.

References

1. Chrupała, G., *Feature Engineering* [Powerpoint slides], 2019, retrieved from https://tilburguniversity.instructure.com/courses/1830/files/239777/download?download_frd=1.
2. Decision Tree Implementation using Python, available at https://www.geeksforgeeks.org/decision-tree-implementation-python/.
3. Karen Simonyan, Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. ILSVRC Competition, 2014.
4. Zhihong Cui, Xiangwei Zheng, Xuexiao Shao and Lizhen Cui. *Automatic Sleep stage Classification based on Convolutional Neural Network and Fine-Grained Segments*. Wiley, Hindawi, 2018.
5. Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15.1 (2014): 1929-1958
6. Caruana, Rich, Steve Lawrence, and C. Lee Giles. "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping." *Advances in neural information processing systems*. 2001.

Álvaro Meléndez 2047635 u203902
Francesco Lauria 2041074 u819639
Miftahul Ridwan 2040778 u989303

Codalab Account: FAR_Challenge

Appendix

Appendix 1
Tested Algorithms

Algorithm	**Data**	**Feature Engineering**	**Model Accuracy**	**Codalab Accuracy**	**Notes**
Decision Tree	Raw Signals	Using statistical summary, such as mean, min, max, skewness, kurtosis for each observation	47.5%	Not uploaded	
Convolutional 2D	Spectrogram	Reshaping data into 15375, 100, 30, 2 (changing sensor positional input)	85.7%	64.9%	
Convolutional 2D	Spectrogram	-	80%	67%	Use of dropout and early stopping to avoid overfitting
VGG16 (Convolutional 2D)	Spectrogram	Reshaping data into 15375, 100, 30, 2 (changing sensor positional input)	83%	64%	Added padding = “same” to the last MaxPooling2D layer to match dense input dimensions
Convolutional 2D and LSTM	Spectrogram	-	81.9%	65.6%	

Appendix 2
CNN 2D Architecture

4 layers with this structure

Layer type	Units	Units type	Kernel Size
Conv2D	64	LeakyReLU	4,4
MaxPool2D			2,2
DropOut(0.1)			

5 layers with this structure

Layer type	Units	Units type	Kernel Size
Conv2D	128	LeakyReLU	4,4
MaxPool2D			2,2
DropOut(0.1)			

1 layer with this structure

Layer type	Units	Units type	Kernel Size
Conv2D	256	LeakyReLU	4,4
MaxPool2D			2,2
DropOut(0.1)			

1 layer with this structure

Layer type	Units	Units type

Dense	128	LeakyReLU

Output layer

Layer type	Units	Activation
Dense	6	Softmax