International Journal of
Science and Engineering Investigations
vol. 9, issue 104, September 2020
Received on September 21, 2020
ISSN: 2251-8843

Brain State Classification with Cost-Effective EEG Sensor and Deep Learning

David Lin[1], Gregory Warner[2], Steven Fernandes[3]
[1]Seven Lakes High School
[2]Department of Computer Science and Engineering Technology, University of Houston-Downtown
[3]Department of Computer Science, Design & Journalism, Creighton University
([1]davidzwlin@gmail.com, [2]warnerg1@gator.uhd.edu, [3]StevenFernandes@creighton.edu)

Abstract- We implemented an system to track daily physiological states with a hand-made sensor implemented using a Micro:bit microprocessor. The system is capable of live streaming electroencephalogram (EEG) data, feature extraction, and brain state classification using deep learning models. Empirical Mode Decomposition (EMD) is used to decompose and filter data. Two deep learning models are used to classify brain states based on the decomposed data. The first model uses an ensemble voting mechanism with Keras classifier, and the second model applies a convolutional neural network (CNN) to images generated from the raw EEG data. Both methods confirmed satisfactory performance in brain state classification.

Keywords- *EEG, Brain State classification, Deep Learning*

I. INTRODUCTION

Advanced sensor based wearable devices have shown to be successful in various applications that use physiological data to detect people's brain states. For example, we built an electroencephalogram (EEG) based sleep enhancement system with Delta waves and musical interventions to decrease the frustration and dread associated with sleep complications [1]. Our goal is to develop a light weight cost-effective EEG system for convenient daily usage. The system should have the capability of collecting EEG data, analyzing it, and providing feedback about the brain state classification.

The challenges lie in maximizing the portability and cost-effectiveness while ensuring sufficient accuracy in brain state classification. We investigated the potential of using single channel EEG data to identify brain states. Proper positioning of the EEG sensor, feature extraction, and modeling with effective machine learning algorithms became critical to overcome the limitations of single channel EEG data. Research has discovered that the forehead is a sensitive position in reflecting brain activities [2-3]. As for the machine learning model, deep learning with a neural network has been proven to be the most effective method.

Dry EEG sensors use Electrodermal Activity (EDA), i.e., the human skin's ability to conduct electricity. The conductivity varies with the state of the sweat glands in the skin. This dermic response can be monitored by measuring the changes in skin resistance caused by the sweat [4]. With Micro:bit, EEG is measured by placing two electrodes on the skin, one inch apart. A small voltage is applied to the electrodes, a circuit is formed, and an electrical current flow is generated. The changes in the voltage are used to evaluate the stress level [5]. The forehead is one of the places on human body that have the greatest number of sweat glands [6]. This raises the ability of the EEG data to reflect brain activities.

The organization of this paper is as follows: We present the design of the Micro:bit based EEG device and how it reads EEG data in Section II. In Section III, we present data processing using empirical mode decomposition, and then brain state classification with the Keras and CNN models. In Section IV, we discuss the limitations of the study, future work, and concluding remarks.

II. EEG SENSOR DESIGN

A. EEG Sensor

Existing physiological reading systems, e.g., those used in patient monitoring, are ineffective for daily practices. We aim to implement an environment for daily physiological tracking with EEG sensors. With Micro:bit microprocessor, we implemented an affordable, reusable, expandable, and wireless, device that can monitor the user's brain waves. The device is capable of live streaming collected data and exporting data to a central server via a mobile app. Fig. 1 shows the system architecture.

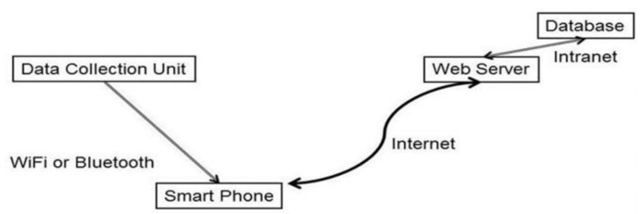
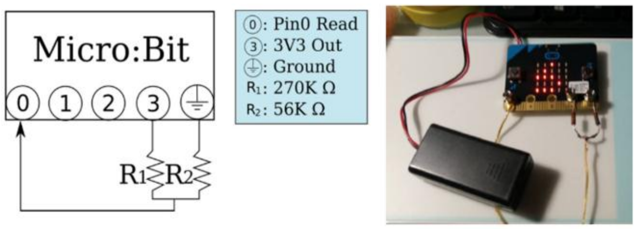


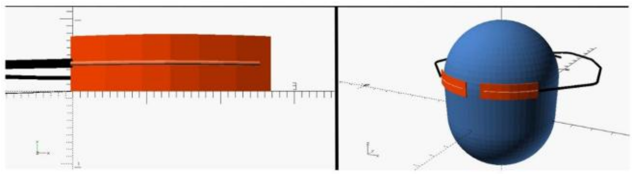
Figure 1. System architecture

For the collection of EEG brain wave signals, a hand-made headset is connected to the mobile phone via a Bluetooth module. Data is collected from the EEG headset and sent to the central server via the mobile phone. The design of the EEG headset is shown in Fig. 2. A voltage divider circuit is used to detect voltage changes between two poles of the electrodes. Two rectangular copper electrodes (35x15mm) are placed on the forehead 15 mm apart. Silicon is placed around the outer edge of the electrodes. The electrodes are connected to a Micro:bit, and a voltage of 0.5 V is sent to the first electrode. The electrical current passes through the skin and into the second electrode. The Micro:bit sends and receives the voltage as an analog value between 0 – 1023 with the maximum voltage being 3.3 V. The analog value is converted to volts with the equation

$$V_{out} = \frac{V_{\max a} \times AO}{AO_{\max}} \tag{1}$$



A. The circuit of the voltage divider (left) and the picture of wiring (right)



B. The electrodes (left) and how to place them on the forehead (right)

Figure 2. EEG device

A voltage divider reduces the 3.3 V output to 0.5 V. The reduced voltage is determined by

$$V_{out} = V_{in} \times \frac{R_2}{R_1 + R_2} \tag{2}$$

The data rendering software on the mobile phone is able to connect with the EEG device and relay the EEG data to the central server, and render the brain state curve on the phone based on server's feedback. Fig. 3 is a snapshot of the mobile app.

B. The Server

The central server handles machine learning for brain state recognition and rendering of brain state information. Both the deep learning methods and the data processing method, i.e., the ensemble empirical mode decomposition (EEMD), have high time and space complexity, and thus require sufficient computing power to ensure real-time data processing and responses to the mobile app's data input.

Fig. 4 illustrates a service log that shows the timestamps (in seconds) of the server's responses to EEG data uploaded in every second. Our EEG acquisition device sends 128 samples of EEG readings every time to the server. It reads a sample whenever it detects a change in voltage readings. Although slow brain activities entail longer data collection time for 128 samples, statistically, it collects 128 samples in every second. Fig. 4 indicates that the server's response time to data uploading allows real-time processing of input data.

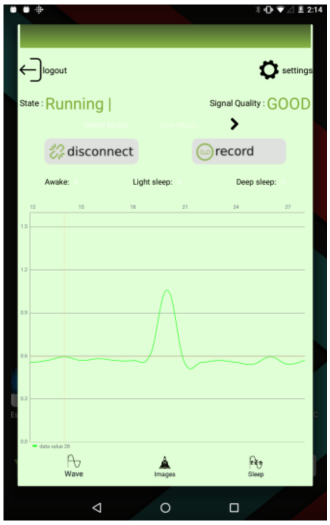


Figure 3. Mobile app

Brain state classification feedback is sent back in a configurable time interval. The 30 second feedback interval is the most frequently used in the current system.

"user_id": 10, "filename": "10_1560363446010.csv"}, {"id": 257, "state": "available", "filetype": "csv", "time": 1560363451,
"user_id": 10, "filename": "10_1560363447158.csv"}, {"id": 258, "state": "available", "filetype": "csv", "time": 1560363452,
"user_id": 10, "filename": "10_1560363448248.csv"}, {"id": 259, "state": "available", "filetype": "csv", "time": 1560363453,
"user_id": 10, "filename": "10_1560363449290.csv"}, {"id": 260, "state": "available", "filetype": "csv", "time": 1560363454,
"user_id": 10, "filename": "10_1560363450439.csv"}, {"id": 261, "state": "available", "filetype": "csv", "time": 1560363455,
"user_id": 10, "filename": "10_1560363451560.csv"}, {"id": 262, "state": "available", "filetype": "csv", "time": 1560363456,
"user_id": 10, "filename": "10_1560363452652.csv"}, {"id": 263, "state": "available", "filetype": "csv", "time": 1560363457,
"user_id": 10, "filename": "10_1560363453733.csv"}, {"id": 264, "state": "available", "filetype": "csv", "time": 1560363459,
"user_id": 10, "filename": "10_1560363454910.csv"}, {"id": 265, "state": "available", "filetype": "csv", "time": 1560363460,
"user_id": 10, "filename": "10_1560363456007.csv"}, {"id": 266, "state": "available", "filetype": "csv", "time": 1560363461,
"user_id": 10, "filename": "10_1560363457148.csv"}, {"id": 267, "state": "available", "filetype": "csv", "time": 1560363462,
"user_id": 10, "filename": "10_1560363458264.csv"}, {"id": 268, "state": "available", "filetype": "csv", "time": 1560363463,
"user_id": 10, "filename": "10_1560363459348.csv"}, {"id": 269, "state": "available", "filetype": "csv", "time": 1560363464,
"user_id": 10, "filename": "10_1560363460496.csv"}, {"id": 270, "state": "available", "filetype": "csv", "time": 1560363465,
"user_id": 10, "filename": "10_1560363461594.csv"}, {"id": 271, "state": "available", "filetype": "csv", "time": 1560363467,
"user_id": 10, "filename": "10_1560363462693.csv"}, {"id": 272, "state": "available", "filetype": "csv", "time": 1560363468,
"user_id": 10, "filename": "10_1560363463808.csv"}, {"id": 273, "state": "available", "filetype": "csv", "time": 1560363469,
"user_id": 10, "filename": "10_1560363465017.csv"}, {"id": 274, "state": "available", "filetype": "csv", "time": 1560363470,
"user_id": 10, "filename": "10_1560363466072.csv"}, {"id": 275, "state": "available", "filetype": "csv", "time": 1560363471,
"user_id": 10, "filename": "10_1560363467182.csv"}, {"id": 276, "state": "available", "filetype": "csv", "time": 1560363472,

Figure 4. Server's log

III. Data Processing and Modeling

There are various methods for EEG data processing [7-8], feature extraction [9-10], and modeling [11-13]. Our experiments indicated that EMD is an effective method for exposing features of EEG data that enhance the performance of brain state classifier.

A. *Empirical Mode Decomposition*

Empirical Mode Decomposition (EMD) transforms wave forms into a series of components called Intrinsic Mode Functions (IMFs). Some recording artifacts such as low frequency drift can be identified by examining the IMFs. Ensemble Empirical Mode Decomposition (EEMD) is a noise-assisted method to improve shifting and generate better EEG data from a selected set of IMFs. Low frequency drift can be removed by eliminating IMFs that show a consistent increasing/decreasing tendency. We then use the remaining IMFs directly as features in the next steps. Fig. 5 shows the IMFs obtained by applying EEMD to a 30 second recording of Micro:bit EEG data, where the red waveform on top is the raw data, and green waveforms thereafter are IMFs. The monotonic IMFs towards the bottom can be regarded as low-frequency drift.

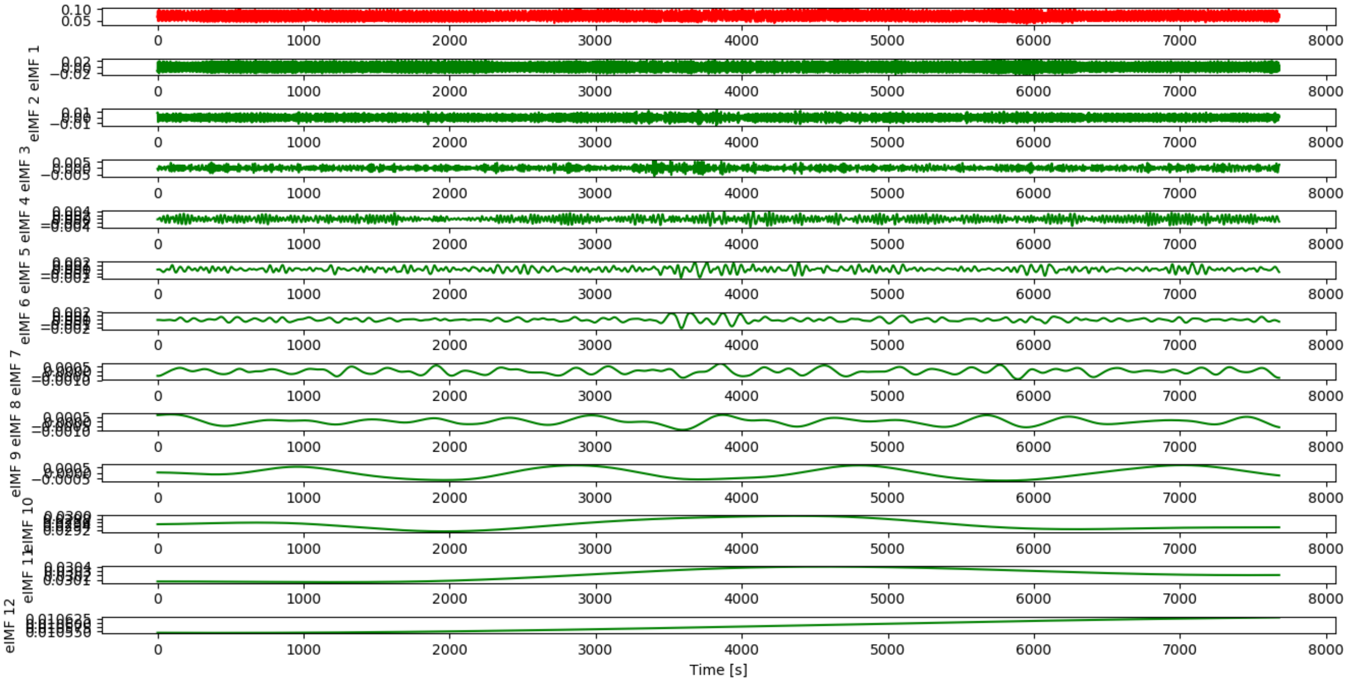


Figure 5. EEMD of Microbit EEG data

B. *Keras Classifier with Ensemble Voting Method*

We use a 30 second window to take a block of EEG sequence to perform EEMD. The number of IMFs returned by EEMD is around 12. We take the first 10 IMFs as the new EEG data for classification. This way, we get rid of low-frequency drift, which exists in the lowest IMFs beyond the first 10.

The dataset we used for the experiment had recordings in 4 brain states, viz., “Control,” “noAction,” “Reading,” and “Piano.” “Control” is the recording of the Micro:bit readings when two electrodes are connected. Therefore, the “Control” state was used as the baseline. “noAction” was the recording while the subject stayed in an idle state. “Reading” was the recording while the subject was reading an article on the computer screen. “Piano” was the recording while the subject was playing piano. All data was generated from a single subject.

We used a Keras classifier to build the model. Fig. 6 shows the structure of the model. The model takes 10 IMFs from EEMD as the input and classifies every EEG sample against the 4 brain states.

```
def baseline_model():
   model = Sequential()
   model.add(Dense(10, input_dim=10,
activation='relu'))
   model.add(Dense(7, activation='relu'))
   model.add(Dense(4, activation='softmax'))
   model.compile(loss='categorical_crossentropy',
        optimizer='adam', metrics=['accuracy'])
   return model

estimator =
KerasClassifier(build_fn=baseline_model,
   epochs=10, batch_size=128, verbose=0)
```

Figure 6. Keras model

The classification performance of the raw IMF data is shown in Fig. 7. The overall accuracy was 68.03%.

The ensemble voting is done by taking the biggest class as the winner for the 30 second segment. The sampling rate is 128, thus each segment has 3,840 predicted labels (and hence voters). The brain state predicted for the corresponding segment is the biggest predicted class.

```
Confusion Matrix
    Pred   0     1      2     3          Legend
Label 0 [46852 13595  5478 10875]        0 - Control
      1 [10349 53464  1377 11610]        1 - noAction
      2 [ 1871    38 74736   155]        2 - Piano
      3 [16816 22424  3625 33935]        3 - Reading
   label    precision   recall
     0       0.617  0.610
     1       0.597  0.696
     2       0.877  0.973
     3       0.600  0.442
precision total: 0.6729
recall total: 0.6803
accuracy: 0.6803
```

Figure 7. Keras raw IMF data classifier performance

The result of classification is shown in Fig. 8, where we can see the classification performance, viz., precision and recall, for each brain state, and the overall accuracy 79.31%. Please note that this performance is validation performance.

```
Confusion Matrix
    Pred  0  1  2  3        Legend
Label 0 [23  5  0  1]       0 - Control
       1 [ 2 24   0   3]        1 -
noAction
       2 [ 0  0 29  0]      2 - Piano
       3 [ 4  9  0 16]      3 - Reading
    label    precision   recall
      0       0.793  0.793
      1       0.632  0.828
      2       1.000  1.000
      3       0.800  0.552
precision total: 0.8062
recall total: 0.7931
```

Figure 8. Keras ensemble classifier performance

This experiment suggests that features embedded in 30 second segments can reflect characteristics of the brain state and enhance the performance of the classifier. We then converted segment EEG data to images and used 2-D convolutional neural networks to classify the images directly.

C. 2-D CNN Classifier Method

Every 30 second segment was converted to an image by stacking up the sub-images converted from each IMF. Each IMF is converted to an 16×240 image and images converted from 10 IMFs are vertically stacked to form an 160×240 image. These images are 80-20% random split for the training and testing datasets.

The CNN classifier is defined using pyTorch in Fig. 9. The model is trained for 100 epochs and then tested. The prediction performance of the model is shown in Fig. 10.

```
class Mish(nn.Module):
    def mish(input):
        return input * torch.tanh(F.softplus(input))
    def __init__(self):
        super().__init__()
    def forward(self, input):
        return self.mish(input)

class EEGImgNet(Module):
    def __init__(self):
        super(EEGImgNet,self).__init__()
        self.block_1 = Sequential(OrderedDict([
            ('conv_1', Conv2d(3, 3,
                 kernel_size = (16, 24), stride = 1)),
            ('conv_2', Conv2d(3, 8,
                 kernel_size = (16, 24), stride = 1)),
            ('bn_1', BatchNorm2d(8)),
            ('act_1', Mish()),
            ('pooling_1', MaxPool2d(kernel_size = (1, 1),
                 stride = (1, 1))),
            ])
        )
        self.block_2 = Sequential(OrderedDict([
            ('conv_1', Conv2d(8, 8,
                 kernel_size = (8, 12), stride = 1)),
            ('bn_1', BatchNorm2d(8)),
            ('act_1', Mish()),
            ('conv_2', Conv2d(8, 8,
                 kernel_size = (8, 12), stride = 1)),
            ('bn_2', BatchNorm2d(8)),
            ('act_2', Mish()),
            ('pooling_1', MaxPool2d(kernel_size = (1, 1),
                 stride = (1, 1))),
            ])
        )
        self.block_3 = Sequential(OrderedDict([
            ('conv_1', Conv2d(8, 16,
                 kernel_size = (4, 6), stride = 1)),
            ('bn_1', BatchNorm2d(16)),
            ('act_1', Mish()),
            ('conv_2', Conv2d(16, 16,
                 kernel_size = (4, 6), stride = 1)),
            ('bn_2', BatchNorm2d(16)),
            ('act_2', Mish()),
            ('conv_3', Conv2d(16, 16,
                 kernel_size = (4, 6), stride = 1)),
            ('bn_3', BatchNorm2d(16)),
            ('act_3', Mish()),
            ('pooling_1', MaxPool2d(
                 kernel_size = (1, 1), stride = (1, 1))),
            ])
        )
        self.fc = Sequential(Linear(32*3*77, 4096),
            Mish(), Dropout2d(2.2),
            Linear(4096, 4096), Mish(), Dropout2d(2.2),
            Linear(4096, 2048), Mish(), Dropout2d(2.2),
            Linear(2048, 4)
        )
    def forward(self,inp):
        tmp = self.block_1(inp)
        tmp = self.block_2(tmp)
        tmp = self.block_3(tmp)
        tmp = tmp.view(tmp.size(0), -1)
        tmp = self.fc(tmp)
        return tmp
```

Figure 9. 2D-CNN model

Comparing the performance of this 2D-CNN model to that of the ensemble Keras model, we can confirm that 30 second segmentation is effective in exposing features of brain states from the EEG data recorded by our Micro:bit-based EEG device.

IV. Discussion and Concluding Remarks

A. Limitations of the Work

The work has two limitations. Firstly, the amount of available data for the experiment is relatively small. This is because our EEG device is still undergoing improvement. Since the wearable headset is used for EEG data collection, the design of the headset plays a big role in users' experience and may affect the quality of the data collected. This also affects the set of the brain states in which quality data can be collected for proper brain state classification. For example, in our experiment, we can observe that the "Piano" state is the most distinguishable state in the ensemble Keras model. This could be since more electromyography and electro-oculography dynamics were involved in the EEG data recording. This effect was leveled off in the CNN model, probably due to CNN model's finer feature extraction capability.

```
Confusion Matrix
    Pred   0   1   2   3          Legend
Label 0 [13   0   6   4]          0 - Control
      1  [ 0  23   0   0]         1 - noAction
      2 [ 4   0 19   0]           2 - Piano
      3 [ 2   1   1 19]           3 - Reading
   label    precision    recall
    0       0.684   0.565
    1       0.958   1.000
    2       0.731   0.826
    3       0.826   0.826
precision total: 0.7999
recall total: 0.8043
```

Figure 10. 2D-CNN classifier performance

Fig. 11 shows the current design of the headset.

Figure 11. EEG heradset design

The second limitation lies in the stability of the EEG data recording function of the device. There are a lot of factors that can affect the recorded EEG data, e.g. the circuitry for detecting the voltage change on the electrodes and the size, shape, and spacing of the electrodes.

B. Future Work

Continuous improvements on the design of the EEG device are needed to improve the quality of the collected data and enrich the set of distinguishable brain states by analyzing the data.

More examination of feature extraction and modeling methods will be done during the improvements of the EEG device design. It is worthwhile to examine whether different feature extraction and segmentation methods work better for different brain states. For example, 30 second segmentation is widely used in EEG data classification [14]. However, different brain states may be suitable for different segmentation window sizes.

Different machine learning methods may be apt for different types of brain state classification as well. For example, recurrent neural networks are good at capturing memory mechanisms in time sequence analysis, generative adversarial networks can be used in data translation and augmentation, etc. It should be useful to develop a machine learning algorithm that considers the historical information of past brain states in the classification of the current brain state.

C. Concluding Remrks

We designed an EEG device by adding circuitry and electrodes to a Micro:bit microprocessor. This low-cost single channel EEG sensor is able to read EEG data by capturing changes of voltage on the human scalp. The recorded EEG data is filtered by ensemble empirical mode decomposition (EEMD), and after dropping the lowest frequency intrinsic mode functions (IMFs) to remove low frequency drift, the remaining IMFs are segmented using a 30 second window. We tested a Keras classifier with ensemble voting and a two-dimensional convolutional neural network that takes the images converted from the EEG segments as input. The prediction accuracy is around 80%. This experiment gives us promising results and sets a starting point for further improvements in the system design.

Acknowledgments

The authors give thanks to Weijie V. Lin, a resident at New York Presbyterian Hospital, Columbia University, for her involvement in setting the application areas and goals of the research and Dr. Venkatesan Rajinikanth at St. Joseph's College of Engineering, India, for his advice in machine learning algorithms.

References

[1] D. Lin, G. Warner, and W. Lin, "A Sleep State Detection and Intervention System," Proceedings of 22ND International Conference on Human-Computer Interaction (HCI International 2020), Copenhagen, Denmark, July 19-24, 2020.

[2] S. Ancoli, E. Peper, and M. Quinn, "Mind/Body integration," New York: Plenum Press (1983).

[3] I. Das and H. Anand, "Effect of Prayer and 'OM' Meditation in Enhancing Galvanic Skin Response," Psychological Thought, 2012.

[4] V. Ramachandran and S. Blakeslee, "Phantoms in the Brain," London: Fourth Estate, 2012, pp. 47.

[5] M. Schwartz and F. Andrasik, "Biofeedback: A Practitioner's Guide," New York: Guilford Press, 2016, pp. 57.

[6] L. Bolis, J. Licinio, and S. Govoni, “Handbook of the Autonomic Nervous System in Health and Disease,” New York: Marcel Dekker, 2003, pp. 257-260.

[7] A. Zeiler, et al, “Empirical Mode Decomposition - an Introduction,” The 2010 International Joint Conference on Neural Networks (IJCNN), Neural Networks (IJCNN), 2010, p. 1. EBSCOhost. doi:10.1109/IJCNN.2010.5596829.

[8] A. Petrosian, "Kolmogorov complexity of finite sequences and recognition of different preictal EEG patterns," Proceedings of Eighth IEEE Symposium on Computer-Based Medical Systems, Lubbock, TX, USA, 1995, pp. 212-217.

[9] F. S. Bao, X. Liu, and C. Zhang, “PyEEG: An Open Source Python Module for EEG/MEG Feature Extraction,” Computational Intelligence & Neuroscience, Jan. 2011, pp. 1–7. EBSCOhost, doi:10.1155/2011/406391.

[10] V. Srinivasan, C. Eswaran, and N. Sriraam, "Approximate Entropy-Based Epileptic EEG Detection Using Artificial Neural Networks," in *IEEE Transactions on Information Technology in Biomedicine*, vol. 11, no. 3, pp. 288-295, May 2007. doi: 10.1109/TITB.2006.884369

[11] S. Siuly, Y. Li, and Y. Zhang, “EEG Signal Analysis and Classification,” in Cham: Springer International Publishing: Imprint: Springer, 2016, ch.1, pp. 3-19.

[12] C. Saitis,M. Z. Parvez, and K. Kalimeri, “Cognitive Load Assessment from EEG and Peripheral Biosignals for the Design of Visually Impaired Mobility Aids,” Wireless Communications & Mobile Computing, Feb. 2018, pp. 1–9. EBSCOhost, doi:10.1155/2018/8971206.

[13] A. Fandango, “Mastering TensorFlow,” Birmingham, UK: Packt Publishing, 2018.

[14] A. Supratak, H. Dong, C. Wu, and Y. Guo, “DeepSleepNet: a Model for Automatic Sleep Stage Scoring based on Raw Single-Channel EEG,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, Volume: 25, Issue: 11, Nov. 2017, pp. 1998 – 2008.

How to Cite this Article:

Lin, D., Warner, G. & Fernandes, S. (2020). Brain State Classification with Cost-Effective EEG Sensor and Deep Learning. International Journal of Science and Engineering Investigations (IJSEI), 9(104), 39-44. http://www.ijsei.com/papers/ijsei-910420-05.pdf