BCNN: A Bantamweight Convolutional Neural Network for P300 Detection

M. Liu[1], *S. Wang*[2,3] *and F. R. Beyette Jr.*[1]

1. School of Electrical and Computer Engineering, University of Georgia, Athens, Georgia, USA
2. Sch. of Math. and Info. Sci. & Tech., Hebei Normal Univ. of Sci. & Tech., Qinhuangdao, Hebei, China
3. School of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei, China
{maohua.liu, fred.beyette}@uga.edu, wangshi2665@hevttc.edu.cn

***Abstract*— As one kind of Event-Related Potentials (ERPs), P300 plays an important role in studying neural activities and cognitive processes, and lays the foundation for P300 speller – the Brain Computer Interface (BCI) working by detecting P300. However, due to P300's high subject variability and low Signal Noise Ratio (SNR), it's always a challenge to do P300 detection. Convolutional Neural Networks (CNNs) have been widely recognized as effective methods for P300 detection, leading to the development of increasingly complex CNN architectures for P300 detection in recent years. However, this paper takes a different approach by proposing the bantamweight CNN (BCNN), the simplest and fast-training CNN for P300 detection. BCNN comprises only one convolutional filter and a total of 141 parameters. Surprisingly, it achieves state-of-the-art performance after just 2 epochs of training, making it an exceptionally lightweight and fast CNN for P300 detection. BCNN not only provides new ideas for CNN design, but also effectively address resource and time constrained situations.**

***Keywords*— ERPs, BCI, CNN, P300 detection, lightweight**

I. Introduction

The measurement known as ERP offers a valuable means of investigating the human brain and mind [1]. In 1929, Hans Berger made a discovery that the electrical activity of the human brain could be detected by placing electrodes on the scalp. This electrical activity is referred to as the electroencephalogram (EEG) and has proven to be quite beneficial. However, the EEG is a rough measurement that represents a mixture of various neural responses. Subsequently, researchers developed techniques to separate out the particular neural responses to an external stimulus. These techniques yield brainwave signals known as ERP, which are utilized for studying highly specific neural processes.

The P300, an ERP component, is produced by the brain when it receives external sensory input such as sight or sound. The P300 response can be amplified through the use of an experimental paradigm known as the oddball paradigm. In this paradigm, both common and uncommon stimuli are presented. As depicted in Figure 1, the EEG signal generated in response to the uncommon stimulus "Os" exhibits a significantly larger deviation in amplitude (occurring around 300ms after the stimulus) compared to the signals elicited by the common stimuli "Xs". The disparity in amplitude deviation between EEG signals evoked by common and uncommon stimuli forms the basis of one popular Brain-Computer Interface - P300 speller. The P300 speller is particularly valuable for individuals who lack the ability to speak or move their limbs, as it enables communication solely through brain activity, independent of limbs or mouth. When it comes to implementing the P300 speller, one of the crucial factors is detecting the P300 signal, which poses a significant challenge. As illustrated in Figure 1, the P300 signal is extremely faint in the EEG, with an amplitude of only a few millivolts, making it highly vulnerable to noises and artifacts of a similar magnitude, such as eye blinks, eye movements, muscle movements, limb movements and tongue movements. Moreover, P300 has high subject variability. Different people have different P300, even the same person can have different P300 in different time periods, making the task of P300 detection consistently challenging.

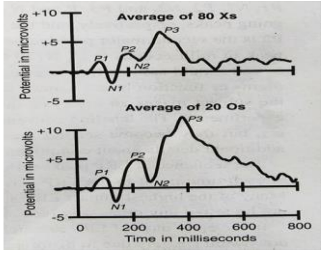

Figure 1. Example of ERP experiment using the oddball paradigm [1]. The subject viewed frequent Xs and infrequent Os on a computer monitor while the EEG was recorded from the active electrode Pz. Separate averages were computed for the X and O epochs. The amplitude of the bottom wave elicited by the uncommon stimuli Os, is obviously greater than the amplitude of the top wave elicited by the common stimuli Xs.

In recent years, CNN has demonstrated its effectiveness in detecting P300 signals and has garnered significant attention. Its ability to autonomously extract relevant features proves invaluable for establishing an automated P300 detection pipeline. Moreover, CNN exhibits promising performance in detecting P300 signals from individual trials [2]. However, CNN's performance in P300 detection still falls short of its remarkable achievements in image processing. Firstly, the generalizability of CNN is relatively low. A CNN model trained using data from one subject does not yield satisfactory results on another subject, and even fine-

979-8-3503-4125-6/23/$31.00 ©2023 IEEE

tuning fails to bridge this gap. Secondly, unlike image processing, simply increasing the depth or number of layers in the network does not significantly improve CNN's performance in P300 detection. The depth of the network and the abundance of parameters do not justify the performance obtained. The challenging factors of P300 data, including substantial subject variability and low SNR, pose significant obstacles for designing a CNN architecture.

This paper explores the underlying principles of cutting-edge CNN architectures utilized in P300 detection and introduces the bantamweight CNN (BCNN), a novel CNN architecture incorporating separable convolution, batch normalization and step learning rate schedule to tackle main issues of P300 data. By leveraging batch normalization to minimize variability and enhance generalizability, separable convolution with minimal parameters to extract features from noisy data and step learning rate to speed up convergence, BCNN aims to achieve a high detection rate while conserving computational resources and time. We conducted a comparative analysis between BCNN and other state-of-the-art CNN architectures, evaluating their performance in terms of detection rate and complexity.

The subsequent sections of this manuscript are organized as follows: In Section 2, a concise overview is provided on state-of-the-art CNN architectures for P300 detection. Section 3 offers a comprehensive explanation of the BCNN architecture. Section 4 details the experimental design and datasets. Section 5 encompasses the experimental results and discussions. Finally, Section 6 concludes the manuscript.

II. Main ideas of cutting-edge CNN architectures for P300 detection

While CNN underwent a similar development path in P300 detection as it did in image processing, there are significant divergences due to the unique qualities of P300 data. The dissimilarities encompass different factors such as feature extraction, complexity, effective convolution and batch normalization.

A. Feature extraction

Feature extraction is the basis and premise for P300 detection. There are two key types of features: spatial features and temporal features. P300 exhibits low spatial resolution, and research has shown that only six channels (Fz, Cz, Pz, PO8, Oz, and PO7) are most relevant for detecting P300 [3]. On the other hand, P300 demonstrates a high temporal resolution accurate to million seconds. As shown in Table 1, almost all cutting-edge CNN architectures have both spatial and temporal filters, but the temporal filters account for more proportion in general. Additionally, some advanced convolutional methods like depth-wise convolution, have been introduced to extract P300 features more effectively.

B. Complexity

Although increasing the number of layers or filters often results in significant performance improvements in image processing, this trend does not seem to apply to P300 detection. On the contrary, substantial increments in network complexity only yield minor enhancements and occasionally even setbacks. As shown in Table 1, initially, a lot of filters and parameters were employed in CNN design for P300 detection, and the number reached the peak in 2015, then the number began to drop to only 225 parameters in 2021. However, the performance is still the same level. Even architecture with only 1 or 2 layers can still achieve state-of-the-art performance.

C. Effective convolution

Due to the marginal performance gains achieved by increasing network complexity, researchers shifted their focus towards finding more efficient convolution methods for P300 detection. Various convolution techniques, such as depth-wise convolution and separable convolution, which can take care of both spatial and temporal features, have been applied for feature extraction in P300. These advancements serve as the groundwork for the creation of efficient and simplified architectures [4].

Table 1. Cutting-edge CNNs for P300 detection (S: spatial, T: temporal, Se: separable, D: depth-wise, F: filters, L: layers) [4].

Architecture	No. Conv filters	No. Conv layers	No. Dense filters & layers	No. Batch layers	No. Params	AUC value Dataset1, Dataset2	Training epochs Dataset1, Dataset2	Time
CNN1	10 S, 50 T	1 S, 1 T	102 F, 2 L	0	1,036,922	0.82±0.05, 0.78±0.04	97±33, 71±14	2010
UCNN1	10 S, 50 T	1 S, 1 T	102 F, 2 L	0	1,036,922	0.84±0.06, 0.78±0.05	88±27, 76±24	2010
CNN3	1 S, 50 T	1 S, 1 T	102 F, 2 L	0	1,031,009	0.78±0.11,0.73±0.08	111±37, 93±31	2010
UCNN3	1 S, 50 T	1 S, 1 T	102 F, 2 L	0	1,031,009	0.83±0.06, 0.76±0.07	114±42, 87±30	2010
CNN-R	96 S, 256 T	1 S, 2 T	6146 F, 3 L	0	19,848,098	0.83±0.06, 0.79±0.04	61±2, 64±2	2015
DeepConvNet	25 S, 375 T	1 S, 4 T	2 F, 1 L	4	139,877	0.84±0.06, 0.79±0.04	122±40, 106±24	2017
ShallowConvNet	40 S, 40 T	1 S, 1 T	2 F, 1 L	1	12,082	0.82±0.07, 0.79±0.03	177±29, 157±33	2017
BN[3]	16 S, 16 T	1 S, 1 T	1 F, 3 L	2	44,589	0.83±0.06, 0.78±0.04	113±21, 95±9	2018
EEGNet	8 T, 16 Se	1 T, 1 D, 1 Se	2 F, 1 L	3	1,394	0.84±0.06, 0.80±0.03	200±3, 198±7	2018
OCLNN	16 T	1 T	2 F, 1 L	0	1,842	0.83±0.06, 0.79±0.04	199±5, 161±26	2018
FCNN	None	None	3 F, 2 L	0	2,477	0.83±0.06, 0.75±0.04	197±7, 132±12	2021
SepConv1D	4 Se	1 Se	1 F, 1 L	0	225	0.84±0.06, 0.78±0.04	199±5, 183±24	2021

D. Batch normalization

First of all, batch normalization plays a critical part in enhancing the generalizability of CNNs for P300 detection by aligning the P300 data within a similar distribution, thereby reducing the high subject variability. Batch normalization's advantage in dealing with high subject variability is increasingly acknowledged, evident in the design of BN^3 [5] [6]. Furthermore, as shown in Figure 2, batch normalization can also smooth the loss landscape during optimization, facilitating training with larger learning rates [7]. Consequently, this capability can significantly accelerate the training process [8] [9]. This is very helpful for most state-of-the-art CNN architectures which always require hundreds of training epochs.

In sum, advanced convolution like depth-wise or separable convolution showcases remarkable efficiency, achieving outstanding performance even with very few filters and layers. Additionally, batch normalization can not only deal with subject variability, but also has the potential to accelerate training. These findings inspired us to construct a CNN architecture that integrates separable convolution, batch normalization and larger learning rates, taking advantage of their benefits.

III. METHONDS

The BCNN is the simplest architecture based on an effective integration of separable convolution, batch normalization and the step learning rate schedule. It can achieve state-of-the-art performance with very few filters and training epochs. Its structure is detailed below.

A. Visulization of BCNN architecture

Figure 3 depicts the visual layout of BCNN, centered around separable convolution and batch normalization techniques. By employing separable convolution, the architecture can effectively combine spatial and temporal convolutions while minimizing the number of parameters involved. This feature proves advantageous in capturing essential spatial and temporal characteristics present in P300 data. Additionally, batch normalization acts as an efficient solution to tackle the considerable subject variability observed in P300 data. It can also aid in quicker convergence. The output layer comprises a solitary neuron that utilizes a sigmoid activation function.

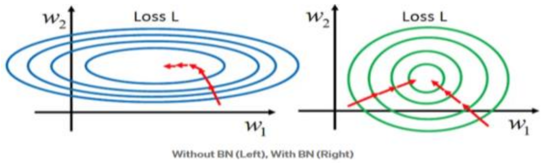

Figure 2. Batch normalization can smooth the loss landscape, thereby enabling large learning rate [6].

B. Visulization of BCNN architecture

Table 2 presents a comprehensive summary of the BCNN architecture, outlining its specific components. The table also provides the following information:

C: Represents the number of channels.

T: Indicates the number of time points in each P300 wave.

F: Refers to the number of filters in the layer.

k: Denotes the size of the kernel.

s: Represents the size of the stride.

p: Indicates the padding size.

C. Step learning rate schedule

Since batch normalization smooths the loss landscape and enables large learning rate, we implement a step learning rate schedule for training. As shown in Figure 4, this is the step learning rate schedule we use for our 2-

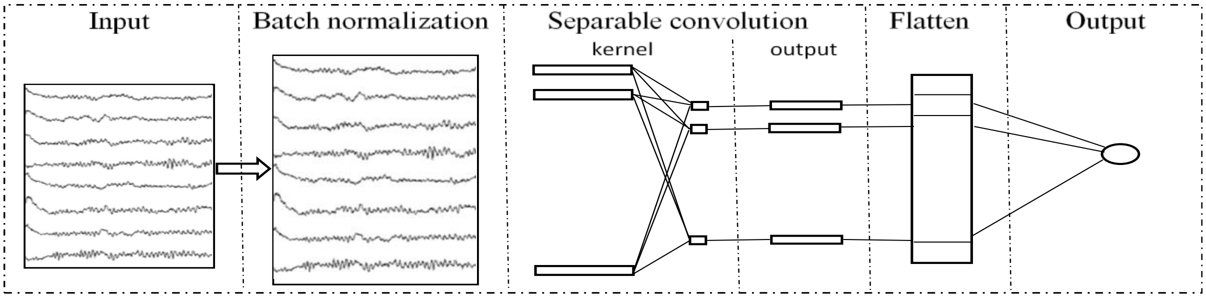

Figure 3. Visualization of BCNN architecture.

Table 2. BCNN architecture.

Block	Layer	# Filters	Size	# Parameters	Output	Activation	Options
1	Input		$T \times C$				
	BatchNorm		$T \times C$	$2\times C$	(T, C)		
	ZeroPadding				$(T+2\times p, C)$		Padding = p
	Separable Convolution	F	Kernel = k Stride = s	$k\times C+F\times C+F$	$(1+(T+2\times p-k)/s, F)$		
	Activation				$(1+(T+2\times p-k)/s, F)$	*Tanh*	
	Flatten				$(1+(T+2\times p-k)/s)$		
Classifier	Dense	1		$1+F\times(1+(T+2\times p-k)/s)$	(1)	*sigmoid*	

epoch training for the 1-filter BCNN. Initially, the learning rate is set to be a high value throughout the first epoch, then decays for the second epoch.

IV. EXPERIMENTAL DESIGN AND ANALYSIS

This section covers the details of implementing BCNN, including the datasets, implementation details, results and analysis.

A. Datasets

We used two benchmark datasets. Dataset1 is "P300 Akimpech database" [10] and Dataset2 is "BNCI Horizon 2020" [11].

Dataset1 was collected from 22 healthy students. It was recorded with ten EEG channels: Fz, C4, Cz, C3, P4, Pz, P3, PO8, Oz, and PO7, following the international 10-20 system. The right earlobe served as the reference location, while the right mastoid was used as the ground location. For our analysis, we specifically focused on six channels (Fz, Cz, Pz, PO8, Oz, and PO7) based on previous studies that suggested their efficacy in P300 detection. The EEG signal was digitized at a sampling rate of 256 Hz and underwent real-time processing. Each stimulus was highlighted for 62.5 ms, with a 125 ms interval between successive stimuli.

Dataset2 was acquired by Riccio et al. from 8 subjects with Amyotrophic Lateral Sclerosis. It was recorded with 8 EEG channels (Fz, Cz, Pz, Oz, P4, P3, PO8, PO7) per the international 10–10 system. Channels referenced to right earlobe, grounded to left mastoid. Signal was digitized at 256 Hz and band-pass filtered (0.1–30 Hz). Each stimulus was highlighted for 125 ms, with a 125 ms interval between successive stimuli.

B. Implementation

The experiments were conducted on a single PC running Windows Enterprise N. The PC was equipped with an Intel(R) Core (TM) i7-8850H CPU operating at 2.60 GHz and 2.59 GHz, along with 16.0 GB of RAM. The implementation of the architectures utilized Keras with Tensorflow 2.10.1 as the backend.

For the implementation of BCNN, the specific configuration parameters are as follows: the number of time points '*T'* is set to 206, the number of channels '*C*' is set to 6 or 8, the padding size '*p*' is set to 4, the number of filters '*F*' used is 1, the kernel size '*k*' is 16, and the stride size '*s*' is 8.

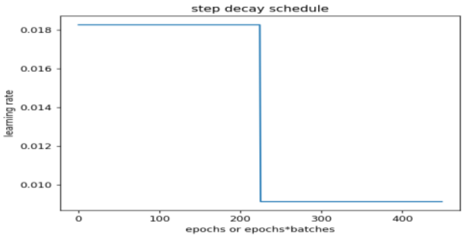

Figure 4. The step learning rate schedule.

For cross-subject P300 detection, we employed a training model that utilized data from a subset of subjects to predict the response of a different subject. This involved a leave-two-out cross-validation method, where one subject was designated for testing, another for validation, and the remaining subjects for training. This process was repeated for each subject, resulting in a number of folds. In each fold, the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) was calculated over the test set, resulting in a number of AUC values. To obtain a representative value, we calculated the mean of these AUC values. The implementation we used is completely identical to the calculation method of the AUC values in Table 1, which were derived using the datasets introduced above. By utilizing the same datasets and calculation method, we enable a fair comparison with previous approaches.

C. Results and analysis

In this analysis, we evaluate the performance including AUC values, number of parameters and saliency maps of 4-filter SepConv1D, 1-filter SepConv1D, and 1-filter BCNN on both Dataset1 and Dataset2.

AUC values on Dataset1. We have computed cross-subject AUC values for each of the three architectures on Dataset1 which contains 22 test subjects, labeled from 0 to 21. This amounts to a total of 110 AUC values, plus the mean AUC value for each architecture, resulting in a grand total of 115 AUC values, as shown in Table 3. In the case of training only 2 epochs, BCNN achieves 0.83 mean AUC value which is higher than two thirds state-

Table 3. AUC values on Dataset1.

Test subject	SepConv1D 4 filters 2 epochs	SepConv1D 1 filter 2 epochs	BCNN 1 filter 2 epochs	SepConv1D 4 filters 33 epochs	BCNN 3 filters 33 epochs
0	0.80	0.77	0.85	0.86	0.87
1	0.86	0.65	0.86	0.87	0.86
2	0.87	0.83	0.87	0.87	0.88
3	0.69	0.71	0.73	0.73	0.73
4	0.63	0.69	0.71	0.7	0.73
5	0.84	0.84	0.89	0.89	0.89
6	0.72	0.64	0.78	0.78	0.79
7	0.74	0.70	0.78	0.77	0.80
8	0.85	0.80	0.86	0.86	0.88
9	0.83	0.80	0.85	0.83	0.85
10	0.81	0.77	0.77	0.77	0.82
11	0.78	0.67	0.77	0.78	0.79
12	0.80	0.77	0.84	0.83	0.83
13	0.84	0.65	0.85	0.84	0.87
14	0.80	0.74	0.82	0.79	0.83
15	0.86	0.79	0.83	0.84	0.84
16	0.67	0.64	0.73	0.71	0.74
17	0.88	0.81	0.90	0.9	0.91
18	0.91	0.88	0.89	0.9	0.91
19	0.85	0.87	0.88	0.87	0.88
20	0.87	0.77	0.90	0.89	0.90
21	0.78	0.71	0.82	0.82	0.83
Mean	**0.80±0.07**	**0.75±0.07**	**0.83±0.06**	**0.82±0.06**	**0.84±0.06**

of-the-art architectures. The second-highest mean AUC value is 0.80 from the 4-filter SepConv1D architecture. The lowest mean AUC value is 0.75 from the 1-filter SepConv1D. Furthermore, after training 33 epochs, BCNN achieves 0.84 mean AUC value, the best performance until now while the 4-filter SepConv1D achieves 0.82. It is worth noting that, the 4-filter SepConv1D can also achieve 0.84, but needs as many as 199 epochs of training. This comparation shows the advantage of the integration of separable convolution, batch normalization and step schedule.

Number of parameters on Dataset1. We also recorded the number of parameters for each architecture. This encompasses trainable parameters, non-trainable parameters, and their cumulative sum. As shown in Table 4, the 4-filter SepConv1D has a higher parameter count compared to 1-filter SepConv1D and BCNN. Conversely, 1-filter SepConv1D and BCNN possess a similar number of parameters, with a negligible disparity, showing the advantage of cost-effectiveness.

Saliency maps on Dataset1. In order to look into what happened in BCNN, we implemented the average saliency maps of different CNNs on the same average P300 trial. Figure 5 is the average saliency maps of three CNNs on the average P300 trial of subject 0 of Dataset1, Brown color (positive) and blue color (negative) denotes two different directions to weight the P300 features. As shown in Figure 5, the first row is a visualization of the average P300 trial in which a peak around 350 ms is very clear. The 4-filter SepConv1D and 1-filter SepConv1D have similar saliency map in general, but the 1-filter SepConv1D suffers more from the noise before 200 ms, and BCNN suffers the least among all. All three CNNs have high saliency after 300 ms and BCNN is the highest.

AUC values on Dataset2. Dataset2 contains 8 test subjects, labeled from 0 to 7, which means a total of 24 AUC values, plus the average AUC value for each architecture, resulting in a grand total of 27 AUC values. As shown in Table 5, after training 2 epochs, the 1-filter BCNN can achieve 0.78 which is higher than two thirds state-of-the-art architectures. It is worth noting that, the best performance of the 4-filter SepConv1D is also 0.78, but needs as many as 183 epochs of training. This result shows the advantage of BCNN in cost-effectiveness, indicating the effective integration of separable convolution, batch normalization and the step schedule.

Number of parameters on Dataset2. Table 6 illustrates the number of parameters required for the 4-filter SepConv1D, 1-filter SepConv1D, and BCNN on Dataset2. Similar with Dataset1, the 4-filter SepConv1D has more parameters than either 1-filter SepConv1D or BCNN. Conversely, 1-filter SepConv1D and BCNN possess a similar number of parameters, with a negligible disparity.

Table 4. Number of parameters on Dataset1.

Parameters	SepConv1D (4 filters)	SepConv1D (1 filter)	BCNN (1 filter)
Trainable	225	141	153
Non-trainable	0	0	12
Total	225	141	165

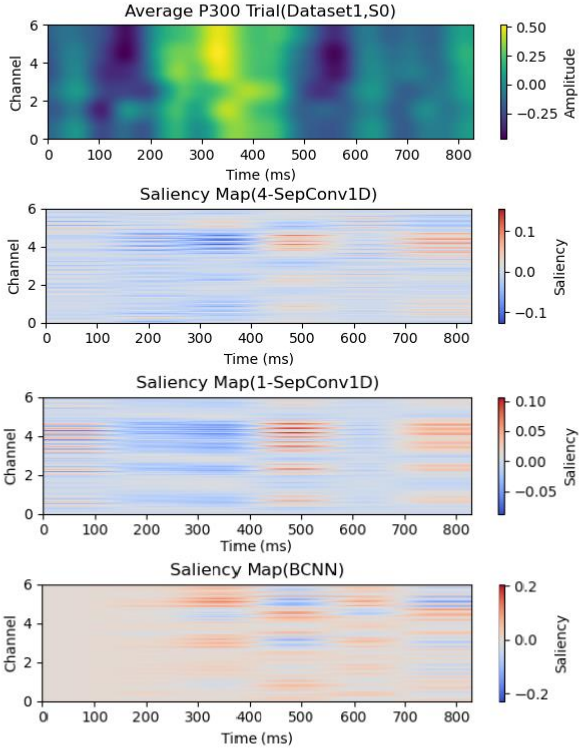

Figure 5. The average saliency maps of three CNNs (4-filter SepConv1D, 1-filter SepConv1D and BCNN) on the average P300 trial of subject 0 in Dataset1.

Saliency maps on Dataset2. Figure 6 is the average saliency maps of three CNNs on the average P300 trial of subject 0 of Dataset2. As shown in Figure 6, the first row is a visualization of the average P300 trial, and there seems no obvious peak after 300 ms like Figure 6 perhaps due to the situation of the subjects. As for the saliency, similar with Dataset1, BCNN has the highest saliency among all (especially around 300 ms and 700 ms). The 4-filter SepConv1D and 1-filter SepConv1D have similar saliency map, but the 1-filter SepConv1D suffers more

Table 5. AUC values on Dataset2.

Test subject	SepConv1D 4 filters 2 epochs	SepConv1D 1 filter 2 epochs	BCNN 1 filter 2 epochs
0	0.65	0.50	0.77
1	0.68	0.64	0.74
2	0.63	0.68	0.82
3	0.57	0.62	0.70
4	0.49	0.51	0.80
5	0.54	0.53	0.82
6	0.75	0.55	0.78
7	0.71	0.55	0.81
Mean	**0.63±0.09**	**0.57±0.07**	**0.78±0.04**

Table 6. Number of parameters on Dataset2.

Parameters	SepConv1D (4 filters)	SepConv1D (1 filter)	BCNN (1 filter)
Trainable	265	163	179
Non-trainable	0	0	16
Total	265	163	195

from the noise (before 200 ms) while the 4-filter SepConv1D has higher saliency (around 700 ms).

V. SUMMARY

In this paper, we presented BCNN for P300 detection. BCNN incorporates separable convolution, batch normalization and step schedule as its fundamental design elements. Through a comparative analysis with 4-filter SepConv1D and 1-filter SepConv1D in cross-subject P300 detection, BCNN showcases exceptional performance, despite having significantly fewer parameters and requiring the fewest training epochs. This remarkable achievement establishes BCNN as the bantamweight CNN solution for P300 detection.

BCNN is specifically designed to leverage the benefits of separable convolution, batch normalization and large learning rate. Separable convolution, renowned for its efficacy in P300 detection, efficiently captures features from P300 signals while employing significantly fewer parameters compared to alternative architectures. Batch normalization effectively tackles the challenge of "high subject variability" by normalizing the data from each subject, promotes smoother loss landscapes, and enables the utilization of large learning rate schedules, thereby speeding up training. Despite only undergoing two epochs of training, BCNN achieves a detection rate comparable to that of the 4-filter SepConv1D architecture which requires about 199 epochs to reach similar performance. Moreover, BCNN's parameter count is impressively low, with only 141 trainable parameters on Dataset1 and 179 parameters on Dataset2. This characteristic renders BCNN highly suitable for situations where both high detection rates and limited computational resources or time constraints are present.

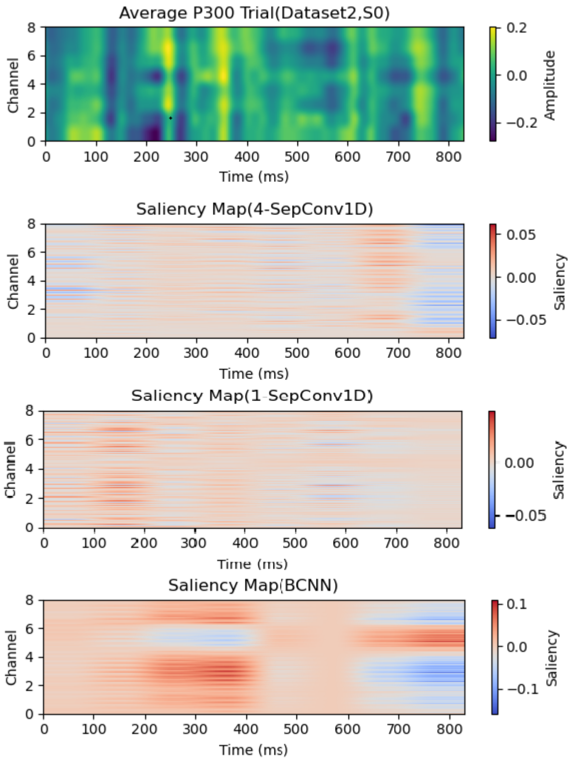

Figure 6. The average saliency maps of three CNNs (4-filter SepConv1D, 1-filter SepConv1D and BCNN) on the average P300 trial of subject 0 in Dataset2.

Several potential avenues can be explored to further enhance BCNN's performance, including investigating the optimal positioning of batch normalization layers, determining the appropriate number of batch normalizations to utilize, and making better use of the other interpretable-AI tools for assistance.

REFERENCES

[1] S. J. Luck, An Introduction to the Event-Related Potential Technique. Cambridge, Massachusetts, MIT press, May 2014.

[2] Du, P., Li, P., Cheng, L., Li, X. and Su, J., "Single-trial P300 classification algorithm based on centralized multi-person data fusion CNN," *Frontiers in Neuroscience*, *17*, p.1132290.

[3] Alvarado-González, M., Garduño, E., Bribiesca, E., Yáñez-Suárez, O. and Medina-Bañuelos, V., "P300 detection based on EEG shape features," *Computational and mathematical methods in medicine*, *2016*.

[4] Alvarado-Gonzalez, M., Fuentes-Pineda, G. and Cervantes-Ojeda, J., "A few filters are enough: Convolutional neural network for P300 detection," *Neurocomputing*, *425*, pp.37-52.

[5] Liu, M., Wu, W., Gu, Z., Yu, Z., Qi, F. and Li, Y., "Deep learning based on batch normalization for P300 signal detection," *Neurocomputing*, *275*, pp.288-297.

[6] Lawhern, V.J., Solon, A.J., Waytowich, N.R., Gordon, S.M., Hung, C.P. and Lance, B.J., "EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces," *Journal of neural engineering*, *15*(5), p.056013.

[7] https://sh-tsang.medium.com/review-batch-normalization-inception-v2-bn-inception-the-2nd-to-surpass-human-level-18e2d0f56651

[8] Santurkar, S., Tsipras, D., Ilyas, A. and Madry, A., "How does batch normalization help optimization?" *Advances in neural information processing systems*, *31*.

[9] Bjorck, N., Gomes, C.P., Selman, B. and Weinberger, K.Q., "Understanding batch normalization," *Advances in neural information processing systems*, *31*.

[10] C. Ledesma-Ramirez, E. Bojorges-Valdez, O. Yáñez-Suarez, C. Saavedra, L. Bougrain, G.G. Gentiletti, "An open-access P300 speller database," Fourth International Brain-Computer Interface Meeting, poster (May 2010).

[11] A. Riccio, L. Simione, F. Schettini, A. Pizzimenti, "Attention and P300-based BCIperformance in people with amyotrophic lateral sclerosis," Frontiers in Human Neuroscience 7 (2013) 732.