An Empirical Study on Sensor-aware Design of Convolutional Neural Networks for P300 Speller in Brain Computer Interface

Hongchang Shan
Leiden University, The Netherlands
hongchang.shan@outlook.com

Yu Liu
KU Leuven, Belgium
yu.liu@esat.kuleuven.be

Todor Stefanov
Leiden University, The Netherlands
t.p.stefanov@liacs.leidenuniv.nl

Abstract—**A Brain Computer Interface (BCI) character speller allows human-beings to directly spell characters using eye-gazes, thereby building communication between the human brain and a computer. Convolutional Neural Networks (CNNs) have achieved state-of-the-art results on the BCI character spelling accuracy. Unfortunately, to the best of our knowledge, it has not been studied whether the CNN should be designed differently to increase the spelling accuracy when the number of sensors used to acquire EEG signals is different. This paper performs an empirical study to investigate this issue. First, we show a motivational example which motivates us for this investigation. Then, we propose a method to design CNNs according to the number of sensors used in the BCI character speller. This method automatically configures a parametric CNN we have devised according to the given number of sensors. Experimental results on six datasets show that we need to design different CNNs when different number of sensors are used for the acquisition of EEG signals. Experimental results also show that our designed sensor-aware CNNs outperform other CNNs in terms of spelling accuracy in most cases. Our CNNs can increase the spelling accuracy achieved by other CNNs with up to 34%.**

I. Introduction

A Brain Computer Interface (BCI) enables direct communication between the human brain and a computer by analyzing the human's neural activities. Traditionally, BCIs are conceived as a pathway for people suffering from motor disabilities [1]. With the rapid development of BCIs, recent research is also focused on developing BCIs for healthy users to allow users' hands-free interaction with consumer applications such as games [2], the dynamic vehicle environment [3] and the home networking system [4]. Due to their non-invasiveness, easiness and safety, Electroencephalogram (EEG)-based BCIs have been studied by many researchers. An EEG-based BCI includes an EEG headset for the acquisition of EEG signals and a hardware/software platform for processing and translating EEG signals into computer commands. An important application of EEG-based BCIs is the P300 speller [5] because the P300 speller has a good performance on BCI character spelling.

In recent years, (deep) Convolutional Neural Networks (CNNs) have been widely studied for character spelling in the P300 speller [6]–[9]. CNNs achieve better character spelling accuracy than traditional machine learning methods [6], [8], [9] because CNNs have the advantage of automatically learning P300-related features from raw EEG signals, thus CNNs can learn not only some features we know but also some features which are important and unknown to us [9].

CNNs learn P300-related features from EEG signals acquired from a number of sensors positioned in the EEG headset. The number of sensors is often different. For example, current popular EEG headsets in BCI systems used for the P300 speller employ different number of sensors. The BCI systems such as EMOTIV Insight, EMOTIV EPOC+, Brain Products ActiCHamp and Biosemi employ 5, 14, 160, and 256 sensors, respectively. In addition, different battery-powered mobile BCI systems use different number of sensors. Such system employs a wireless EEG headset and a resource-constrained hardware platform for data processing. Different number of sensors provide different amount of the data needed to be recorded and processed, so the power consumption of the wireless BCI headset and the hardware platform is different. Different mobile BCI systems have different capacity of batteries which are able to afford different amount of power consumption. As a result, different mobile BCI systems use different number of sensors.

When the EEG signals are recored using different number of sensors, the montages of EEG signals are different, thereby providing different information related to P300 singals. To the best of our knowledge, it has not been studied whether the CNN should be designed differently to further increase the spelling accuracy when the number of sensors used to acquire EEG signals is different. Therefore, in this paper, we perform an empirical study to investigate this issue. More specifically, we aim to answer the following research questions:

RQ1: Should the design of CNN be aware of the number of sensors used in the P300 speller? More specifically, in order to achieve high spelling accuracy, do we need to design different CNNs when different number of sensors are used to acquire EEG signals?

RQ2: If the answer to RQ1 is yes, how do we design a sensor-aware CNN for the P300 speller? More specifically, how do we design a different CNN according to the number of sensors used to acquire EEG signals?

The contributions in this paper are summarized as follows:

- We propose a method to design CNNs according to the number of sensors for the acquisition of EEG signals. In this method, we design a parametric CNN for the P300 speller. Then, we automatically configure this parametric CNN according to the given number of sensors.
- Experimental results show that when using different number of sensors to acquire EEG signals, our method designs completely different CNNs for the P300 speller. Our designed CNNs outperform other CNNs in terms of spelling accuracy in most cases. This shows that the design of CNNs should be aware of the number of sensors

used in the P300 speller. In addition, the experimental results also show that our designed CNNs can increase the spelling accuracy achieved by other CNNs with up to 34%. This shows the effectiveness of our sensor-aware CNN design method.

The rest of the paper is organized as follows: Section II describes the related work. Section III provides some background information on the P300 speller and the datasets used in this paper. Section IV shows a motivational example on sensor-aware CNN. Section V presents our method for designing a CNN according to the number of sensors used to acquire EEG signals. Section VI shows the CNNs designed by our method and compares the spelling accuracy achieved by our designed CNNs and other CNNs. Section VII ends the paper with conclusions.

II. Related Work

[6], [7] and [8] propose CNNs for character spelling in the P300 speller. We call them CCNN [6], CNN-R [7] and BN3 [8], respectively. CCNN, CNN-R and BN3 first use a spatial convolution layer to extract P300-related spatial features. After this spatial convolution layer, they use several temporal convolution layers to extract P300-related temporal features. [9] proposes a CNN, called OCLNN, for character spelling in the P300 speller. OCLNN has only one convolution layer. This layer performs the spatial-temporal convolution, which extracts P300-related spatial-temporal features. However, the aforementioned works have not studied whether the CNN should be designed differently in order to further increase the spelling accuracy when the number of sensors used to acquire EEG signals is different. In this paper, we perform an empirical study to investigate this issue and explore the method to design a sensor-aware CNN for the P300 speller.

III. Background

In this section, we provide some background information for the P300 speller and the datasets used in this paper.

A. P300 Speller

The P300 speller is one of the most investigated applications in BCI [5]. A target character is spelled using the property of the P300 signal. As shown in Fig. 1, a P300 signal, recorded in EEG, occurs as a positive deflection in voltage with a latency of about 300ms after a rare stimulus is presented to a subject (person). The following method is used to evoke a P300 signal in a subject's brain and then the evoked P300 signal is used to spell characters. In this method, the subject is presented with a matrix (see Fig. 2). It is a 6 by 6 matrix and filled with characters. This matrix performs random, separate and successive row or column intensification while the subject is focusing his attention on a target character. When the target row or column is intensified, it is a rare stimulus to the subject because there are two target intensifications out of 12 intensifications. This rare stimulus evokes the subject's brain to generate a P300 signal. Then, with the detection of the P300 signal, the position of the target row or column is inferred. By combining the target row position and the target column position, the position of the target character is inferred. Assume that one epoch consists of 12 intensifications, which contains one target row intensification and one target column intensification. In practice, people use many epochs for the P300 speller, because it is hard to use only one epoch to correctly spell one target character [9], [10]. The employment of a large number of epochs provides the high spelling accuracy. However, this slows down the communication speed of the P300 speller.

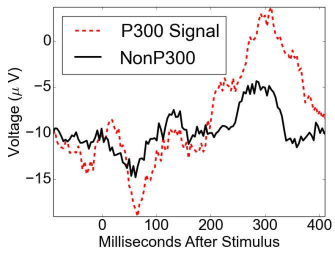

Fig. 1. P300 signal.

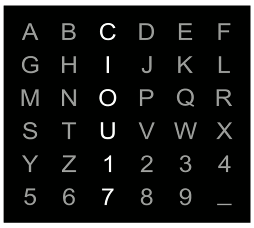

Fig. 2. P300 speller character matrix.

B. Datasets

We use 6 datasets to perform the experiments in this paper. We call these datasets III-A-48, III-A-32, III-A-16, III-B-48, III-B-32 and III-B-16. Here we provide some information on these datasets.

III-A-48, III-A-32 and III-A-16 are from BCI Competition III - Data set II Subject A [11]. III-B-48, III-B-32 and III-B-16 are from BCI Competition III - Data set II Subject B [11]. We use the Signal to Signal and Noise Ratio (SSNR) based sensor selection method [12] to select different number of sensors from an initial set of 64 sensors for Subject A and Subject B, respectively. III-A-48, III-A-32 and III-A-16 are EEG signals from Subject A recorded with the selected 48 sensors, 32 sensors and 16 sensors, respectively. III-B-48, III-B-32 and III-B-16 are EEG signals from Subject B recorded with the selected 48 sensors, 32 sensors and 16 sensors, respectively. Table I shows the detailed information of each dataset.

TABLE I
Subject name and number of sensors for each dataset.

	III-A-48	III-A-32	III-A-16	III-B-48	III-B-32	III-B-16
Subject	A	A	A	B	B	B
# Sensors	48	32	16	48	32	16

The EEG signals in these datasets are recorded when performing the P300 speller described in Section III-A. The EEG signals are then sampled at the frequency of 240Hz. One row or column is intensified for 100ms. After each row/column intensification, the matrix is blank for 75ms. 15 epochs are used for one character. After every group of 15 epochs, the matrix is blank for 2.5s. This informs that the subject should focus his attention on the next character to spell.

For each dataset, there are two separate sub-datasets. We use the first sub-dataset to design a sensor-aware CNN, and we call this sub-dataset the preliminary dataset. We use the second sub-dataset to evaluate the character spelling accuracy achieved by our sensor-aware CNN, and we call this sub-dataset the evaluation dataset. Each preliminary dataset has 85 characters and each evaluation dataset has 100 characters.

IV. Motivational Example on Designing Different CNNs for P300 Speller

In this section, we show a motivational example which motivates us to investigate **RQ1** and **RQ2** (introduced in Section I). In this example, we explore whether the CNN, which achieves the highest spelling accuracy, is different when the number of sensors used to acquire EEG signals is different.

We use each evaluation dataset of III-B-48, III-B-32 and III-B-16 to evaluate the spelling accuracy achieved by CCNN [6], CNN-R [7], BN3 [8] and OCLNN [9]. The spelling accuracy for different epoch numbers is calculated by Eq. (1), where acc_k denotes the spelling accuracy when using the first k epochs for each character, R_k denotes the number of correctly predicted characters when using the first k epochs for each character, and A denotes the number of all characters.

$$acc_k = \frac{R_k}{A} \tag{1}$$

We also evaluate the spelling accuracy achieved by another CNN called Double Temporal Layer Neural Network (DTLNN). The architecture of DTLNN is shown in Table II. The first column in this table lists different layers. The second column presents the operation performed in the corresponding layer. The third column describes the kernel size in a convolution layer. The last layer provides the number of feature maps/neurons used in a convolution/fully-connected layer. DTLNN has three layers in total, i.e., Layer 1, Layer 2 and Layer Out. Layer 1 and Layer 2 both perform the convolution operation with the kernel size (4,1). The number of feature maps for Layer 1 and Layer 2 is 16. The activation function we use for Layer 1 and Layer 2 is the Rectified Linear Unit (ReLU) function. Layer Out performs the fully-connected operation with 2 neurons. The activation function we use for Layer Out is the Softmax function.

TABLE II
THE ARCHITECTURE OF DTLNN.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(4,1)	16
2	Convolution	(4,1)	16
Output	Fully-Connected	—	2

The experimental results are shown in Table III, IV and V. In these tables, the first column lists different CNNs for comparison. Each row provides the spelling accuracy achieved by a CNN for different epoch numbers $k \in [1, 15]$. An accuracy number in bold indicates that the corresponding CNN achieves the highest spelling accuracy among all CNNs.

TABLE III
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-B-48.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
DTLNN	35	**55**	60	67	76	80	**85**	89	92	92	94	95	95	**96**	97
OCLNN	**41**	53	64	70	79	82	82	84	91	**96**	**95**	**97**	**96**	**96**	**98**
CCNN	38	42	54	64	71	78	81	85	87	86	88	92	93	92	96
CNN-R	33	52	**67**	**74**	**80**	**84**	84	**91**	**93**	93	93	95	95	**96**	**98**
BN3	33	43	53	65	75	76	80	82	87	92	92	96	93	93	93

Table III shows that when using 48 sensors to record Subject B's brain signals, for large epoch numbers ($k \in [10, 15]$), OCLNN achieves the highest spelling accuracy compared to other CNNs. For most small epoch numbers ($k = 3, 4, 5, 6, 8, 9$), CNN-R achieves the highest spelling accuracy compared to other CNNs. Table IV shows that when using 32 sensors to record Subject B's brain signals, OCLNN achieves the highest spelling accuracy compared to other CNNs in most cases (13 out of 15 different epoch numbers). Table V shows that when using 16 sensors to record Subject B's brain signals, DTLNN achieves the highest spelling accuracy compared to other CNNs in most cases (12 out of 15 different epoch numbers).

TABLE IV
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-B-32.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
DTLNN	39	48	55	66	74	80	84	82	88	91	90	93	94	94	**97**
OCLNN	**40**	51	**61**	**72**	**78**	81	**85**	**86**	**91**	**93**	**94**	**94**	**95**	**96**	**97**
CCNN	36	50	58	**72**	75	79	84	83	89	90	91	92	92	94	96
CNN-R	39	**57**	56	65	77	**82**	82	84	90	91	91	92	92	93	**97**
BN3	32	42	53	64	68	72	78	79	85	89	90	92	94	95	96

TABLE V
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-B-16.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
DTLNN	**41**	**60**	**65**	**73**	75	**83**	**85**	**88**	**90**	90	91	**95**	**95**	**96**	**97**
OCLNN	**41**	53	**65**	70	**82**	82	83	84	89	**92**	90	91	90	95	96
CCNN	36	42	50	57	61	66	75	82	81	86	88	87	91	90	94
CNN-R	**41**	56	60	70	73	**83**	83	85	87	**92**	92	92	92	95	96
BN3	33	35	50	63	68	72	78	80	85	88	**93**	91	94	89	95

These experimental results show that when different number of sensors are used to acquire EEG signals, the CNN which achieves the highest spelling accuracy is different. The CNN, which performs well on the EEG signals acquired with certain number of sensors, may not perform well on the EEG signals acquired with other number of sensors. This motivates us to further investigate **RQ1** and **RQ2**.

V. OUR SENSOR-AWARE CNN DESIGN FOR P300 SPELLER

This section introduces our proposed method for designing a CNN according to the number of sensors used to record EEG signals. First, we design a parametric CNN for the P300 speller. The achitecture of this parametric CNN is described in Section V-A. Then, we automatically configure this parametric CNN according to the given number of sensors used to acquire EEG signals. The configuration of the parametric CNN is described in Section V-B. After this configuration, our method outputs the CNN designed for the given number of sensors.

A. Parametric CNN for P300 Speller

This section introduces how we construct the parametric CNN for the P300 speller with a given number of sensors. We call this parametric CNN PaC. First, we describe the input to PaC in Section V-A1. Then, we introduce the architecture and training of PaC in Section V-A2 and Section V-A3, respectively. Finally, we describe how PaC is used for character spelling in Section V-A4.

1) Input Tensor: The input to PaC is the tensor ($Tem \times C$). C denotes the number of given sensors used to acquire EEG signals. Tem is the number of signal samples in the time domain. In this tensor, in order to remove the high frequency noise, the temporal signal samples are bandpass filtered between 0.1Hz and 20Hz. Then, we normalize the temporal signal samples to make the signal samples to have zero mean and unit variance based on each individual pattern and for each sensor. Here an individual pattern denotes the Tem signal samples. Normalization is a common requirement for processing the input data to CNNs. Normalization helps the CNN perform well for the P300 spelling [6], [8].

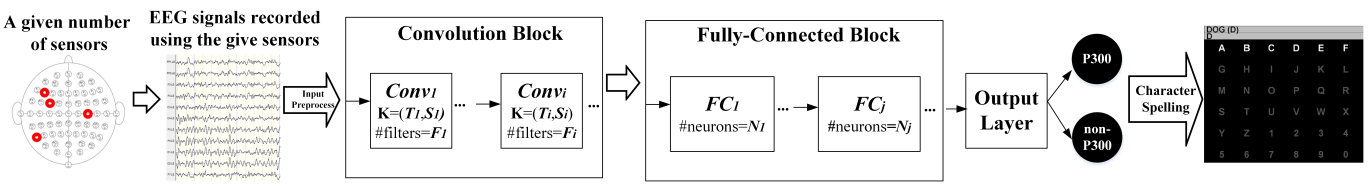

Fig. 3. Illustration of our parametric CNN for P300 spelling, where K denotes the kernel size in a convolution layer.

2) Network Architecture: The overall network architecture of PaC is shown in Fig. 3. The symbols used in Fig. 3 are described in VI. Our PaC has three parts, i.e., the convolution block, the fully-connected block and the output layer.

TABLE VI
THE SYMBOLS USED TO CONSTRUCT PARAMETRIC CNN.

Symbol	Description
$Conv_i$	The convolution operation type in the ith convolution layer.
T_i	The value for the kernel size in the time domain for the ith convolution layer.
S_i	The value for the kernel size in the space domain for the ith convolution layer.
F_i	The number of feature maps generated by the ith convolution layer.
FC_j	Whether or not using the jth fully-connected layer.
N_j	The number of neurons used in the jth fully-connected layer.

The first part, the convolution block, contains several convolution layers. We parameterize this block as the following. For the ith convolution layer, the kernel size is (T_i, S_i). This kernel size is controlled by $Conv_i$. $Conv_i$ is used to parametrize the convolution operation type. When $Conv_i$ is 1, the ith convolution layer performs the temporal convolution with the kernel size (T_i, 1). When $Conv_i$ is 2, the ith convolution layer performs the spatial convolution with the kernel size (1, C). When $Conv_i$ is 3, the ith convolution layer performs the spatial-temporal convolution with the kernel size (T_i, C). When $Conv_i$ is 0, we do not use the ith convolution layer in our PaC. The ith convolution layer generates F_i feature maps. The activation function for each layer employs the Rectified Linear Unit (ReLU) function. We do not use overlapped convolution, because overlapped convolution proves to be not helpful for increasing the spelling accuracy for the P300 speller [8].

The second part, the fully-connected block, contains several fully-connected layers. We parameterize this block as the following. We use FC_j to control whether or not we use the jth fully-connected layer in PaC. When FC_j is 0, we do not use the jth fully-connected layer. When FC_j is 1, we use the jth fully-connected layer and the number of neurons used in this layer is N_j. The activation function for each layer employs the Rectified Linear Unit (ReLU) function. After the last fully-connected layer in this block, we use dropout [13] with a rate of 0.4 to prevent PaC from being overfitting.

The last part is the output layer. This layer performs the fully-connected operation with two neurons. These two neurons represent the class "P300" (the presence of a P300 signal) and the class "non-P300" (the absence of a P300 signal), respectively. This layer employs the Softmax function which outputs the predicted probability for "P300" class and "non-P300" class.

3) Training: The training is carried out by minimizing the binary cross-entropy loss function. It uses Stochastic Gradient Descent optimizer with momentum and weight decay. The momentum is 0.9 and the weight decay is 0.0005. The learning rate is fixed with 0.01. The batch size is 128. The setup of the training considers the suggestion in [14].

4) Character Spelling Using PaC: We use the predicted probability by PaC for class "P300" to calculate the position of the target character in the P300 speller. The calculation for the position of the target character when using the first k epochs is defined by Eq. (2), (3) and (4), where $P_{(q,r)}$ denotes the predicted probability by PaC for "P300" class at intensification q and epoch r, $Sump_{(q)}$ denotes the sum of the predicted probabilities at intensification q, $index_{col}$ denotes the index of the target column position, and $index_{row}$ denotes the index of the target row position. q denotes a column intensification when $q \in [1,6]$ and q denotes a row intensification when $q \in [7,12]$.

$$Sump_{(q)} = \sum_{r=1}^{k} P_{(q,r)} \tag{2}$$

$$index_{col} = \underset{1 \leq q \leq 6}{argmax}\ Sump_{(q)} \tag{3}$$

$$index_{row} = \underset{7 \leq q \leq 12}{argmax}\ Sump_{(q)} \tag{4}$$

B. Automated Network Configuration

This section introduces how we automatically configure PaC according to a given number of sensors in the P300 speller.

Sequential Model-based Algorithm Configuration (SMAC) [15] is used as a configuration procedure for our PaC, because SMAC is one of the best-performing and versatile algorithm configuration procedures currently available. It is a state-of-the-art, general-purpose algorithm configurator based on the concept of sequential model-based optimisation (also known as Bayesian optimisation). The detailed explanation of SMAC please refer to [15].

The objective of the automated network configuration is to configure our PaC such that PaC is able to achieve high spelling accuracy across different epoch numbers. Therefore, we use $Quality$ as the minimizing objective for SMAC. $Quality$ is defined by Eq. (5) and (1) where E denotes the number of epochs. $1/E \times \sum_{k=1}^{E} acc_k$ is the average spelling accuracy across all epoch numbers $k \in [1,E]$.

$$Quality = 1 - 1/E \times \sum_{k=1}^{E} acc_k \tag{5}$$

The input to SMAC is the initialized parameters of PaC. These parameters are described in Table VI. The initialized values of these parameters are shown in Table VII. The first row in this table lists different parameters to initialize. The second row describes the initialized values for these parameters. With the initialized parameters given in Table VII, the initialized PaC is OCLNN [9].

The configuration space for the parameters of PaC is shown in Table VIII. The first column lists different parameters.

TABLE VII
THE INITIALIZED PARAMETERS FOR PAC.

$Conv_1$	$Conv_t$	T_1	T_t	S_1	S_t	F_1	F_t	FC_t	N_t
	$t \in [2,5]$		$t \in [2,5]$		$t \in [2,5]$		$t \in [2,5]$	$t \in [1,5]$	$t \in [1,5]$
3	0	16	0	C	1	16	0	False	0

The second column provides the type of each parameter. The last column describes the value of each parameter. $Tr_{(i)}$ in this table denotes the number of the input temporal signal samples to the *i*th convolution layer after performing the (*i*-1)th convolution layer.

TABLE VIII
THE CONFIGURATION SPACE FOR THE PARAMETERS OF PAC .

Parameter	Type	Domain
$Conv_i$	Categorical	{0, 1, 2, 3}
T_i	Integer	[1, $Tr_{(i)}$]
S_i	Categorical	{1, C}
F_i	Integer	[1, 256]
FC_j	Boolean	{True,False}
N_j	Integer	[1, 256]

The configuration process is the following. In each iteration of the configuration, first, PaC is constructed by the parameters selected by SMAC. Then, we use the training data to train the constructed PaC. The trained PaC is evaluated on the validation data to calculate *Quality* using Eq. (5) and (1). Based on the calculated *Quality*, SMAC reconfigure the parameters of PaC such that the reconfigured PaC achieves minimized *Quality* in the next iteration. In this process, the training data and the validation data are different EEG signals but acquired with the same sensors. The training data and the validation data influence the value of *Quality*, thereby influencing the optimizing process of SMAC. In this way, our automated network configuration is aware of the give number of sensors used in the P300 speller.

Algorithm 1: Rule for Configuring S_i.

```
  Input: $S_i$
  Output: $S_i$
1 if $Conv_i$ = 1 then
2     $S_i = 1$
3 else
4     $S_i = C$
```

Algorithm 2: Rule for Configuring $Conv_i$.

```
  Input: $Conv_i$
  Output: $Conv_i$
1 for $1 \le h \le i-1$ do
2     if $Conv_h$ = 2 or 3 then
3         $Conv_i = 1$
```

In the configuration process, we set two rules. The first rule is for configuring S_i, shown in Algorithm 1. This rule is to force a convolution layer to learn P300-related features from the EEG signals acquired using all the given sensors when this layer is configured to perform a spatial convolution or a spatial-temporal convolution. The reason for using all sensors is that it is more helpful than using part of all sensors to increase the spelling accuracy for the P300 speller [6]–[9]. The second rule is for configuring $Conv_i$, shown in Algorithm 2. This rule is that the current convolution layer has to perform a temporal convolution when any of the previous convolution layers performs a spatial convolution or a spatial-temporal convolution. The reason is that EEG signals acquired using all the given sensors are used to extract P300-related spatial features through the spatial/spatial-temporal convolution in PaC. If a previous convolution layer performs the spatial/spatial-temporal convolution, the current layer can not perform the spatial/spatial-temporal convolution any more. As a result, this layer can perform only the temporal convolution.

The output of SMAC is the optimized parameters of PaC. We use these parameters to construct the optimized PaC for the P300 speller with the given number of sensors.

VI. EXPERIMENTAL RESULTS

In this section, first, we introduce our experimental setup in Section VI-A. Then, in Section VI-B, we show the PaC configured by our method when using different number of sensors to record EEG signals. In the last, we compare the spelling accuracy achieved by our configured PaC and other CNNs in Section VI-C.

A. Experimental Setup

Our experiments have two part: the first part is the configuration of PaC according to the given number of sensors in the P300 speller. The second part is the evaluation of our configured PaC in terms of spelling accuracy.

We perform the automated network configuration for PaC on each preliminary dataset of Dataset III-A-48, III-A-32, III-A-16, III-B-48, III-B-32 and III-B-16. In the configuration process, 60% of each preliminary dataset is used to train PaC for the corresponding dataset, and the left 40% of each preliminary dataset is used as the validation data to calculate *Quality* (described in Section V-B). For the input tensor to PaC, Tem = 240 because the sampling frequency for EEG signals is 240Hz and we take each individual pattern to be the signal samples between 0 and 1000 ms posterior to the beginning of each intensification. C is different and dependent on the dataset which we configure PaC for. More specifically, C = 48, 32, 16, 48, 32 and 16 when configuring PaC for Dataset III-A-48, III-A-32, III-A-16, III-B-48, III-B-32 and III-B-16, respectively.

To compare the spelling accuracy achieved by our configured PaC and other CNNs, we use the evaluation dataset of Dataset III-A-48, III-A-32, III-A-16, III-B-48, III-B-32 and III-B-16 to calculate the spelling accuracy achieved by different CNNs. The spelling accuracy for different epoch numbers $k \in [1, 15]$ is calculated by using Eq. (1). The CNNs for comparison with our PaC is CCNN [6], CNN-R [7], BN3 [8] and OCLNN [9].

B. Configured PaC for Different Datasets

Table IX, X, XI, XII, XIII, and XIV describe the architecture of the configured PaC for different datasets. We use PaC-IIIA48, PaC-IIIA32, PaC-IIIA16, PaC-IIIB48, PaC-IIIB32 and PaC-IIIB16 to denote the configured PaC for Dataset III-A-48, Dataset III-A-32, Dataset III-A-16, Dataset III-B-48, Dataset III-B-32, and Dataset III-B-16, respectively. In these tables, the first column lists different layers. The second column describes the operation performed in the corresponding layer. The third layer introduces the kernel size used in a convolution layer. The last layer provides the number of feature maps/neurons in a convolution/fully-connected layer.

From Table IX, X, XI, XII, XIII, and XIV, we can see that for both two subjects, the configured PaC is different when different number of sensors are used to acquire EEG signals.

TABLE IX
CONFIGURED PAC-IIIA48 FOR DATASET III-A-48.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(4,1)	49
2	Convolution	(30,1)	40
3	Convolution	(2,1)	63
4	Fully-Connected	—	14
5	Fully-Connected	—	58
6	Fully-Connected	—	57
Output	Fully-Connected	—	2

TABLE X
CONFIGURED PAC-IIIA32 FOR DATASET III-A-32.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(5,32)	22
2	Convolution	(15,1)	53
3	Fully-Connected	—	30
Output	Fully-Connected	—	2

TABLE XI
CONFIGURED PAC-IIIA16 FOR DATASET III-A-16.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(16,1)	29
Output	Fully-Connected	—	2

TABLE XII
CONFIGURED PAC-IIIB48 FOR DATASET III-B-48.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(1,48)	30
2	Convolution	(16,1)	34
Output	Fully-Connected	—	2

TABLE XIII
CONFIGURED PAC-IIIB32 FOR DATASET III-B-32.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(2,32)	21
2	Convolution	(16,1)	52
3	Convolution	(5,1)	3
4	Fully-Connected	—	23
Output	Fully-Connected	—	2

TABLE XIV
CONFIGURED PAC-IIIB16 FOR DATASET III-B-16.

Layer	Operation	Kernel Size	Feature Maps/Neurons
1	Convolution	(2,16)	23
2	Convolution	(24,1)	13
3	Convolution	(2,1)	3
4	Convolution	(2,1)	24
5	Fully-Connected	—	4
6	Fully-Connected	—	34
Output	Fully-Connected	—	2

C. Character Spelling Accuracy

This section compares the character spelling accuracy achieved by our configured PaC and other CNNs on Dataset III-A-48, Dataset III-A-32, Dataset III-A-16, Dataset III-B-48, Dataset III-B-32 and Dataset III-B-16, respectively. The experimental results are shown in Table XV, XVI, XVII, XVIII, XIX, and XX. Overall, the spelling accuracy achieved by our configured PaC (i.e., PaC-IIIA48, PaC-IIIA32, PaC-IIIA16, PaC-IIIB48, PaC-IIIB32, PaC-IIIB16) is higher than the spelling accuracy achieved by other CNNs in 86 out of 90 cases. Our configured PaC is able to increase the spelling accuracy achieved by other CNNs with up to 34%.

TABLE XV
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-A-48.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PaC-IIIA48	**27**	**34**	**51**	**63**	**66**	**76**	**80**	**83**	**88**	**91**	**92**	**94**	**96**	**96**	97
OCLNN	21	24	44	54	63	68	78	81	83	86	90	92	93	94	**98**
CCNN	18	28	40	52	58	61	67	71	74	85	85	87	89	93	97
CNN-R	17	31	43	41	51	65	72	74	79	85	90	90	92	92	96
BN3	14	16	29	32	41	43	46	55	60	68	68	75	76	75	82

TABLE XVI
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-A-32.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PaC-IIIA32	**22**	**32**	**49**	**53**	**63**	**68**	**77**	**78**	**81**	**87**	**88**	**92**	**93**	**95**	**96**
OCLNN	14	23	41	48	60	65	75	75	76	86	85	89	91	90	**96**
CCNN	16	23	37	45	54	64	65	67	70	79	80	83	85	90	**96**
CNN-R	14	24	38	42	50	61	71	71	72	78	84	87	91	90	95
BN3	13	16	24	29	41	41	49	58	63	65	70	73	73	78	79

TABLE XVII
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-A-16.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PaC-IIIA16	**17**	**28**	**40**	**54**	**61**	**64**	**71**	**75**	**79**	**85**	**89**	**93**	**92**	**95**	**95**
OCLNN	14	26	32	43	54	58	**71**	69	77	81	84	86	86	94	92
CCNN	13	22	28	43	47	54	57	66	65	70	67	75	81	83	86
CNN-R	14	20	23	42	48	60	61	65	68	78	77	85	86	87	91
BN3	14	11	24	32	39	44	47	51	57	60	64	74	75	77	82

TABLE XVIII
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-B-48.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PaC-IIIB48	**46**	**65**	**72**	**79**	**84**	**89**	**91**	**93**	**95**	**96**	**96**	**97**	**97**	**98**	**98**
OCLNN	41	53	64	70	79	82	82	84	91	**96**	95	**97**	96	96	**98**
CCNN	38	42	54	64	71	78	81	85	87	86	88	92	93	92	96
CNN-R	33	52	67	74	80	84	84	91	93	93	93	95	95	96	**98**
BN3	33	43	53	65	75	76	80	82	87	92	92	96	93	93	93

TABLE XIX
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-B-32.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PaC-IIIB32	**44**	**57**	**68**	**77**	**83**	**85**	**88**	**89**	**91**	**94**	**95**	**95**	**96**	**96**	**97**
OCLNN	40	51	61	72	78	81	85	86	**91**	93	94	94	95	**96**	**97**
CCNN	36	50	58	72	75	79	84	83	89	90	91	92	92	94	96
CNN-R	39	**57**	56	65	77	82	82	84	90	91	91	92	92	93	**97**
BN3	32	42	53	64	68	72	78	79	85	89	90	92	94	95	96

For Dataset III-A-48 (see Table XV), for epoch numbers $k \in [1, 14]$, the spelling accuracy achieved by our PaC-IIIA48 is higher than the spelling accuracy achieved by other CNNs. Our PaC-IIIA48 is able to increase the spelling accuracy achieved by other CNNs with up to 34%. The largest accuracy improvement occurs when comparing the spelling accuracy achieved by our PaC-IIIA48 with the spelling accuracy achieved by BN3 on epoch number $k = 7$.

TABLE XX
CHARACTER SPELLING ACCURACY ACHIEVED BY DIFFERENT CNNS ON DATASET III-B-16.

Network	Epochs														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PaC-IIIB16	**43**	**64**	**70**	**75**	79	**85**	**86**	**86**	**90**	**92**	**94**	**93**	**94**	94	95
OCLNN	41	53	65	70	**82**	82	83	84	89	**92**	90	91	90	**95**	**96**
CCNN	36	42	50	57	61	66	75	82	81	86	88	87	91	90	94
CNN-R	41	56	60	70	73	83	83	85	87	**92**	92	92	92	**95**	**96**
BN3	33	35	50	63	68	72	78	80	85	88	93	91	**94**	89	95

For Dataset III-A-32 (see Table XVI), for all epoch numbers $k \in [1, 15]$, the spelling accuracy achieved by our PaC-IIIA32 is higher than the spelling accuracy achieved by other CNNs. Our PaC-IIIA32 is able to increase the spelling accuracy achieved by other CNNs with up to 28%. The largest accuracy improvement occurs when comparing the spelling accuracy achieved by our PaC-IIIA32 with the spelling accuracy achieved by BN3 on epoch number $k = 7$.

For Dataset III-A-16 (see Table XVII), for all epoch numbers $k \in [1, 15]$, the spelling accuracy achieved by our PaC-IIIA16 is higher than the spelling accuracy achieved by other CNNs. Our PaC-IIIA16 is able to increase the spelling accuracy achieved by other CNNs with up to 25%. The largest accuracy improvement occurs when comparing the spelling accuracy achieved by our PaC-IIIA16 with the spelling accuracy achieved by BN3 on epoch numbers $k = 10,\ 11$.

For Dataset III-B-48 (see Table XVIII), for all epoch numbers $k \in [1, 15]$, the spelling accuracy achieved by our PaC-IIIB48 is higher than the spelling accuracy achieved by other CNNs. Our PaC-IIIB48 is able to increase the spelling accuracy achieved by other CNNs with up to 23%. The largest accuracy improvement occurs when comparing the spelling accuracy achieved by our PaC-IIIB48 with the spelling accuracy achieved by CCNN on epoch number $k = 2$.

For Dataset III-B-32 (see Table XIX), for all epoch numbers $k \in [1, 15]$, the spelling accuracy achieved by our PaC-IIIB32 is higher than the spelling accuracy achieved by other CNNs. Our PaC-IIIB32 is able to increase the spelling accuracy achieved by other CNNs with up to 15%. The largest accuracy improvement occurs when comparing the spelling accuracy achieved by our PaC-IIIB32 with the spelling accuracy achieved by CCNN on epoch number $k = 2,\ 3,\ 5$.

For Dataset III-B-16 (see Table XX), for epoch numbers $k \in [1, 4] \cup [6, 13]$, the spelling accuracy achieved by our PaC-IIIB16 is higher than the spelling accuracy achieved by other CNNs. Our PaC-IIIB16 is able to increase the spelling accuracy achieved by other CNNs with up to 29%. The largest accuracy improvement occurs when comparing the spelling accuracy achieved by our PaC-IIIB16 with the spelling accuracy achieved by BN3 on epoch number $k = 2$.

The aforementioned experimental results show the following. For Subject A, PaC-IIIA48, PaC-IIIA32 and PaC-IIIA16 achieves the highest spelling accuracy in most cases when signals are acquired using 48, 32 and 16 sensors, respectively. PaC-IIIA48, PaC-IIIA32 and PaC-IIIA16 have different network architectures. For Subject B, PaC-IIIB48, PaC-IIIB32 and PaC-IIIB16 achieves the highest spelling accuracy in most cases when signals are acquired using 48, 32 and 16 sensors, respectively. PaC-IIIB48, PaC-IIIB32 and PaC-IIIB16 have different network architectures. This answers **RQ1**, i.e, the design of CNNs should be aware of the number of sensors used in the P300 speller. More specifically, we need to design different CNNs when different number of sensors are used to acquire EEG singals. In addition, the CNNs designed by our method achieve the highest spelling accuracy in 86 out of 90 cases, which answers **RQ2**, i.e., our proposed method is able to design a sensor-aware CNN for the P300 speller with high spelling accuracy.

VII. CONCLUSIONS

In this paper, we perform an empirical study on the sensor-aware design of CNNs for the P300 speller. First, we show an example which motivates us for this investigation. Then, we propose the method for designing a CNN according to the given number of sensors used to acquire EEG signals. In this method, we automatically configure the parametric CNN we have devised according to the given number of sensors. Experimental results on six datasets show that in order to achieve high spelling accuracy, we need to design different CNNs when different number of sensors are used to acquire EEG signals. Experimental results also show the effectiveness of our sensor-aware CNN design method, i.e., the sensor-aware CNNs designed by our method achieves higher spelling accuracy than other CNNs in most cases.

REFERENCES

[1] E. W. Sellers and E. Donchin, "A P300-based brain–computer interface: initial tests by ALS patients," *Clinical Neurophysiology*, vol. 117, no. 3, pp. 538–548, 2006.

[2] L. Bonnet, F. Lotte, and A. Lécuyer, "Two brains, one game: design and evaluation of a multiuser bci video game based on motor imagery," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 5, no. 2, pp. 185–198, 2013.

[3] C.-T. Lin, S.-F. Tsai, and L.-W. Ko, "EEG-based learning system for online motion sickness level estimation in a dynamic vehicle environment," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 10, pp. 1689–1700, 2013.

[4] C.-T. Lin, B.-S. Lin *et al.*, "Brain computer interface-based smart living environmental auto-adjustment control system in UPnP home networking," *IEEE Systems Journal*, vol. 8, no. 2, pp. 363–370, 2014.

[5] R. Fazel-Rezai, B. Z. Allison, C. Guger, E. W. Sellers, S. C. Kleih, and A. Kübler, "P300 brain computer interface: current challenges and emerging trends," *Frontiers in neuroengineering*, vol. 5, p. 14, 2012.

[6] H. Cecotti and A. Graser, "Convolutional neural networks for P300 detection with application to brain-computer interfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 433–445, 2011.

[7] R. Manor and A. B. Geva, "Convolutional neural network for multi-category rapid serial visual presentation BCI," *Frontiers in Computational Neuroscience*, vol. 9, 2015.

[8] M. Liu, W. Wu, Z. Gu, Z. Yu, F. Qi, and Y. Li, "Deep learning based on batch normalization for P300 signal detection," *Neurocomputing*, vol. 275, pp. 288–297, 2018.

[9] H. Shan, Y. Liu, and T. Stefanov, "A simple convolutional neural network for accurate P300 detection and character spelling in brain computer interface." in *IJCAI*, 2018, pp. 1604–1610.

[10] A. Rakotomamonjy and V. Guigue, "BCI competition III: dataset II-ensemble of SVMs for BCI P300 speller," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 3, pp. 1147–1154, 2008.

[11] B. Blankertz, "BCI competition III webpage," *http://www.bbci.de/competition/iii/*, 2008.

[12] H. Cecotti, B. Rivet *et al.*, "A robust sensor-selection method for P300 brain–computer interfaces," *Journal of Neural Engineering*, vol. 8, no. 1, p. 016001, 2011.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[15] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Proc. of LION-5*, 2011, p. 507523.