Capsule Neural Network Based Error Correlation Potential Detection for EEG Topographies

Qingyu Wang*, Lan Wei, Zhengkang Zhou, Zhuoer Wang

School of Information Engineering, Wuhan University of Technology, Wuhan, China

*Corresponding author: wqy19990806@whut.edu.cn

Abstract. At present, the detection of error-related potentials is useful for the application of real-time error instruction correction techniques in brain-machine interface online systems. This paper, however, proposes a strategy for error-correlation potential detection based on EEG topographies, which translate the sequence of EEG topographies over time into a spatial position relationship between the features contained in different pictures. As the capsule network incorporates relative position relationships between features, i.e., positional information, a high classification accuracy can be achieved with a small dataset. Experimental evaluation has shown that the proposed method yields significant performance improvements compared to conventional processing methods.

Keywords: Error-correlation potential, brain-machine interface, EEG topography, capsule neural network.

1. Experimental background

Brain-computer interface technology enables direct use of the human mind to control external devices by monitoring information about the user's intentions in brain activity for external communication. Due to the weak and random nature of the EEG signals, processing and identifying the signals recorded by non-invasive brain-machine interfaces is complex and difficult and can lead to malfunctioning of external devices. Neurophysiological studies have shown that when an individual perceives an error, he or she can record an EEG signal specifically associated with the error response, known as error related negativity (ERN). Errors are identified to improve the reliability of the brain-machine interface.

The identification and classification of ERNs is a challenge. Because the stimulation paradigm of the ERN is not fixed, different tasks at the brain-machine interface can lead to phase shifts in the error potential components. In addition, the physiological and psychological differences between individuals will lead to poor generalisation of ERN classification.

Representative methods now include (1) calculating the power spectral density (PSD) of low frequency rhythms (4-8 Hz) using the stable frequency domain features of the ERN, followed by a single trial classification using a support vector machine (SVM), which has the disadvantage of being susceptible to noise interference. (2) Extracting the full channel time, frequency and air domain feature set of the EEG signal and then downscaling the feature set twice through the neural network to avoid dimensional disasters. (3) Linear Discriminant Analysis (LDA) is used to classify positive-error trials.
using the Directional Transfer Function (DTF) as a feature extraction method that reflects directional connectivity across multiple channels.

This paper proposes the use of deep learning to recognise ERNs by converting the original EEG signal into an appropriate intermediate representation while retaining the discriminatory information. After conversion, the intermediate representation can be decoded using a neural network structure.

There are two conventional methods of representing EEG signals, one showing the EEG waveform directly and the other using EEG topography, which reflects the spatial distribution of individual potential changes and reveals subtle abnormalities in the EEG that are more difficult to distinguish.

Machine vision is one of the most important branches of deep learning. Convolutional neural networks (CNNs) are widely used in machine vision. They use neurons as the basic computational unit to transfer information from layer to layer, but it is difficult to perceive consistency in images that have been transformed by rotation, pan, etc. In addition, CNNs use a large number of pooling operations, resulting in a large amount of valuable information being lost. In addition, CNNs use a large number of pooling operations, resulting in the loss of a large amount of valuable information, which seriously affects the recognition accuracy.

The capsule neural network is a revolutionary network architecture. Capsules are used instead of neurons in CNNs, making the output data a set of vectors. Dynamic routing algorithms are introduced so that the lower capsule in the network can predict the activation state of the higher capsule. The capsule neural network preserves the position information between objects in the image and takes into account the spatial hierarchy between objects, effectively overcoming the disadvantages of CNNs. Taking advantage of the capsule network, only a small number of training sets are required to achieve good training results. Of course, CNNs and capsule neural networks are not mutually exclusive, and the underlying layers of a capsule neural network can also be convolutional.

In order to develop an ERN detection method that is resistant to noise interference and has good generalisation, it is proposed that, firstly, at fixed intervals, the EEG waveform at that moment is transformed into an EEG topological map, thereby obtaining a time-dependent sequence of EEG topographies. The sequence of images over time can then be transformed into a spatial position relationship between the features contained in the different images. Finally, a capsule network is used to classify the images.

Based on the above background, this paper will present the spatial position relationship between the waveforms of EEG signals, their translation into picture features, and the use of capsule neural networks to identify EEG topographies.

2. Experimental methods

2.1. Construction of EEG topographies

EEG topography is the processing of EEG into an EEG power spectrum, graded according to the amount of power, which converts the EEG signal into a two-dimensional brainwave image that can be quantified. The quantitative signs can be numerical or coloured. The construction of an EEG topography consists of three steps: calculation of the power spectrum at the acquisition point, spatial interpolation and colour mapping.

(a) Modern spectrum estimation, including AR models, MA models, ARMA models, etc., are used to calculate the power spectrum at the acquisition point.

(b) As the EEG signals are collected from only a limited number of fixed points on the subject's head, the gaps between the points need to be filled in by interpolation using an interpolation formula that depends on the power of the points and the distance from the points to the point where the interpolation is calculated.

\[ f(X) = \frac{a/X^{X^2} + b/XB^2 + \cdots + p/XP^2}{1/X^{X^2} + 1/XB^2 + \cdots + 1/XP^2} \]
Where X is the location of the points to be interpolated, a, b... p represents the power value of each acquisition point, and XA, XB...XP is the distance from the points to each acquisition point to be interpolated.

(c) Each RGB component in the image is assigned an intensity value in the range 0-255. A power of 0 corresponds to blue, a power of maximum corresponds to red, and the other power values correspond to the colours in the middle band.

![Figure. 1 Gradient process diagram](image)

2.2. Principle of capsule networks

Capsule networks use the capsule as the basic computational unit. The capsule is a vector consisting of a series of neurons, the value of each neuron representing pose parameters such as scaling, orientation and position information, the length of the capsule representing the probability of the existence of a particular object and the transfer between capsules is achieved by a dynamic routing algorithm.

The capsule structure is shown in the figure.

![Figure. 2 Capsule structure diagrams](image)

The formulas and principles used for capsule networks include.

(1) Nonlinear vector compression function.

\[
V_j = squash(S_j) = \frac{\| S_j \|^2}{1 + \| S_j \|^2} \cdot \frac{S_j}{\| S_j \|^2} \rightarrow \frac{\| S_j \|}{1 + \| S_j \|^2} \cdot S_j \rightarrow \frac{\| S_j \|}{\sqrt{2}} \cdot S_j
\]

Where \( V_j \) is the output vector of the first capsule, \( j \) is the input vector of the first capsule, \( j \) is \( S_j \) the modulus of the vector and squash is the activation function. It preserves the direction of the input vector; it compresses the modulus of the input vector to between 0 and 1; and it measures the probability of the appearance of an entity by the size of the modulus of the vector, the higher the value of the modulus, the greater the probability.

(2) Matrix multiplication formula for input vectors.

\[\hat{u}_j = W_{ij} \cdot u_i\]
Where $u_i$ is the output of the capsule network from the previous layer, $u_i$ the length of the encoding represents the probability of detecting the corresponding feature and the orientation encoding represents some internal state of the detected object. $W_j$ a weight matrix (pose matrix) that encodes important spatial and other relationships between lower-level features and higher-level features, which can be seen as the output of each capsule neuron in the upper layer with different strength of connections to a neuron in the lower layer. $\hat{u}_j$. It is a prediction vector that represents an attempt by the previous layer of capsules to predict what the next layer of capsules will output.

(3) Weighted summation formula for the input vectors.

$$S_j = \sum_i c_{ij} \cdot \hat{u}_j$$

Where $c_{ij}$ is the coupling coefficient, which needs to be calculated using $b_{ij}$ the $b_{ij}$ update of the dynamic routing algorithm that is at the heart of the capsule network.

The formula for the coupling coefficient: $c_{ij} = \frac{e^{b_{ij}}}{\sum_k e^{b_{jk}}}$, needs to be guaranteed $\sum_j c_{ij} = 1$.

(4) Dynamic routing algorithm process.

All prediction vectors are obtained first, defining the number of iterations $r$ and the layer of the network to which the current input capsule belongs. For all input $i$ and output capsules $j$, a parameter $b_{ij}$ which is initialised to 0.

Then calculate the values of the vectors $c_{ij}$, i.e., all the routing weights of the capsule $i$, here using a softmax function (normalised exponential function) to ensure that each $c_{ij}$ non-negative sum is 1 $S_j$ $V_j$ obtained by activating the function.

The new weights are obtained by adding the dot product of the capsule output $V_j$ and the prediction vector $\hat{u}_j$ to the original weights $b_{ij}$ $b_{ij}$. The expression is: $b_{ij} \leftarrow b_{ij} + \hat{u}_j \cdot V_j$, and the dot product process is performed in order to check the similarity between the input and output of the capsule.

After updating the weights, the next iteration is carried out. Repeat the iteration $r$.

2.3. Topological map classification using capsule networks

The framework for the construction of the capsule neural network mainly consists of 4 convolutional layers to obtain the features of the images and 1-2 capsule layers to obtain the spatial position relationships of the image features. It also includes an InputLayer input layer, an average pooling layer, a Reshape layer, and a Lambda layer. The open-source machine learning platform tensorflow was used to build the network model as shown in the figure.

(a) The InputLayer is the input layer and the output is an array of $32 \times 32 \times 3$.

(b) Conv2D is a convolutional layer that extracts different features from the input by means of a convolutional operation, the first convolutional layer can only extract low-level features, and further layers can extract more complex features from the low-level features.

(c) The Reshape layer reshapes the output of Conv2D to obtain a set of vectors at each position, constituting the input to a low-dimensional capsule layer.

(d) Capsule layer (number of layers can be adjusted) outputs two vectors (dimensions can be adjusted) of dimension 16, each vector representing a classification result.

(e) The Lambda layer outputs a modulus of 2 vectors as probability values representing the probability of two classification outcomes in a biclass task, the higher probability value being the output outcome.
3. Experimental procedures

3.1. Experimental set-up

(1) Description of raw data, production of data sets

(a) The UK public physiological dataset Mahnob-HCI-Tagging Database was used as the raw data and the dataset can be accessed at https://mahnob-db.eu/hci-tagging/accounts/register/. The dataset is described below.

In experiments, the image is displayed together with a label at the bottom of the screen. In some cases, the label correctly describes the situation. However, in other cases the label did not apply to the media item. After each item, the green button was pressed if the participant agreed that the label applied to the media item, otherwise the red button was pressed. As shown in the diagram, the stimulus starts at around 30 seconds and an image with a 'correct' or 'incorrect' label is displayed at around 35 seconds to obtain a 'yes' or 'no' from the participant. The answer is "No".
Audio, video, gaze data and physiological data were recorded simultaneously throughout the experiment. A total of 32 EEG channels in the dataset were placed according to the International Electroencephalography Society's recommendation for a 10-20 system with 32 electrode positions.

(b) 140 experimental data were selected and the dataset was processed using MATLAB. As the ERN can be recorded in the central frontal region, the EEG channels were selected as FP1, F3, F7, FC5, FC1, FP2, FZ, F4, F8, FC6, FC2. common average was used to re-reference the data. the frequency band of the main ERN concentration was 4-12 Hz and the data were band-pass filtered from 4-16 Hz.

Time series of waveforms converted into EEG topographies. Taking into account the different human eye reaction times and the time it takes to understand a label, the data were intercepted at intervals of 30ms (too small an interval means that the topology of the brain remains essentially unchanged, too large an interval means that useful information is lost) and converted into an EEG topology up to 950ms. 25 images were obtained for each experiment, arranged in $5 \times 5$ order. As can be seen, the energy distribution states of the
EEG topographies in different regions at different moments are represented by different colours, so the time domain waveform was translated into the spatial position of the 25 images in relation to the different colour blocks.

The initial images were transformed into 3-channel images of size $32 \times 32$ and then into an array of $32 \times 32 \times 3$, labelled 1 and 0 respectively, corresponding to the participants' 'yes' or 'no' responses. The images were divided into 120 pieces of training data and 20 pieces of validation data for the dataset of this paper.

(2) Methodological assessment.

The methods presented in this paper are evaluated and compared with conventional treatment methods.

(a) Using capsule neural networks.
(b) Using the K Nearest Neighbors algorithm (KNN).
(c) Support Vector Machine (SVM) with Support Vector Machine.

(3) Parameter settings

Figure 6 Sequence of brain topographies over time after participants observed incorrect labelling

Figure 7 Participants observed the sequence of brain topographies after correct labeling over time
For capsule neural networks, three parameter settings are designed, batch_size is the total number of samples in a batch and epochs is the number of iterations.

(a) batch_size=64, epochs=9, number of capsule layers: 1 (output 2 16-dimensional capsules)
(b) batch_size=64, epochs=9, number of capsule layers: 2 (output 2 8-dimensional capsules)
(c) batch_size=64, epochs=9, number of capsule layers: 2 (output 2 14D capsules)

For the KNN algorithm, the value of the parameter K is increased from 1 to 12 in order.
For the SVM, some of the parameters are set as follows (python).

```
Sklearn.svm.SVC(C=1.0, kernel='rbf', gamma='auto', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, max_iter=-1, random_state=None)
```

Where C: penalty factor, kernel: kernel function type, gamma: kernel function coefficient, coef0: constant value of the kernel function, tol: residual convergence condition, cache_size: buffer size, max_iter: maximum number of iterations.

3.2. Results
acc means accuracy
val_loss is validation_loss
val_acc is validation_accuracy

3.2.1. Capsule neural networks.
(a) batch_size=64, epochs=9, number of capsule layers: 1 (output 2 16-dimensional capsules) Five experiments

**Chart. 1** Experiment 1.

| epochs | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| acc    | 0.4979 | 0.5482 | 0.6520 | 0.7368 | 0.7941 | 0.8556 | 0.9387 | 0.9415 | 0.9791 |
| val_loss | 0.2118 | 0.1810 | 0.2112 | 0.2175 | 0.2207 | 0.2569 | 0.2415 | 0.2874 | 0.3005 |
| val acc | 0.50 | 0.70 | 0.80 | 0.70 | 0.65 | 0.45 | 0.50 | 0.45 | 0.45 |

**Chart. 2** Experiment 2.

| epochs | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| acc    | 0.4989 | 0.5431 | 0.6415 | 0.7497 | 0.7816 | 0.8724 | 0.9781 | 0.9856 | 0.9981 |
| val_loss | 0.2116 | 0.1971 | 0.1889 | 0.1803 | 0.2075 | 0.2438 | 0.3482 | 0.3288 | 0.3429 |
| val acc | 0.55 | 0.80 | 0.65 | 0.60 | 0.50 | 0.50 | 0.45 | 0.45 | 0.40 |

**Chart. 3** Experiment 3.

| epochs | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| acc    | 0.4982 | 0.5056 | 0.5871 | 0.6527 | 0.6914 | 0.7490 | 0.8312 | 0.8598 | 0.9026 |
| val_loss | 0.2132 | 0.2126 | 0.1898 | 0.2215 | 0.2561 | 0.3179 | 0.3486 | 0.3451 | 0.3324 |
| val acc | 0.50 | 0.65 | 0.65 | 0.60 | 0.55 | 0.50 | 0.45 | 0.50 | 0.45 |

**Chart. 4** Experiment 4.

| epochs | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| acc    | 0.4962 | 0.5323 | 0.5833 | 0.6351 | 0.6883 | 0.7395 | 0.8096 | 0.8458 | 0.9475 |
| val_loss | 0.2126 | 0.1968 | 0.1827 | 0.1795 | 0.2124 | 0.2503 | 0.2837 | 0.3041 | 0.3356 |
| val acc | 0.50 | 0.70 | 0.75 | 0.65 | 0.55 | 0.50 | 0.50 | 0.40 | 0.50 |
### Chart. 5  Experiment 5.

| epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| acc    | 0.4955 | 0.5082 | 0.5215 | 0.5538 | 0.6089 | 0.6216 | 0.6903 | 0.8341 | 0.9073 |
| val_loss | 0.2130 | 0.2133 | 0.1988 | 0.2031 | 0.2785 | 0.3258 | 0.3372 | 0.3408 | 0.3495 |
| val_acc | 0.50 | 0.60 | 0.85 | 0.65 | 0.55 | 0.45 | 0.45 | 0.45 | 0.50 |

(b) batch size=64, epochs=9, number of capsule layers: 2 (output 2 8-dimensional capsules) Three experiments

### Chart. 6  Experiment (1)

| epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| acc    | 0.4981 | 0.5551 | 0.6926 | 0.7874 | 0.9038 | 0.9563 | 0.9792 | 0.9945 | 0.9961 |
| val_loss | 0.2133 | 0.2123 | 0.1645 | 0.2518 | 0.2465 | 0.2636 | 0.3528 | 0.2837 | 0.3823 |
| val_acc | 0.50 | 0.75 | 0.90 | 0.70 | 0.65 | 0.65 | 0.60 | 0.55 | 0.50 |

### Chart. 7  Experiment (2)

| epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| acc    | 0.4973 | 0.5641 | 0.6815 | 0.7487 | 0.8972 | 0.9446 | 0.9701 | 0.9955 | 0.9963 |
| val_loss | 0.2126 | 0.2094 | 0.1705 | 0.2452 | 0.2684 | 0.2975 | 0.3326 | 0.3562 | 0.3347 |
| val_acc | 0.50 | 0.75 | 0.90 | 0.80 | 0.60 | 0.65 | 0.55 | 0.55 | 0.60 |

### Chart. 8  Experiment (3)

| epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| acc    | 0.4985 | 0.5526 | 0.6937 | 0.7782 | 0.8048 | 0.9641 | 0.9845 | 0.9950 | 0.9977 |
| val_loss | 0.2113 | 0.2106 | 0.1934 | 0.1817 | 0.2349 | 0.2804 | 0.3259 | 0.2926 | 0.3065 |
| val_acc | 0.55 | 0.70 | 0.85 | 0.85 | 0.75 | 0.55 | 0.60 | 0.65 | 0.55 |

(c) batch size=64, epochs=9, number of capsule layers: 2 (output two 14-dimensional capsules) Three experiments

### Chart. 9  Experiment 1.

| Epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| Acc    | 0.4952 | 0.4905 | 0.4913 | 0.5103 | 0.6029 | 0.6984 | 0.8193 | 0.9016 | 0.9583 |
| val loss | 0.2134 | 0.2133 | 0.2133 | 0.1915 | 0.1810 | 0.1800 | 0.1826 | 0.2015 | 0.2463 |
| val acc | 0.50 | 0.50 | 0.50 | 0.80 | 0.75 | 0.75 | 0.75 | 0.65 | 0.60 |

### Chart. 10  Experiment 2.

| Epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| acc    | 0.4963 | 0.5025 | 0.5483 | 0.5778 | 0.6396 | 0.7354 | 0.8249 | 0.9138 | 0.9654 |
| val loss | 0.2131 | 0.2203 | 0.1951 | 0.1732 | 0.1870 | 0.1966 | 0.2384 | 0.2519 | 0.2802 |
| val acc | 0.50 | 0.60 | 0.65 | 0.85 | 0.70 | 0.70 | 0.60 | 0.65 | 0.65 |

### Chart. 11  Experiment 3.

| Epochs | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|--------|----|----|----|----|----|----|----|----|----|
| acc    | 0.4956 | 0.5018 | 0.5479 | 0.5763 | 0.6276 | 0.7028 | 0.8114 | 0.9345 | 0.9751 |
| val loss | 0.2137 | 0.2147 | 0.1850 | 0.1832 | 0.1937 | 0.2267 | 0.2245 | 0.2374 | 0.2633 |
| val acc | 0.50 | 0.55 | 0.75 | 0.70 | 0.70 | 0.70 | 0.65 | 0.50 | 0.60 |
3.2.2. **KNN algorithms**

**Chart. 12** Increase the value of parameter K from 1 to 12 in order

| K  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|
| accuracy | 0.35 | 0.35 | 0.35 | 0.55 | 0.50 | 0.55 | 0.65 | 0.50 | 0.60 | 0.55 | 0.60 | 0.55 |

3.2.3. **SVM algorithms.** Accuracy = 51%. Validation accuracy = 45%

4. **Summary**

The experimental results show that for a capsule network, the average peak accuracy is 84.2% for a set number of capsule layers of 2, and 77% for a set number of capsule layers of 1. Overfitting generally occurs if the epochs exceed 3, mainly due to the small number of images, and overtraining can easily lead to a degradation of its generalisation performance. For the K Nearest Neighbors algorithm, the peak accuracy is 65%. For the SVM algorithm, the classification accuracy is 45%. In terms of classification accuracy, the capsule network network is superior to the KNN and SVM algorithms. This may be due, in addition to the capsule network's sensitivity to positional information, to its better anti-noise performance, which makes it more efficient at learning truly meaningful discriminatory information.

Although the method proposed in this paper has achieved cutting-edge performance in ERN identification, there are still limitations to the current work. Firstly, for raw data, no EEG artefacts are removed during the pre-processing stage, which has an impact on classification accuracy. Then, due to the different timing of the ERN in different subjects, using the same time window for intercepting the EEG topographies may lead to the omission of important information. Finally, the structure of the neural network can be further optimised by modifying the number of layers of the convolutional layer to better extract the features of the EEG topography, and by modifying the number of layers of the capsule layer, and the dimensionality of the output capsule to better construct relationships between different features. Our future work will focus on addressing these limitations, using more test subjects, and performing online tests.

In conclusion, temporal information from EEG signals can improve the accuracy of ERN identification. However, existing methods for extracting EEG features neglect the further use of temporal information. The method proposed in this paper, firstly, converts the raw EEG signals into an EEG topological map. Then, a capsule neural network is used to learn the discriminative features of the EEG topogram and return test results. This strategy has significant performance improvements over traditional classification strategies, with experimental results on the Mahnob-HCI-Tagging Database dataset demonstrating cutting-edge performance. Therefore, the method is a promising study.

**National innovation and entrepreneurship training program for college students S202010497208**

**References**

[1] Bartsch R P, Liu K K, Ma Q D, et al. Three independent forms of cardio-respiratory coupling: transitions across sleep stages, 2014 [C]. IEEE.

[2] Liu K K, Bartsch R P, Lin A, et al. Plasticity of brain wave network interactions and evolution across physiologic states, 2015, 9.

[3] Liu K K, Bartsch R P, Ma Q D, et al. Major component analysis of dynamic networks of physiologic organ interactions, 2015 [C]. IOP Publishing.