Pattern Recognition 121 (2022) 108226

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Variable weight algorithm for convolutional neural networks and its applications to classification of seizure phases and types

Guangyu Jia [a], Hak-Keung Lam [a,*], Kaspar Althoefer [b]

[a] The Centre for Robotics Research, Department of Engineering, King's College London, Strand, London WC2R 2LS, United Kingdom
[b] The Centre for Advanced Robotics @ Queen Mary, Faculty of Science & Engineering, Queen Mary University of London, Mile End Rd, London E1 4NS, United Kingdom

ARTICLE INFO

Article history:
Received 13 March 2020
Revised 24 July 2021
Accepted 3 August 2021
Available online 4 August 2021

Keywords:
Variable weight convolutional neural networks
Machine learning
Seizure phase classification
Seizure type classification

ABSTRACT

Deep learning techniques have recently achieved impressive results and raised expectations in the domains of medical diagnosis and physiological signal processing. The widely adopted methods include convolutional neural networks (CNNs) and recurrent neural networks (RNNs). However, the existing models possess static connection weights between layers, which might limit the generalization capability and the classification performance of the models as the weights of different layers are fixed after training. Furthermore, to deal with a large amount of data, a neural network with a sufficiently large size is required. This paper proposes the variable weight convolutional neural networks (VWCNNs), which are a type of network structure employing dynamic weights instead of static weights in their convolutional layers and fully-connected layers. VWCNNs are able to adapt to different characteristics of input data and can be viewed as an infinite number of traditional, fixed-weight CNNs. We will show that the proposed VWCNN structure outperforms the conventional CNN in terms of the classification accuracy, generalization capability, and robustness when the inputs are contaminated by noise. In this paper, VWCNNs are applied to the classification of three seizure phases (seizure-free, pre-seizure and seizure) based on measured electroencephalography (EEG) data. VWCNNs achieve 100% test accuracy and show strong robustness in the classification of the three seizure phases, and thus show the potential to be a useful classification tool for medical diagnosis. Furthermore, the classification of seven types of seizures is investigated in this paper using the world's largest open source database of seizure recordings, TUH EEG seizure corpus. Comparisons with conventional CNNs, RNN, MobileNet, ResNet, DenseNet and traditional machine learning methods including random forest, decision tree, support vector machine, K-nearest neighbours, standard neural networks, and Naïve Bayes are being conducted using realistic test data sets. The results demonstrate that VWCNNs have advantages over other classifiers in terms of classification accuracy and robustness.

© 2021 Elsevier Ltd. All rights reserved.

1. Introduction

As one of the most important and prominent models in deep learning methods, convolutional neural networks (CNNs) have become highly successful in many application areas, such as computer vision, speech recognition and computer aided diagnosis. CNNs were developed from multilayer perception type networks —— these were firstly trained using stochastic gradient descent methods instead of manual feature selection [1]. In 1988, Rumelhart, Hinton and Williams presented a new method which later became famous, named backpropagation learning [2]. Backpropagation learning has proved to be a very efficient learning procedure capable of providing solutions to a wide range of tasks. One year later, LeCun et al. proposed LeNet which showed an application of backpropagation networks to hand-written digit recognition with better performance than other contemporary recognition methods [3].

Although LeNet has achieved satisfactory results, CNNs did not receive wide attention until 2012. In the ImageNet LSVRC-2012 competition, Alex, Hinton et al. successfully applied deep CNNs to ImageNet classification and achieved by that time the best results ever reported on those publically-available datasets [4]. Since the creation of AlexNet in 2012, deep CNNs have become a hot research field and have developed at a very fast rate.

Although CNNs have developed rapidly and clearly shown their benefits, there are still limitations in CNNs such as the need for a

* Corresponding author.
E-mail addresses: guangyu.jia@kcl.ac.uk (G. Jia), hak-keung.lam@kcl.ac.uk (H.-K. Lam), k.althoefer@qmul.ac.uk (K. Althoefer).

https://doi.org/10.1016/j.patcog.2021.108226
0031-3203/© 2021 Elsevier Ltd. All rights reserved.

large training dataset, static weights, and high computational costs. Existing CNNs have static weights which process all input data by using fixed weights between layers after training. The drawback of static weights is that it hinders the model in adapting to different characteristics of the input data and, hence, limiting the learning and generalization capabilities of the model.

Extending from conventional CNNs, we present here a novel type of CNNs whose weights in convolutional layers and fully-connected layers can change adaptively according to the input data. When compared to conventional CNNs and traditional classifiers, the proposed variable weight convolutional neural networks (VWCNNs) demonstrate better learning and generalization capabilities as well as showing more convincing results due to the improved robustness.

Motivated by the research in [5], the general architecture of the proposed VWCNNs is developed (shown in Fig. 1). To generalize the variable-weight structure proposed in [5], the VWCNN is designed in this paper by combining the original model and tuning blocks, in an attempt to achieve better performance and make CNN more generalized and flexible to different data types. VWNN in [5] utilizes matrix transformation, layer skipping, and multi-connection [5] to realise variable parameters, which has limited generalisability and can only be used in feed-forward neural networks. Furthermore, the backpropagation of VWNN was not discussed in [5].

For the first time, this paper gives the forward and backward propagations of variable-weight convolutional layers and fully-connected layers, which makes it possible to be applied to any deep learning algorithms. The proposed method is more straightforward, easier to generalise, and flexible to modify with the provided theories and code.

The proposed VWCNN consists of a tuned CNN and a tuning block containing fully-connected layers or convolutional layers or their combinations. Assume that the input data is $x(t)$, $\widehat{x}(t)$ is the input of the VWCNN obtained through data pre-processing (e.g., windowing, feature extraction) of the raw data $x(t)$. As shown in Fig. 1, the tuned CNN is used to classify inputs. The tuning layers are employed to generate weights $dw(t)$ which will be added to the original weights, $w(t)$. α is a learnable factor used for adjusting the impact of the adding term $dw(t)$. Therefore, the weights of the tuned CNN become $w(t)+\alpha dw(t)$. In doing so, the model is able to adapt to different inputs and become more robust, as the VWCNN can be viewed as an infinite number of CNNs.

In application aspect, this research aims to achieve effective and accurate predictions of epileptic seizure phases (seizure-free, pre-seizure, and seizure). Capturing specific brain activity related to epilepsy is difficult, and specialists were always needed for signals interpretation. Since the disease is unpredictable and the collected data is real-time, powerful tools are needed for assisting diagnosis. In recent years, more and more deep learning techniques have been applied to this domain. CNNs, autoencoder and long short-term memory (LSTM) approaches were employed in [6] for the classification of an epilepsy related EEG dataset named CHB-MIT and achieved 99.6% prediction accuracy. Stacked CNNs with adaptive and discriminative feature weighting fusion [7] were adopted to classify the same dataset and reached more than 84.0% accuracy for 5 classes. In [8], Discrete wavelet transform combined with CNNs was used to classify the EEG data and had 100% test accuracy.

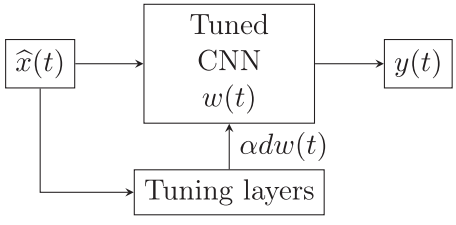

Fig. 1. A block diagram of VWCNN.

This paper employs VWCNNs to automatically classify three seizure phases using raw electroencephalography (EEG) data as input, aiming to improve the accuracy of diagnosis compared with state-of-the-art approaches. In the process of analysing EEG data for the classification of seizure phases, we learnt that the obtained EEG data are dynamic time series obtained from electrode measurements on the three-dimensional scalp surface —— this dataset is very different from the static images that CNNs have been most successful on. It is relatively small and insufficient for training in the context of deep learning. Considering the need of standard CNNs for millions of training samples to achieve robust results, a solution is come up with to cope with the low number of training samples. At the data processing stage, we adopted windowing and majority voting methods to process the inputs [9], which further improves the classification performance of the VWCNN.

The rest of the paper is organized as follows: Section 2 presents the preliminaries that will be used in this paper. In Section 3, the backpropagation of VWCNNs is introduced, two types of VWCNN (VWCNN-C and VWCNN-F) are proposed and training strategies are formulated accordingly. Section 4 shows the application of VWCNNs as well as the training strategies on epileptic seizures phase classification. Section 5 presents the results of the proposed method on the benchmark dataset TUH EEG seizure corpus (V1.4.0) [10]. Section 6 draws the conclusion of the paper.

2. Preliminaries

In this section, the preliminaries of this paper including the mathematical foundations and architectures of multilayer neural networks and CNNs are introduced.

2.1. Multilayer neural networks

Neural networks (NNs) are types of algorithms drawing inspiration from our understanding of the brain [11]. Through training procedures, NNs are configured for applications like function approximation, pattern recognition, data classification [12], etc. A three-layer feed forward network, which has 4 nodes in the input layer, 5 nodes in the hidden layer, and 4 nodes in the output layer, is shown in Fig. 2. In this network, assume that $x_i^{(1)}$ is the input, $z_h^{(2)}$ and $z_j^{(3)}$ are the inputs of activation function $f(\cdot)$, and $a_h^{(2)}$ and $a_j^{(3)}$ represent the outputs of activation function, where $i \in$

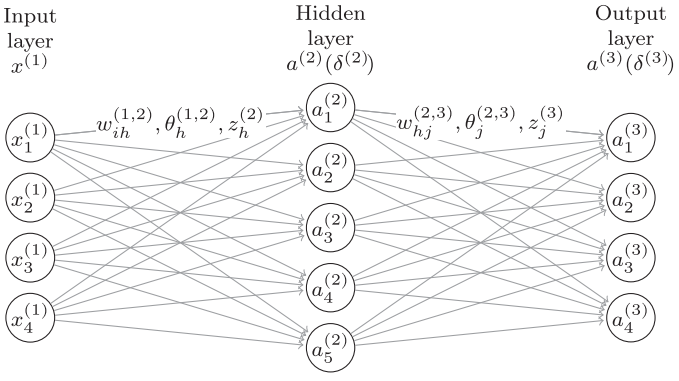

Fig. 2. A three-layer neural network including 4 nodes in the input layer, 5 nodes in the hidden layer and 4 nodes in the output layer.

$\{1,2,3,4\}$, $h \in \{1,2,3,4,5\}$, $j \in \{1,2,3,4\}$. δ (e.g., $\delta_h^{(2)}$ and $\delta_j^{(3)}$) represents the sensitivity of each node. Denote $w_{ih}^{(1,2)}, w_{hj}^{(2,3)}$ as the weights which map the inputs $x_i^{(1)}, a_h^{(2)}$ to the output $z_h^{(2)}, z_j^{(3)}$ respectively and denote $\theta_h^{(1,2)}, \theta_j^{(2,3)}$ as the corresponding bias.

Here, we have

$$z_h^{(2)} = \sum_{i=1}^{4} w_{ih}^{(1,2)} x_i^{(1)} + \theta_h^{(1,2)}, \quad a_h^{(2)} = f(z_h^{(2)}),$$
$$z_j^{(3)} = \sum_{h=1}^{5} w_{hj}^{(2,3)} a_h^{(2)} + \theta_j^{(2,3)}, \quad a_j^{(3)} = f(z_j^{(3)}). \tag{1}$$

The loss function (squared error function) is defined as:

$$E = \frac{1}{2n} \sum_x ||y(x) - a^{(3)}(x)||^2, \tag{2}$$

where n is the number of training samples, $y(x)$ represents the target values of the classification task, and $a^{(3)}(x)$ denotes the actual output of the network.

Backpropagation is a method used in artificial neural networks to calculate gradients that are needed in the calculation of the weights to be used in the network [13]. It is the most widely used method to decrease the error between the actual output and the desired output (i.e., the loss function in (1)) by adjusting the weights of nodes. The equations of backpropagation are as follows [14]:

$$\begin{cases} \delta^{(3)} = \nabla_a E \odot f'(z^{(3)})), \\ \delta^{(2)} = ((w^{(2,3)})^T \delta^{(3)} \odot f'(z^{(2)})), \\ \frac{\partial E}{\partial \theta_k^{(l,l+1)}} = \delta_k^{(l+1)}, \\ \frac{\partial E}{\partial w_{mk}^{(l,l+1)}} = a_m^{(l)} \delta_k^{(l+1)}, \end{cases} \tag{3}$$

where $\nabla_a E$ is a vector whose elements are the partial derivatives $\partial E / \partial a_k^{(3)}$, $\odot$ denotes the Hadamard product which is a type of elementwise multiplication, $\delta_k^{(l)}$ denotes the error of neuron k in layer l and $\delta^{(l)}$ represents the vector of errors associated with layer l. Similarly, $z^{(3)}, z^{(2)}, w^{(2,3)}$ denote the vectors of the corresponding components: $z_j^{(3)}$, $z_h^{(2)}$ and $w_{hj}^{(2,3)}$.

2.2. Correlation and convolution

This Section illustrates the formulas of general correlation and convolution operations which will be used later in deriving backpropagation equations for VWCNNs.

The correlation operation of a 2D image A and a kernel (filter) K is as follows:

$$K \otimes A(p,q) = \sum_{u=-N}^{N} \sum_{v=-N}^{N} K(u,v) A(p+u, q+v), \tag{4}$$

where the filter has $2N+1$ elements.

The convolution operation of a 2D image A and a kernel (filter) K can be written as:

$$K * A(p,q) = \sum_{u=-N}^{N} \sum_{v=-N}^{N} K(u,v) A(p-u, q-v), \tag{5}$$

where the filter has $2N+1$ elements.

3. Methodology

3.1. Variable weight convolutional neural networks (VWCNNs)

In fully-connected feed-forward neural networks, a large number of hidden nodes are necessary for the classification of image data, but this may largely increase the computational budget. CNNs overcome these difficulties through convolution operations, allowing the network to be deeper with fewer free parameters. However, existing CNNs have static parameters which process all input data using the same connection weights between layers. This demonstrates a drawback as the weights of different layers are fixed after training, drastically limiting the learning and generalisation capabilities of the trained network model. In this research, we present a type of novel CNN which has dynamic weights in the convolutional layers and fully-connected layers. In doing so, training a VWCNN is equivalent to training two neural networks: a CNN and a tuning network, and the training is accomplished simultaneously. The tuning network can be a conventional NN or a CNN. Due to the existence of the tuning structure, the weights of the VWCNN are varied according to the characteristics of the input data. In a sense, a VWCNN consists of an infinite number of CNNs, as the weights of VWCNN are dynamic and able to change adaptively to the input data.

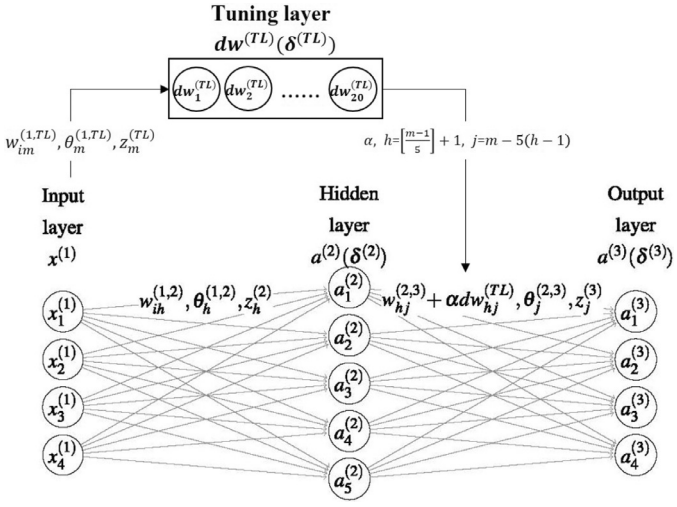

Fig. 3. An example of variable weight structure.

3.1.1. General structure of VWCNN

In form, the VWCNN is comprised of a CNN and the corresponding weight tuning structures. To illustrate the proposed method, we use the network structure shown in Fig. 2 as an example to modify it to have variable weights. As shown in Fig. 3, the original network provides the weights $w_{ih}^{(1,2)}$, $w_{hj}^{(2,3)}$, and the tuning layer provides the summands $dw_{hj}^{(TL)}$ to adjust the weights in the original network, where (TL) denotes the tuning layer. The weights in the second layer of this block are the sum of the original weights and the output of the tuning layer, i.e., $w_{hj}^{(2,3)}$ becomes $w_{hj}^{(2,3)} + \alpha dw_{hj}^{(TL)}$, where α is used for adjusting the impact of the adding term $dw^{(TL)}$. This design attempts to improve the generalization capabilities, classification performance and the robustness of the original CNN through giving the dynamic weights in its layers.

3.1.2. Two types of VWCNN: VWCNN-C and VWCNN-F

In this paper, we modify the convolutional layers and the fully-connected layers of CNN to enable variable weights. The first type of VWCNN is denoted by VWCNN-C, meaning weights in the convolutional layers are variable. An example of VWCNN-C is shown in Fig. 4, in which three convolutional layers' weights are modified by tuning layers.

As CNNs have a large number of parameters in fully-connected layers, in addition to modifying the convolutional layers, we also

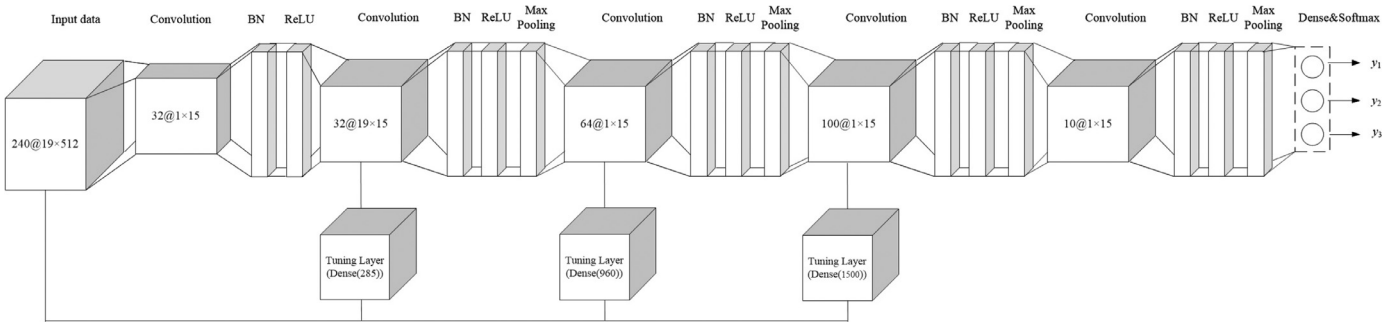

Fig. 4. The architecture of VWCNN-C which has variable weights in convolutional layers.

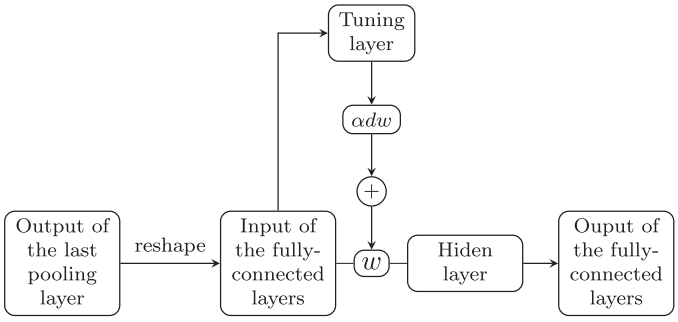

Fig. 5. The architecture of VWCNN-F which has variable weights in the fully-connected layers.

would like to know the impact of the variable weight structure for fully-connected layers. Figure 5 illustrates the structure having variable weights in fully-connected layers, which is denoted by VWCNN-F.

The application results described in the next sections show that the proposed VWCNN is able to improve the performance and robustness of standard CNNs. Also, the models introduced in Section 3.8 are employed for comparative study.

3.1.3. Backpropagation for VWCNN

To give the mathematical foundation of the proposed research, we will derive the forward propagation and the backpropagation of VWCNNs as follows. The example shown in Fig. 3 is used to illustrate the forward pass and backward pass of the VWCNN-F. Afterwards, we will discuss the backpropagation equations for VWCNN-C.

In Fig. 3, the number of nodes of the tuning layer is determined by the number of weights to be adjusted. In this example, $w^{(2,3)}$ has 20 components, which means there are 20 corresponding values to be adjusted. Hence, we have 20 nodes in the tuning layer.

In the implementation, we need to modify the original backpropagation process to make the proposed method realizable. First, derive the forward pass of network of the example shown in Fig. 3. From layer 1 to layer 2, the equations are given by

$$z_h^{(2)} = \sum_{i=1}^{4} w_{ih}^{(1,2)} x_i^{(1)} + \theta_h^{(1,2)}, \;\; a_h^{(2)} = f(z_h^{(2)}). \tag{6}$$

From layer 1 to the tuning layer, the forward pass is characterized by:

$$z_m^{(TL)} = \sum_{i=1}^{4} w_{im}^{(1,TL)} x_i^{(1)} + \theta_m^{(1,TL)}, \;\; dw_m^{(TL)} = f(z_m^{(TL)}), \tag{7}$$

where $z_m^{(TL)}$ denotes the input of activation function $f(\cdot)$ in the tuning layer and $dw_m^{(TL)}, m = 1, 2, \ldots, 20$, represents the output of activation function. Denote $w_{im}^{(1,TL)}$ as the weights which map the inputs $x_i^{(1)}$ to the output $z_m^{(TL)}$ and $\theta_m^{(1,TL)}$ is the corresponding bias.

In the third layer, the outputs of the second layer and the tuning layer are used to produce the final results $a^{(3)}$:

$$z_j^{(3)} = \sum_{h=1}^{5} (w_{hj}^{(2,3)} + \alpha dw_{hj}^{(TL)}) a_h^{(2)} + \theta_j^{(2,3)}, \;\; a_j^{(3)} = f(z_j^{(3)}), \tag{8}$$

as mentioned above, α is a parameter given to control the impact of $dw^{(TL)}$ to the original weights $w^{(2,3)}$. Since $dw^{(TL)}$ is a vector, the conversion of index from $dw_m^{(TL)}$ to $dw_{hj}^{(TL)}$ is derived (but not limited to) as follows:

$$h = [\frac{m-1}{5}] + 1, \;\; j = m - 5(h-1), \tag{9}$$

where the notation $[m]$ for $m \in \mathcal{R}$ means the largest integer no more than m.

In backpropagation, the most important step is the calculation of error $\delta^{(TL)}, \delta^{(2)}, \delta^{(3)}$. The equation for the error in the third layer is given by

$$\delta^{(3)} = \nabla_y E \odot f'(z^{(3)})) = (a^{(3)} - t) \odot f'(z^{(3)}). \tag{10}$$

Due to the use of the tuning layer, $\delta^{(2)}$ is changed according to the adding term $\alpha dw^{(TL)}$. The components of $\delta^{(2)}$ can be deducted as follows.

$$\begin{aligned} \delta_h^{(2)} &= \frac{\partial E}{\partial z_h^{(2)}} = \sum_j \frac{\partial E}{\partial z_j^{(3)}} \frac{\partial z_j^{(3)}}{\partial z_h^{(2)}} = \sum_j \frac{\partial z_j^{(3)}}{\partial z_h^{(2)}} \delta_j^{(3)} \\ &= \sum_j \frac{\sum_{h=1}^{5} (w_{hj}^{(2,3)} + \alpha dw_{hj}^{(TL)}) f(z_h^{(2)}) + \theta_j^{(2,3)}}{\partial z_h^{(2)}} \delta_j^{(3)} \\ &= \sum_j (w_{hj}^{(2,3)} + \alpha dw_{hj}^{(TL)}) \delta_j^{(3)} f'(z_h^{(2)}). \end{aligned} \tag{11}$$

Rewrite Eq. (11) to matrix form as

$$\delta^{(2)} = (w^{(2,3)} + \alpha dw^{(TL)}) \delta^{(3)} \odot f'(z^{(2)}). \tag{12}$$

Similarly, the equation for the error $\delta^{(TL)}$ in the tuning layer is given by

$$\begin{aligned} \delta_m^{(TL)} &= \frac{\partial E}{\partial z_m^{(TL)}} = \sum_j \frac{\partial E}{\partial z_j^{(3)}} \frac{\partial z_j^{(3)}}{\partial z_m^{(TL)}} = \sum_j \frac{\partial z_j^{(3)}}{\partial z_m^{(TL)}} \delta_j^{(3)} \\ &= \sum_j \frac{\sum_{h=1}^{5} (w_{hj}^{(2,3)} + \alpha f(z_m^{(TL)})) a_h^{(2)} + \theta_j^{(2,3)}}{\partial z_m^{(TL)}} \delta_j^{(3)} \\ &= \alpha \sum_j a_h^{(2)} \delta_j^{(3)} f'(z_{hj}^{(TL)}), \end{aligned} \tag{13}$$

where $z_{hj}^{(TL)} = z_m^{(TL)}$. The Eq. (13) can be rewritten as

$$\delta^{(TL)} = \alpha a^{(2)} \delta^{(3)} \odot f'(z^{(TL)}), \tag{14}$$

In this example, $\delta^{(TL)}$ is a matrix having the size of 5×4 based on Eq. (14), and is flattened to a vector of the size 20×1.

After obtaining errors in each layer, the updating of weights and bias for each layer can be achieved through the last two functions in (3).

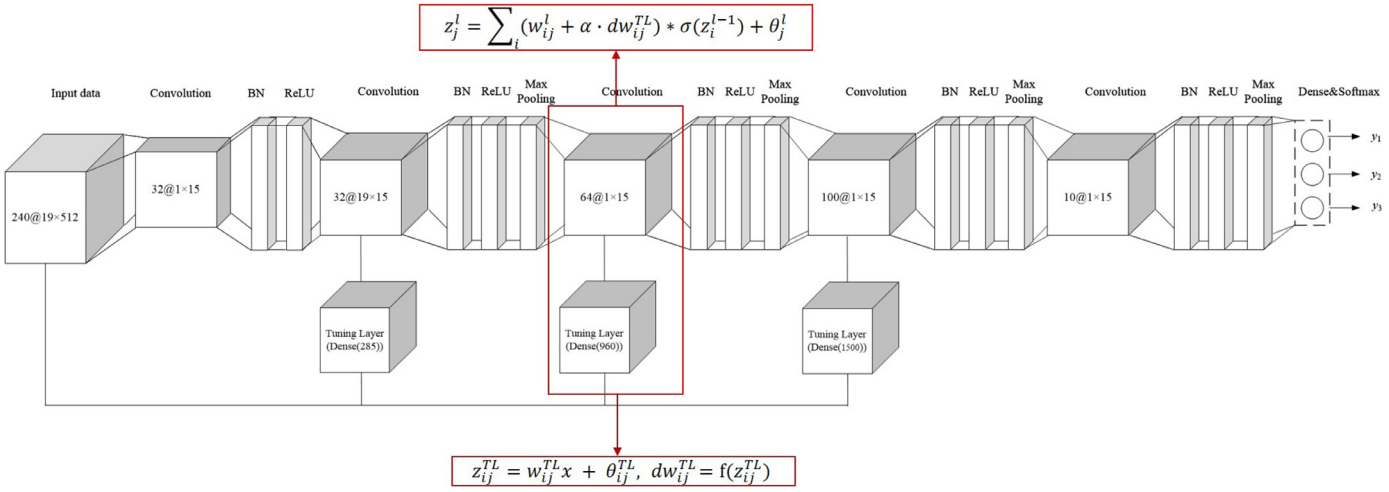

Fig. 6. Forward propagation of VWCNN-C which has variable weights in convolutional layers.

The deductions above show the forward pass and the backpropagation for VWCNN-F (or normal VWNN). With the similar principles, we can obtain the equations for VWCNN-C which has the variable-weight structure in convolutional layers.

$$\begin{aligned}\delta_j^l(p,q) &= \sum_i \sum_{\tilde{p}} \sum_{\tilde{q}} \frac{\partial E}{\partial z_i^{l+1}(\tilde{p},\tilde{q})} \frac{\partial(\sum_j (w_{ji}^{l+1} + \alpha dw_{ji}^{TL}) * \sigma(z_j^l) + \theta_i^l)}{\partial z_j^l(p,q)} \\ &= \sum_i \sum_{\tilde{p}} \sum_{\tilde{q}} \delta_i^{l+1}(\tilde{p},\tilde{q}) \frac{\partial(\sum_u \sum_v (w_{ji}^{l+1}(u,v) + \alpha dw_{ji}^{TL}(u,v))\sigma(z_j^l(\tilde{p}-u,\tilde{q}-v)))}{\partial z_j^l(p,q)} \\ &= \sum_i \sum_{\tilde{p}} \sum_{\tilde{q}} \delta_i^{l+1}(\tilde{p},\tilde{q})(w_{ji}^{l+1}(\tilde{p}-p,\tilde{q}-q) + \alpha dw_{ji}^{TL}(\tilde{p}-p,\tilde{q}-q))\sigma'(z_j^l(p,q)). \end{aligned} \tag{19}$$

Assume that $z_j^l(p,q)$ represents an element in the jth feature map z_j^l of the lth layer. Naturally,

$$z_j^l = \sum_i w_{ij}^l * \sigma(z_i^{l-1}) + \theta_j^l, \tag{15}$$

where w_{ij}^l is the filter related to the feature map z_i^{l-1} in layer $(l-1)$ and the feature map z_j^l in layer l, θ_j^l denotes the bias.

Through the convolution operation shown in Eq. (5), $z_j^l(p,q)$ can be written as

$$z_j^l(p,q) = \sum_i \sum_u \sum_v w_{ij}^l(u,v)\sigma(z_i^{l-1}(p-u,q-v)) + \theta_j^l. \tag{16}$$

For the VWCNN-C structure, we just need to derive the forward and back propagation for tuned layers, the update rule for weights in other layers are as same as that of normal CNNs.

As shown in Fig. 6, the forward pass for the lth layer is

$$\begin{aligned} z_j^l &= \sum_i (w_{ij}^l + \alpha dw_{ij}^{TL}) * \sigma(z_i^{l-1}) + \theta_j^l, \\ z_j^l(p,q) &= \sum_i \sum_u \sum_v (w_{ij}^l(u,v) \\ &\quad + \alpha dw_{ij}^{TL}(u,v))\sigma(z_i^{l-1}(p-u,q-v)) + \theta_j^l, \end{aligned} \tag{17}$$

and the tuning layer has the same forward equation as (7).

Next, we start from calculating the sensitivities (errors) of the convolutional layer having variable weights. With the chain rule, we have

$$\delta_j^l(p,q) = \frac{\partial E}{\partial z_j^l(p,q)} = \sum_i \sum_{\tilde{p}} \sum_{\tilde{q}} \frac{\partial E}{\partial z_i^{l+1}(\tilde{p},\tilde{q})} \frac{\partial z_i^{l+1}(\tilde{p},\tilde{q})}{\partial z_j^l(p,q)}, \tag{18}$$

where $z_i^{l+1}(\tilde{p},\tilde{q})$ represents the element(s) calculated using $z_j^l(p,q)$ in the forward propagation. Then putting the Eq. (17) into Eq. (18) and replacing the first term by the definition of sensitivity, we have

However, it can be found that the last equation is neither the general expression of convolution nor correlation operation given in Section 2.2. It can be easily converted as follows. Assume that $\hat{p} = \tilde{p} - p, \hat{q} = \tilde{q} - q$:

$$\begin{aligned} \delta_j^l(p,q) &= \sum_i \sum_{\tilde{p}} \sum_{\tilde{q}} \delta_i^{l+1}(\tilde{p},\tilde{q})(w_{ji}^{l+1}(\tilde{p}-p,\tilde{q}-q) \\ &\quad + \alpha dw_{ji}^{TL}(\tilde{p}-p,\tilde{q}-q))\sigma'(z_j^l(p,q)) \\ &= \sum_i \sum_{\hat{p}} \sum_{\hat{q}} \delta_i^{l+1}(\hat{p}+p,\hat{q}+q)(w_{ji}^{l+1}(\hat{p},\hat{q}) \\ &\quad + \alpha dw_{ji}^{TL}(\hat{p},\hat{q}))\sigma'(z_j^l(p,q)) \\ &= \sum_i (\delta_i^{l+1} \otimes (w_{ji}^{l+1} + \alpha dw_{ji}^{TL})(p,q))\sigma'(z_j^l(p,q)). \end{aligned} \tag{20}$$

Thus, for two related sensitivity maps δ_j^l and δ_i^{l+1}, the backward pass of errors can be written in matrix form as

$$\delta_j^l = \sum_i (\delta_i^{l+1} \otimes (w_{ji}^{l+1} + \alpha dw_{ji}^{TL})) \odot \sigma'(z_j^l), \tag{21}$$

which is equivalent to

$$\delta_j^l = \sum_i (\delta_i^{l+1} * \mathrm{ROT180}(w_{ji}^{l+1} + \alpha dw_{ji}^{TL})) \odot \sigma'(z_j^l), \tag{22}$$

where ROT180 denotes the operation that flipping the content 180 ° in order to perform cross-correlation instead of convolution operation. Then, the rule of updating the weights can be easily obtained using the same method:

$$\frac{\partial E}{\partial w_{ij}^l} = \frac{\partial E}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{ij}^l} = \delta_j^l * \mathrm{ROT180}(\sigma(z_j^{l-1})). \tag{23}$$

For the bias θ_j^l, noting that it is a scalar for each feature map, the updating rule should be the summation of all elements in δ_j^l:

$$\frac{\partial E}{\partial \theta_j^l} = \sum_u \sum_v \frac{\partial E}{\partial z_j^l(u,v)} \frac{\partial z_j^l(u,v)}{\partial b_j^l} = \sum_u \sum_v \delta_j^l(u,v). \tag{24}$$

Next, the backpropagation for the tuning layers is

$$\begin{aligned} \delta_{ij}^{TL} &= \frac{\partial E}{\partial (w_{ij}^l + \alpha dw_{ij}^{TL})} \frac{\partial (w_{ij}^l + \alpha dw_{ij}^{TL})}{\partial (dw_{ij}^{TL})} \frac{\partial (dw_{ij}^{TL})}{\partial z_{ij}^{TL}} \\ &= \delta_j^l * \mathrm{ROT180}(\sigma(z_j^{l-1})) \odot \alpha(f'(z_{ij}^{TL})), \end{aligned} \tag{25}$$

After obtaining the sensitivities, the update rule of corresponding weights and bias is as same as that of normal dense layers or convolution layers (if the tuning layer is composed of convolution blocks). The backpropagation algorithm for VWCNN is obtained.

3.2. Time complexity of VWCNN

The time complexity of conventional convolutional layers is as follows [15]:

$$T \sim O(\sum_{l=1}^{D} N_l^2 \cdot F_l^2 \cdot C_{l-1} \cdot C_l), \tag{26}$$

where the number of layers is denoted by D, l represents the lth layer, N_l is the spatial size of the output feature map (assume that it is square) in the lth layer, F_l is the spacial size of filters in the lth layer, C_l denotes the number of output channels in the lth layer.

In a VWCNN, the tuned layers have the time complexity as follows:

$$T \sim O(\sum_{l=1}^{D} N_l^2 \cdot (F_l^2 \cdot C_l + \alpha dw) \cdot C_{l-1}), \tag{27}$$

where dw is the output of corresponding tuning layer, and α is a constant used to adjust the impacts of the tuning layers. The time complexity of the corresponding tuning layer is:

$$T \sim O(I_l^2 \cdot F_l^2 \cdot C_l \cdot C_{l-1}), \tag{28}$$

where I_l is the spatial size of the input of the lth layer.

3.3. Implementation of VWCNN

In the implementation stage, the VWCNN is implemented by following the backpropagation procedures given in Section 3.1.3. However, given that programming the backpropagation of VWCNN or VWNN from scratch is time-consuming, laborious and difficult to replicate in practice, two VWCNN structures, VWCNN-C and VWCNN-F, can be implemented in various frameworks such as Mxnet and PyTorch, and the derivations and backpropagation of VWCNNs can be computed automatically within deep learning frameworks.

3.4. Data splitting

The sampling method used in this paper is the stratified sampling which is implemented by dividing the dataset into subgroups and selecting the samples from each group. As there are three seizure phases, to make the inputs contain a balanced amount of samples from each class, we randomly select training, validation and test samples from each class with the ratio indicated in Fig. 8, and combine them to form the final datasets. Therefore, the classifier is fed by the data consisting of an equal number of samples of each class.

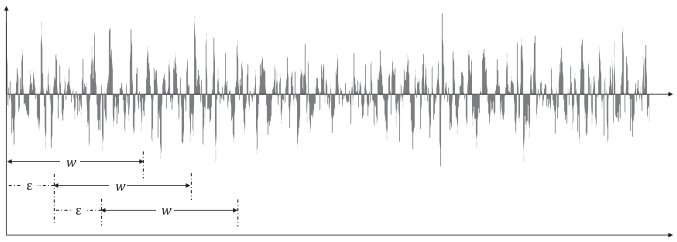

Fig. 7. Cropped training with window size w and increment ϵ.

3.5. Feature extraction

Feature extraction is an important method allowing us to extract useful information from raw data and to remove redundant information and interferences [16]. We use the feature extraction method as one of the data processing methods for the classification of three seizure phases. In order to successfully classify EEG data recorded during an epileptic seizure, appropriate feature vectors should be carefully selected. The commonly used features can be divided into three main groups: time domain, frequency domain, and time-frequency domain which is also known as time-scale representation [16,17]. In this paper, time domain and frequency domain features are adopted [18].

The EEG records referred to in this paper were obtained in the Peking University People's Hospital from 10 patients (6 males and 4 females) who had absence epilepsy, aged from 8 to 21 years old. More details of the dataset are provided in [5]. Following the data representation in [5], there are 19 columns of signals corresponding to the recordings from the 19 channels of the used Neurofile NT digital video EEG system. The data recording was done using a standard international 10–20 electrode placement (Fp1, Fp2, F3, F4, C3, C4, P3, P4, O1, O2, F7, F8, T3, T4, T5, T6, Fz, Cz and Pz), recording for each signal column 100 samples with each sample being 19×512 in size. The 16-bit analogue-to-digital converter has a sampling frequency of 256 Hz and, further, the data is filtered through a frequency band ranging from 0.5 to 35 Hz.

The research in [5] indicated that, out of the 19 channels, the most useful channels in EEG data were the 1st, 2nd, 3rd, 4th, 5th, 6th, 11th, 12th, 13th, 14th channels, containing the most significant information for the EEG signal classification problem. From each of the chosen channels, a feature vector consisting of 10 time-domain components is selected, including the absolute sum, second order norm, third order norm, fourth order norm, infinity norm, maximum value, minimum value, variance, mean value and root mean squared value of the elements in each channel.

3.6. Cropped training

Cropped training, which is also called the windowing approach, is an important preprocessing method which aims to generate more samples and improve the performance of classifiers. As shown in Fig. 7, it is implemented by sliding a window of the size w with increment ϵ over the original sample, to truncate the sample before and after the window while not modifying the contents within the window [19].

In Section 4, we use sliding windows within the set of training samples to generate a largely increased set of training samples. The size of the sliding window (w) is 19 and the increment ϵ is 1, therefore, every sample of the size 19×512 is cropped into 494 new samples and each sample's size is 19×19. As a result, those cropped samples become the new training data and have the same labels as the corresponding original sample.

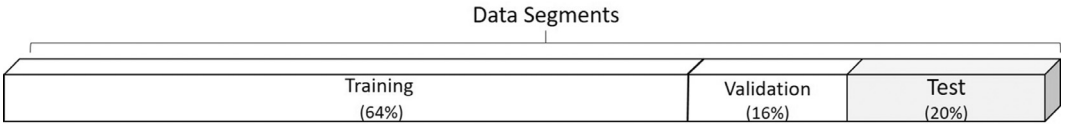

Fig. 8. Data split for training, validation, and testing.

Table 1
Classification results of proposed and comparative models for the raw EEG data.

Models	Training Accuracy	Test Accuracy
VWCNN-C	99.17%	**91.67%**
VWCNN-F	99.48%	**87.59%**
CNN	98.49%	87.50%
MobileNet	90.08%	74.00%
ResNet	92.12%	80.33%
DenseNet	66.65%	64.32%
RNN	83.26%	51.67%
Random Forest	99.17%	82.23%
SVM	100.00%	71.67%
Decision Tree	100.00%	67.33%
KNN	52.50%	45.00%
NN	100.00%	57.83%
Naïve Bayes	78.33%	76.67%

3.7. Majority voting

Majority voting is used for determining the classification results after using the cropped training strategy. As mentioned in Section 3.6, each original sample produces 494 new (cropped) samples. After classification, the final results of the original sample are predicted by majority voting of the corresponding 494 samples' results. In the scenarios that data are insufficient for training (300 training samples in this paper), the integration of the cropping method and majority voting provides a large number of inputs and can force the CNN to learn features from each cropped data rather than the complete sample, which has proved to be efficient for deep learning training [20].

3.8. The compared models

To compare the classification accuracy across a range of classifiers: MobileNet, ResNet, DenseNet, RNN, random forest, decision tree, SVM, KNN, neural network, Naïve Bayes classifier are employed for the EEG signal classification.

The specifications of the above methods are as follows. In the RNN, two GRU layers with 100 units each were adopted, which are followed by a dropout layer and 4 dense layers (optimizer: Adam; batchsize: 500; epoch: 10; loss function: categorical cross-entropy; the number of nodes in 4 dense layers are successively 500, 1000, 500, 3). The hyper parameters of VWCNN-C, VWCNN-F, and CNN are as follows: optimizer: adam, batchsize: 500, epoch: 40, loss function: categorical cross-entropy; initialization: Xavier. MobileNet, ResNet and DenseNet in Tables 1 to 4 respectively represent MobileNetv3_small [21], ResNet18_v2 [22], and DenseNet121 [23] of which the specifications are the same as that of VWCNN-C. For conventional machine learning models including Random Forest, SVM, Decision Tree, KNN, NN, and Naïve Bayes, the hyper parameters refer to the default specifications in scikit-learn. Noting that the feature extraction and MI method inevitably change the intrinsic characteristics of time-series data, therefore, the RNN are not adopted in these two cases.

4. VWCNN on seizure phase classification

This section introduces the application of VWCNNs to the classification of seizure phases.

Epilepsy is a common neurology disorder - a chronic disease of the brain causing sudden paradoxical discharge of cortical neurons, leading to significant impact on the living quality of sufferers as well as their carers [24]. The classification of seizure phases in epilepsy (seizure-free, pre-seizure and seizure) is clinically important for the reason that it allows the understanding and early detection of the transition towards a seizure occurrence. Specifically, seizure refers to the interval occurs during the first 2 s of the absence seizure; pre-seizure stands for the interval between 0 and 2 s before the occurrence of seizure; seizure-free is a 2-second interval before the onset of pre-seizure [25]. We refer the reader to [25] for detailed information of the EEG database used in this application.

Accurate classifications of seizure phases could enable sufferers and their carers to take precautions to either avoid the main seizure phase or, at least, to bring the epilepsy sufferer into a comfortable and risk-free position. In this research, we aim to employ the proposed approach for the classification of three seizure phases. Our approach uses the previously-used CNN-based detection methods as a starting point. However, the proposed approach have parameters that are dynamic, in contrast to the static weight paradigm employed in the CNNs.

We have focussed our study on four scenarios of inputs for the classification of seizure phases: 1) raw EEG data, 2) data after feature extraction, 3) data processed by the permutation mutual information (MI) method, 4) data processed by the cropped training approach. Different VWCNNs are established to deal with each case respectively. Fig. 8 shows the EEG data allocation for training and testing: 80% of the EEG data is used for training and the remaining 20% is used for testing. The validation dataset is used for parameter tuning for optimizing the selected model and avoiding over-fitting problem.

The experiments in this paper were conducted using NVIDIA P100 GPUs. The versions of softwares used in this paper are Tensorflow r2.2, Mxnet 1.6.0, python 3.7.3.

4.1. Two VWCNN models with their performance on the raw EEG data

CNN architectures are established for the classification of three seizure phases first, and the VWCNNs are determined based on the structures of CNNs. Two representative CNN models are shown in Figs. 9 and 10. And the main differences between the backbone CNNs include network depths and filter sizes. The architecture of the CNN is determined mainly through (I) referring to published literature about the successful CNN structures on similar datasets (EEG, ECG and EMG datasets); (II) trial and error; (III) taking into consideration the input size. In the first model shown in Fig. 9, most of the kernels in the convolutional layers are of the size 2×2. Inspired by the work in [20], we also utilize the filters having the size of 1×15 as shown in Fig. 10.

After comparing classification accuracy, CNN architecture 2 shown in Fig. 10 is adopted as the backbone of VWCNNs. VWCNN-C (see Fig. 4) has the same model structure with the CNN except the adjustable parameters. Particularly, the weights in the second,

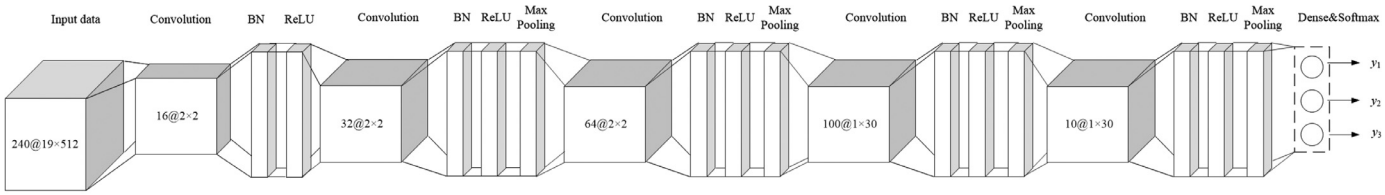

Fig. 9. CNN architecture 1.

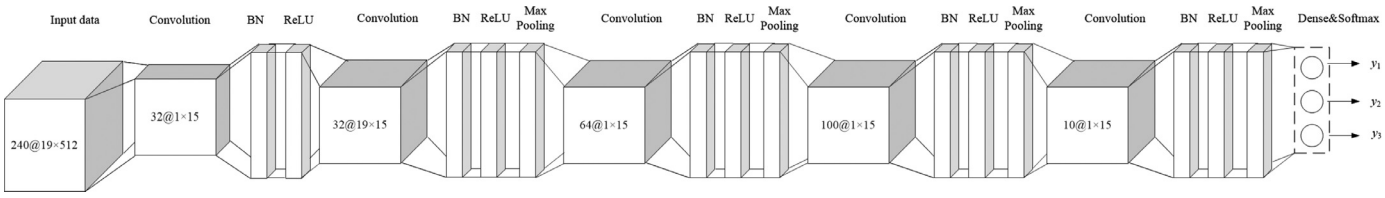

Fig. 10. CNN architecture 2.

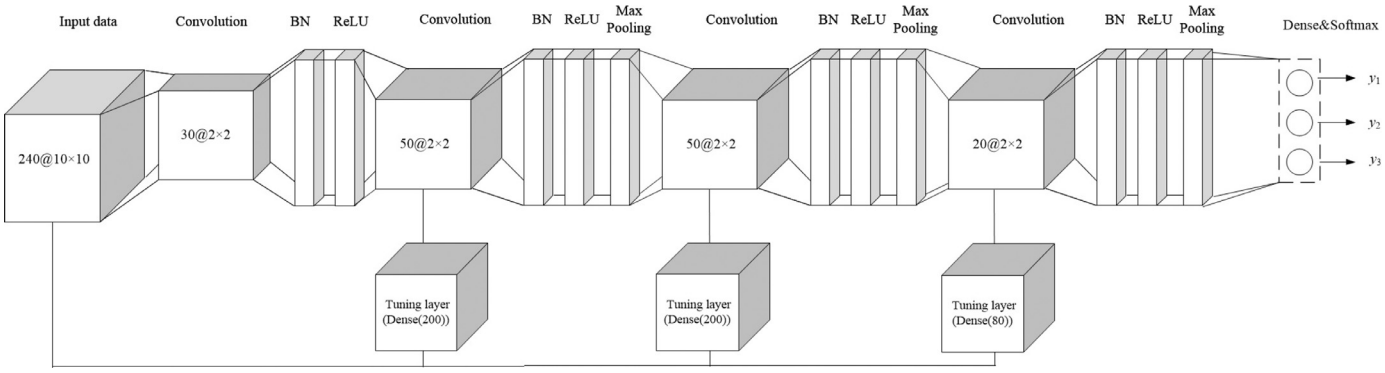

Fig. 11. VWCNN-C used for data after feature extraction.

the third and the fourth convolutional layers are determined by the weights of the original CNN, w, and the outputs of tuning layers, dw. The proposed variable weight structure makes it possible that parameters can change according to the inputs, since the original weight w is adjusted by a term αdw, where dw is the output of the tuning layer and determined by inputs. Therefore, the variable weight structure enables better model flexibility and stronger generalization ability compared to conventional CNNs. VWCNN-F has the same layers as the CNN except for the fully-connected layer.

The classification results of VWCNN-C, VWCNN-F, CNN and other comparative models are listed in Table 1, in which the test accuracy is obtained using the model with the best validation performance in the 5-fold cross validation. It can be found that VWCNN-C markedly increases the test accuracy of the model to 91.67%. VWCNN-F and CNN have a similar test accuracy of around 87.50%.

However, the classification performance shown in Table 1 is unsatisfactory as the best test accuracy is 91.67%. In addition, the problem of over-fitting occurs. In the following sections, data processing methods are employed to improve the classification performance of models.

4.2. Feature extraction

Feature extraction is employed to investigate the effectiveness of the selected features. After using the feature extraction method, the size of each sample is changed to 10×10 and each class has 100 samples. Considering the input size in this case, the CNN with 2×2 filters is employed to classify the EEG data. The structure of a VWCNN is shown in Fig. 11. Table 2 compares the classification performance of various classifiers after using the feature extraction method to process inputs.

Table 2
Classification results of proposed and comparative models for the data processed by feature extraction.

Models	Training Accuracy	Test Accuracy
VWCNN-C	88.70%	**83.33%**
VWCNN-F	87.33%	78.67%
CNN	81.22%	75.87%
MobileNet	59.20%	48.33%
ResNet	76.70%	74.00%
DenseNet	67.10%	51.00%
Random Forest	99.25%	**86.67%**
SVM	100.00%	41.67%
Decision Tree	100.00%	78.50%
KNN	94.58%	76.67%
Neural Network	71.54%	67.17%
Naïve Bayes	77.50%	78.33%

From Table 2, it can be seen that the average training and test accuracies of CNN, VWCNN-C and VWCNN-F are decreased compared to the case when raw EEG data is used as inputs. However, by comparing the results in Tables 1 and 2, it can be found that the feature extraction method improves the performance of random forest, decision tree, KNN, traditional NN and Naïve Bayes classifier. These results indicate that feature extraction is more suitable for conventional models, whereas deep learning models, especially the convolutional structures, are capable of automatically learning the sophisticated features from raw inputs.

To further improve the result, permutation mutual information (MI) method [26] is adopted in data processing. Through the MI

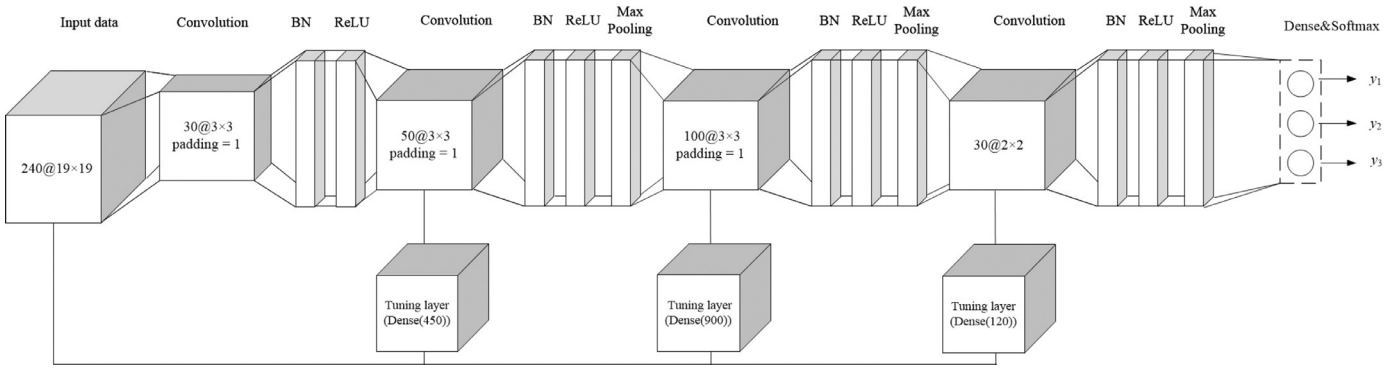

Fig. 12. VWCNN-C for data processed by cropped training and majority voting.

Table 3
Classification results of proposed and comparative models for the data processed by MI method.

Models	Training Accuracy	Test Accuracy
VWCNN-C	99.60%	**86.67%**
VWCNN-F	99.84%	**85.83%**
CNN	99.43%	83.00%
MobileNet	80.80%	69.67%
ResNet	92.50%	73.33%
DenseNet	81.70%	76.33%
Random Forest	99.54%	84.50%
SVM	82.92%	78.33%
Decision Tree	100.00%	80.33%
KNN	92.50%	80.00%
Neural Network	98.70%	85.60%
Naïve Bayes	85.83%	76.67%

Table 4
Classification results of proposed and comparative models for the data processed by cropped training approach.

Models	Training Accuracy	Test Accuracy
VWCNN-C	100.00%	**100.00**%
VWCNN-F	99.60%	**100.00**%
CNN	99.36%	98.33%
MobileNet	99.98%	98.33%
ResNet	100.00%	98.33%
DenseNet	99.60%	98.33%
RNN	99.87%	**100.00**%
Random Forest	99.48%	91.83%
SVM	100.00%	56.67%
Decision Tree	100.00%	95.47%
KNN	100.00%	83.33%
Neural Network	97.35%	94.83%
Naïve Bayes	72.43%	76.67%

method, the size of each sample becomes 19×19. The VWCNN shown in Fig. 11 is employed in this case.

The performance of the CNN and VWCNNs is listed in Table 3, from which it can be seen that the VWCNNs still outperform CNN, and VWCNN-C has the best classification performance. After using the MI method, the classification accuracy of the SVM is largely increased, and the accuracies of KNN and NN are also improved.

However, the highest classification accuracy is less than 90% in this case. In the next section, windowing method and majority voting are adopted, which aim to increase the number of training samples and improve the classification performance and robustness of models.

4.3. Cropped training

Considering a large number of input samples are preferable for the training of CNNs, the cropped training strategy introduced in Section 3.6 is employed for EEG data classification. The cropped training strategy uses sliding input windows within the samples to increase the number of inputs. The classification results of original samples are determined through majority voting of the predicted results of cropped samples.

As the cropped sample has the size of 19×19, the CNN and VWCNN-C are established accordingly (see Fig. 12). Based on the CNN shown in Fig. 12, VWCNN-F is designed by modifying the fully-connected layer to have dynamic weights. The structure of variable-weight fully-connected layer is shown in Fig. 5.

As shown in Table 4, the VWCNN-C as well as the RNN achieve satisfactory classification results. It can be found that after using windowing method, the classification accuracies of VWCNNs and CNN are significantly improved. In particular, the test accuracies of the VWCNN-C, VWCNN-F, and RNN reach 100%, which means that the three seizure phases can be successfully identified using these models. Furthermore, from the comparisons of the weighted-F1 score, the number of parameters, and training time illustrated in Table 5, it can be concluded that VWCNN models have advantages in classification accuracy and computational efficiency over the comparative methods. The results of a 5-fold cross validation are shown in Table 6. The increased performance also indicates that a large training dataset is necessary for deep learning models to learn important features. In addition, the comparative models also achieve better performance after using the windowing method.

Remark 1. It should be acknowledged that this research adopts record-wise classification, where the split of the dataset is at record level and EEG records from three seizure phases are equally and randomly allocated to five folds for cross-validation. Patient-wise classification, where the dataset is split based on patient-ID, is not adopted in this research because of the missing information of patient-ID in the dataset. Results in Tables 1–5 are used for comparing the effectiveness of the proposed VWCNN and conventional machine learning methods. Due to the record-wise classification adopted in this application, the results might be influenced by the correlated samples in the training and testing datasets.

4.4. Robustness

Robustness is also a significant factor to consider when assessing the performance of different neural network models. The robustness of classifiers having top-9 test accuracy are investigated. Assume that a raw input sample is represented by a matrix X whose size is (s_1, s_2), for the EEG data, $s_1 = 19, s_2 = 512$. After including the noise with its additive and multiplicative forms, the input becomes:

$$X' = n \cdot \text{randn}(s_1, s_2) \cdot X + m \cdot \text{randn}(s_1, s_2), \tag{29}$$

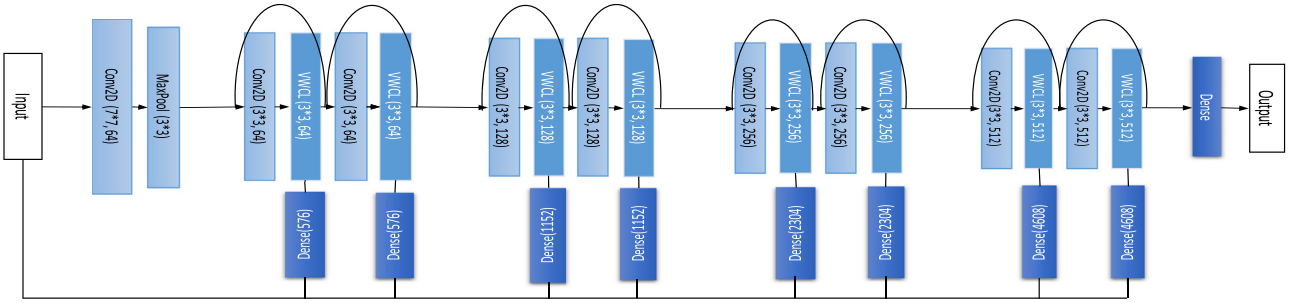

Fig. 13. The structure of variable-weight ResNet18. VWCL denotes the variable weight convolutional layer which is tuned by the corresponding dense layer.

Table 5
Classification performance of the deep learning models in Table 4.

PerformanceCNNs	VWCNN-C	VWCNN-F	CNN	ReNet	MobileNet	DenseNet	RNN
Weighted-F1 score (%)	100	100	98.3	98.3	98.3	98.3	100
Params (MB)	2.6	0.4	0.3	43.7	6.6	27.5	24.1
Training time (sec)	960	840	744	1900	1480	6480	805

Table 6
The validation and test accuracy obtained in a 5-fold cross validation. The test accuracy is obtained using the model which achieved the best validation accuracy in the 5-fold cross validation.

Folds	Cross Validation (data partition)					Validation Acc.	Test Acc.
Fold 1	20%	20%	20%	20%	20%	97.92%	100.00%
Fold 2	20%	20%	20%	20%	20%	93.75%	
Fold 3	20%	20%	20%	20%	20%	95.83%	
Fold 4	20%	20%	20%	20%	20%	89.58%	
Fold 5	20%	20%	20%	20%	20%	97.92%	

where X' is the contaminated input, n and m represent noise levels which are positive integers, randn() represents the function of Gaussian noise, and '$\cdot$' denotes the element-wise multiplication. In the robustness test, five value combinations of n and m are used: $n = 0.5, m = 0.5$; $n = 0.9, m = 0.1$; $n = 1.0, m = 0.0$; $n = 1.5, m = 1.5$; $n = 1.0, m = 2.0$.

Given that windowing method and majority voting largely improves the models' performance, windowing data are adopted in the robustness test. In particular, the model is trained using the uncontaminated windowing data. Afterwards, data perturbations shown in Eq. (29) are added to the windowing data, and models' accuracies are computed in response to the contaminated data.

For all value combinations of n and m listed above, the test accuracy of all classifiers are shown in Table 7. From the comparisons between the performance of all classifiers, VWCNN-C and VWCNN-F show to have a stronger robustness than other classifiers when contaminated data are presented. It indicates that VWCNNs are more likely to resist uncertainty and contaminations of the data.

5. VWCNN on TUH EEG seizure type classification

To further investigate the effectiveness of the proposed method and compare it with baseline methods, we applied the proposed method to the classification of seven seizure types using the world's largest publicly available dataset, the TUSZ database (v1.4.0) [10]. This dataset includes eight different types of seizures: Focal Non-Specific Seizure (FNSZ), Generalized Non-Specific Seizure (GNSZ), Simple Partial Seizure (SPSZ), Complex Partial Seizure (CPSZ), Absence Seizure (ABSZ), Tonic Seizure (TNSZ), Tonic Clonic Seizure (TCSZ), and Myoclonic Seizure (MYSZ) [27]. The research in [27] sets a benchmark for scalp EEG based multi-class seizure type classification with machine learning methods, which has been widely used in the literature for comparative purpose.

As analysed in [27], the seizure type of MYSZ is omitted in the classification since the number of samples of MYSZ seizures is too

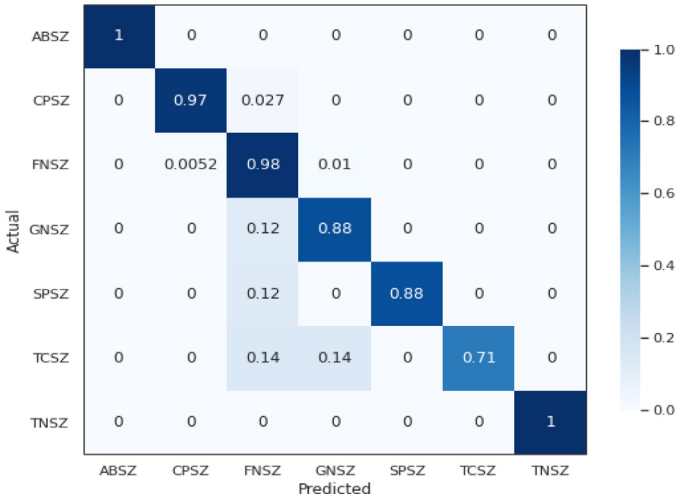

Fig. 14. Normalised confusion matrix for seizure type classification on TUSZ database using the proposed variable-weight ResNet18_v2.

Table 7
The performance of classifiers under noise $X' = n \cdot \text{randn}(s_1, s_2) \cdot X + m \cdot \text{randn}(s_1, s_2)$, $n = 0.9, 1.0, 1.5, 1.0$; $m = 0.1, 0.0, 1.5, 2.0$.

(n, m)	Test accuracy of cropped data with noise								
	CNN	VWCNN-C	VWCNN-F	RF	MobileNet	DT	ResNet	RNN	DenseNet
$n = 0.9, m = 0.1$	68.33%	73.33%	73.33%	73.33%	41.67%	70.00%	60.00%	50.00%	66.67%
$n = 1.0, m = 0.0$	68.33%	76.67%	75.00%	75.00%	43.33%	65.00%	61.67%	53.33%	66.67%
$n = 1.5, m = 1.5$	73.33%	81.67%	76.67%	73.33%	71.67%	63.33%	63.33%	51.67%	71.67%
$n = 1.0, m = 2.0$	68.33%	78.33%	76.67%	75.00%	45.00%	70.00%	61.67%	53.33%	68.33%

Patient ID	6520	1843	9107	1006	7623	281	1981	3760	6000	1324	1587	609	10427	21	8608	5034
Test Acc (%)	0.0	100.0	100.0	66.7	0.0	100.0	0.0	100.0	0.0	100.0	100.0	0.0	100.0	100.0	0.0	0.0

Fig. 15. Classification accuracy for per subject in testing dataset under patient-wise classification using the TUSZ database (v1.4.0).

Table 8
Cross-Validation performance of seizure type classification with the benchmark dataset in [27].

Models	Weighted-F1 score
K-NN [27]	0.883
SGD [27]	0.621
XGBoost [27]	0.844
CNN [27]	0.722
CNN [29,30]	0.901
Plastic NMN [28]	0.945
SeizureNet [30]	0.900
SAE [28]	0.673
LSTM [28]	0.701
CNN [28]	0.716
CNN [28]	0.826
CNN-LSTM [28]	0.831
CNN [28]	0.901
The proposed model	0.940

small for statistical analysis. Thus, seven types of seizures (FNSZ, GNSZ, CPSZ, SZ, TNSZ, TCSZ, SPSZ) are considered in this paper. To compare the VWCNN with baseline models provided in [27], we adopt the pre-processing features and the cross-validation scheme developed in [27] (method 1 with the window length $W_l = 1$, frequency band $f_{\max} = 24$, and $0.75W_l$ overlapping), which are also known as the IBM features for seizure detection (IBMFT). We refer the reader to [10] for the information about the dataset and to [27] for the data pre-processing method.

After conducting experiments with different CNN architectures, we found that the ReNet18_v2 [22] could achieve the best classification performance when used as the backbone of the VWCNN. The structure of the variable weight ResNet18_v2 is shown in Fig. 13. Eight convolutional layers in ResNet18_v2 were modified to have variable weights in this structure. The model specification is as follows: optimizer: adam and sgd; batchsize: 250; epoch: 15 (the first 10 epochs are with adam and the second 5 are with sgd); loss function: categorical cross-entropy; initialization: Xavier.

The classification results of seven seizure types with the proposed variable weight ResNet18_v2 are shown in Table 8 and Fig. 14. From Table 8, it can be seen that the proposed method improves approximately 4% of the weighted-F1 score compared to the baseline models except for the Plastic NMN in [28]. The main advantage of the variable-weight structure is that it can be applied to any deep learning models and improve their generalisability, robustness and classification performance, whereas the parameters of CNNs are fixed after training and cannot adapt to a wider range of inputs such as inevitable data perturbations. Morover, the comparative models in Table 8 are designed specifically for this dataset, while the proposed method is more flexible to modify.

However, all the published results in Table 8 are obtained using record-wise rather than patient-wise cross validation. As mentioned above, record-wise implementation could lead to biased results due to the highly correlated training and testing samples. Therefore, patient-wise classification is also implemented in this application to investigate model generalisation across patients. The benchmark dataset includes 2009 EEG records across 167 patients. In the patient-wise classification, 10% patients (16 patients) are randomly selected for testing, and the samples of the rest patients are used for training. The patient-wise classification results are shown in Fig. 15. From Fig. 15, it can be seen that 100% accuracy is reached for half patients. The average accuracy of patient-wise classification is comparatively lower than that of record-wise classification, which indicates that patient-wise data splitting makes it more challenging to improve classification performance.

6. Conclusion

This research presents a new type of CNNs whose weights can change adaptively to the input data, which differs from conventional CNNs with static weights after training. The forward pass and the back propagation of the proposed method are thoroughly discussed in this paper. Two types of VWCNNs are presented, which were applied to two EEG databases for seizure type classification and seizure phase classification. The main contributions of this paper are as follows. Firstly, the proposed variable-weight structure can be applied to any deep learning models. Secondly, different data processing methods are compared in this paper. The cropped training method combined with majority voting shows the effectiveness of improving most machine learning algorithms on a small dataset. Thirdly, the proposed VWCNNs including VWCNN-C and VWCNN-F outperform conventional CNNs and traditional classifiers in terms of classification accuracy and robustness when classifying three seizure phases. Fourthly, the variable-weight structure is applied to the classification of seven types of seizures using the open source database, TUH EEG seizure corpus (V1.4.0) [10]. Record-wise and patient-wise classifications are investigated in this application.

The weakness of this research is that the variable-weight structure inevitably increases the number of parameters and leads to larger computational costs. To address this problem, future work may focus on reducing the number of parameters of VWCNN. One possible approach is to use dimension reduction methods such as principal component analysis (PCA) or convolutional-autoencoder in tuning layers, in order to extract information from inputs and reduce the number of parameters between adjacent layers. Also, the design of VWCNNs can be more flexible in terms of having learnable hyperparameters such as the number of channels in each layer, the size of filters, etc.

Declaration of Competing Interest

No potential conflict of interest was reported by the authors.

Acknowledgement

This work was partly supported by King's College London and the China Scholarship Council and has been performed using resources provided by the Cambridge Tier-2 system operated by the University of Cambridge Research Computing Service funded by EPSRC Tier-2 capital grant EP/P020259/1.

References

[1] D.W. Ruck, S.K. Rogers, M. Kabrisky, Feature selection using a multilayer perceptron, J. Neural Netw. Comput. 2 (2) (1990) 40–48.
[2] D.E. Rumelhart, G.E. Hinton, R.J. Williams, et al., Learning representations by back-propagating errors, Cognit. Model. 5 (3) (1988) 1.
[3] Y. LeCun, B.E. Boser, J.S. Denker, D. Henderson, R.E. Howard, W.E. Hubbard, L.D. Jackel, Handwritten digit recognition with a back-propagation network, in: Advances in Neural Information Processing Systems, 1990, pp. 396–404.
[4] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[5] H.K. Lam, U. Ekong, B. Xiao, G. Ouyang, H. Liu, K.Y. Chan, S.H. Ling, Variable weight neural networks and their applications on material surface and epilepsy seizure phase classifications, Neurocomputing 149 (2015) 1177–1187.
[6] H. Daoud, M.A. Bayoumi, Efficient epileptic seizure prediction based on deep learning, IEEE Trans. Biomed. Circuits Syst. 13 (5) (2019) 804–813.
[7] J. Cao, J. Zhu, W. Hu, A. Kummert, Epileptic signal classification with deep EEG features by stacked CNNs, IEEE Trans. Cognit. Dev.Syst. (2019).
[8] R. Akut, Wavelet based deep learning approach for epilepsy detection, Health Inf Sci Syst 7 (1) (2019) 8.
[9] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.
[10] V. Shah, E. Von Weltin, S. Lopez, J.R. McHugh, L. Veloso, M. Golmohammadi, I. Obeid, J. Picone, The temple university hospital seizure detection corpus, Front. Neuroinform. 12 (2018) 83.
[11] B. Yegnanarayana, Artificial Neural Networks, PHI Learning Pvt. Ltd., 2009.
[12] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, et al., Recent advances in convolutional neural networks, Pattern Recognit. 77 (2018) 354–377.
[13] Y. Bengio, I.J. Goodfellow, A. Courville, Deep learning, Nature 521 (7553) (2015) 436–444.
[14] M.A. Nielsen, Neural Networks and Deep Learning, vol. 25, Determination Press, San Francisco, CA, USA:, 2015.
[15] K. He, J. Sun, Convolutional neural networks at constrained time cost, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5353–5360.
[16] A. Phinyomark, P. Phukpattaranont, C. Limsakul, Feature reduction and selection for EMG signal classification, Expert Syst. Appl. 39 (8) (2012) 7420–7431.
[17] M.A. Oskoei, H. Hu, Myoelectric control systems a survey, Biomed. Signal Process. Control 2 (4) (2007) 275–294.
[18] K. Englehart, B. Hudgin, P.A. Parker, A wavelet-based continuous classification scheme for multifunction myoelectric control, IEEE Trans. Biomed. Eng. 48 (3) (2001) 302–311.
[19] S. Moein, Medical Diagnosis using Artificial Neural Networks, IGI global, 2014.
[20] R.T. Schirrmeister, J.T. Springenberg, L.D.J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, T. Ball, Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human EEG, arXiv preprint arXiv:1703.05051 (2017).
[21] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al., Searching for MobileNetV3, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1314–1324.
[22] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European Conference on Computer Vision, Springer, 2016, pp. 630–645.
[23] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[24] R. Mohanraj, M.J. Brodie, Early predictors of outcome in newly diagnosed epilepsy, Seizure-Eur. J. Epilepsy 22 (5) (2013) 333–344.
[25] U. Ekong, H.K. Lam, B. Xiao, G. Ouyang, H. Liu, K.Y. Chan, S.H. Ling, Classification of epilepsy seizure phase using interval type-2 fuzzy support vector machines, Neurocomputing 199 (2016) 66–76.
[26] H.H. Yang, S. Amari, Adaptive online learning algorithms for blind separation: maximum entropy and minimum mutual information, Neural Comput. 9 (7) (1997) 1457–1482.
[27] S. Roy, U. Asif, J. Tang, S. Harrer, Seizure type classification using EEG signals and machine learning: Setting a benchmark, in: 2020 IEEE Signal Processing in Medicine and Biology Symposium (SPMB), IEEE, 2020, pp. 1–6.
[28] D. Ahmedt-Aristizabal, T. Fernando, S. Denman, L. Petersson, M.J. Aburn, C. Fookes, Neural memory networks for seizure type classification, in: 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE, 2020, pp. 569–575.
[29] Y. Hao, H.M. Khoo, N. von Ellenrieder, N. Zazubovits, J. Gotman, DeepIED: an epileptic discharge detector for EEG-fMRI based on deep learning, NeuroImage Clinical 17 (2018) 962–975.
[30] U. Asif, S. Roy, J. Tang, S. Harrer, SeizureNet: multi-spectral deep feature learning for seizure type classification, in: Machine Learning in Clinical Neuroimaging and Radiogenomics in Neuro-oncology, Springer, 2020, pp. 77–87.

Guangyu Jia received the B.S. and M.S. degrees from School of Mathematics, Shandong University, Jinan, China, in 2014 and 2017, respectively. She is currently working toward the Ph.D. degree in Center for Robotics Research, Department of Engineering, King's College London, London, UK. Her research interests include machine learning, deep learning, signal processing, image classification, control theory, medical and communication applications.

Hak-Keung Lam received the B.Eng. (Hons.) and Ph.D. degrees from the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong, in 1995 and 2000, respectively. During the period of 2000 and 2005, he worked with the Department of Electronic and Information Engineering at The Hong Kong Polytechnic University as Post-Doctoral Fellow and Research Fellow respectively. He joined as a Lecturer at King's College London in 2005 and is currently a Reader. He is an IEEE fellow. His current research interests include intelligent control, computational intelligence and machine learning. He has served as a program committee member, international advisory board member, invited session chair and publication chair for various international conferences and a reviewer for various books, international journals and international conferences. He was an associate editor for IEEE Transactions on Fuzzy Systems (2009–2018) and is an associate editor for IEEE Transactions on Circuits and Systems II: Express Briefs, IET Control Theory and Applications, International Journal of Fuzzy Systems and Neurocomputing; and guest editor for a number of international journals. He is on the editorial board of a number of international journals. He is a co-editor of two edited volumes: Control of Chaotic Nonlinear Circuits (World Scientific, 2009) and Computational Intelligence and Its Applications (World Scientific, 2012), and author/co-author of three monographs: Stability Analysis of Fuzzy-Model-Based Control Systems (Springer, 2011), Polynomial Fuzzy Model Based Control Systems (Springer, 2016) and Analysis and Synthesis for Interval Type-2 Fuzzy-Model-Based Systems (Springer, 2016).

Kaspar Althoefer Professor Althoefer is an experienced roboticist leading competitively funded research on soft robotics, intelligent micro-sensing systems and interaction dynamics modelling with applications in minimally invasive surgery, assistive technologies and human-robot interaction at Queen Mary University of London. He acquired in excess of *e*5.7M as Principal Investigator from national/international funding bodies and successfully completed 22 PhD projects. Professor Althoefer's research team, currently comprising 10 postdoctoral research associates and PhD students, is involved in funded collaborative research with leading London hospitals, European research organisations and international companies creating novel robot-assisted solutions for cardiac catheterisation, foetal ultrasound monitoring, tissue diagnosis using miniaturised stiffness sensors and ergonomically-optimised human-robot interaction. Over the last decade, the team has built a large portfolio of projects in application-oriented research for the healthcare and manufacturing sectors with funding from organisations such as EPSRC, European Commission (including coordination of two EU-projects), Wellcome Trust and UK-based charities, exceeding *e*30M and producing more than 250 peer-reviewed papers.