A Generalized Zero-Shot Learning Scheme for SSVEP-Based BCI System

Xietian Wang, Aiping Liu, *Member, IEEE*, Le Wu, *Member, IEEE*, Chang Li, *Member, IEEE*, Yu Liu, *Member, IEEE*, and Xun Chen, *Senior Member, IEEE*

***Abstract*—The steady-state visual evoked potential (SSVEP) has been widely used in building multi-target brain-computer interfaces (BCIs) based on electroencephalogram (EEG). However, methods for high-accuracy SSVEP systems require training data for each target, which needs significant calibration time. This study aimed to use the data of only part of the targets for training while achieving high classification accuracy on all targets. In this work, we proposed a generalized zero-shot learning (GZSL) scheme for SSVEP classification. We divided the target classes into seen and unseen classes and trained the classifier only using the seen classes. During the test time, the search space contained both seen classes and unseen classes. In the proposed scheme, the EEG data and the sine waves are embedded into the same latent space using convolutional neural networks (CNN). We use the correlation coefficient of the two outputs in the latent space for classification. Our method was tested on two public datasets and reached 89.9% of the classification accuracy of the state-of-the-art (SOTA) data-driven method, which needs the training data of all targets. Compared to the SOTA training-free method, our method achieved a multifold improvement. This work shows that it is promising to build an SSVEP classification system that does not need the training data of all targets.**

***Index Terms*—Brain-computer interface, steady-state visual evoked potential, generalized zero-shot learning.**

I. Introduction

Due to the high signal-to-noise ratio, steady-state visual evoked potential (SSVEP) is one of the promising paradigms for building user-friendly and multi-command brain-computer interfaces (BCIs) [1]. A typical SSVEP-based BCI system needs the user to stare at a flickering target. SSVEP signals can then be recorded using electroencephalography (EEG) electrodes. An algorithm is applied to search for the stared target based on the EEG data [2], [3].

The past few years have witnessed a dramatic improvement in the accuracy and information transfer rate (ITR) in SSVEP-based BCI systems. SSVEP is considered a periodic signal related to the stimulus frequency and can be classified using a training-free method. Canonical correlation analysis (CCA) is one of the first SSVEP classification algorithms [4] which needs little computational cost [5]. Since CCA can not make good use of the harmonic components in SSVEP, Chen et al. developed the filter bank CCA (FBCCA) [6]. By decomposing the EEG data into several subbands and proceeding separately, FBCCA has a better performance than CCA and becomes the state-of-the-art (SOTA) training-free method. Subsequent studies have focused more on algorithms with training data [7]. These algorithms can optimize the classification model for each subject and improve performance. The task-related component analysis (TRCA) is the most typical one [8]. For each stimulus target, TRCA finds the best weights for EEG electrodes and generates the template. TRCA classifies by finding the maximum correlation coefficient between the test data and the individual templates. The later study proposed the convolutional correlation analysis (ConvCA) method, which exceeds TRCA [9]. ConvCA applies two convolutional neural networks (CNN) to EEG data and templates separately. Then it uses a self-defined correlation layer and a fully-connected layer for classification. Recently, a study claimed to have the SOTA accuracy among data-driven methods [10]. It combined filter bank analysis with the deep neural network (DNN) in SSVEP classification.

In SSVEP classification, the training-based method generally has higher accuracy than the training-free method [11]. The existing training-based SSVEP classification method relies on correlation between training and testing data. Therefore, the training-based method can not identify classes that have not appeared in the training set (i.e., unseen classes). However, obtaining training data for all classes can cause fatigue, especially when there are a large number of classes in the system. Recent studies proposed SSVEP systems with more targets, such as 80-target [12] and 160-target [13]. Therefore, how to accurately classify unseen classes while using only

Manuscript received 5 July 2022; revised 2 December 2022; accepted 5 January 2023. Date of publication 10 January 2023; date of current version 3 February 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFF1200600; in part by the National Natural Science Foundation of China under Grant 32271431, Grant 61922075, Grant 82272070, and Grant 41901350; in part by the Fundamental Research Funds for the Central Universities under Grant KY2100000123; and in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2019-025. *(Corresponding authors: Aiping Liu; Xun Chen.)*

Xietian Wang, Aiping Liu, and Le Wu are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China (e-mail: xtwong@mail.ustc.edu.cn; aipingl@ustc.edu.cn; lewu@ustc.edu.cn).

Chang Li and Yu Liu are with the Department of Biomedical Engineering, Hefei University of Technology, Hefei 230009, China (e-mail: changli@hfut.edu.cn; yuliu@hfut.edu.cn).

Xun Chen is with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China, and also with the Institute of Dataspace, Hefei Comprehensive National Science Center, Hefei 230088, China (e-mail: xunchen@ustc.edu.cn).

Digital Object Identifier 10.1109/TNSRE.2023.3235804

certain classes (i.e., seen classes) as training samples becomes a problem.

The existing methods, such as DNN-SSVEP, classify in the data space. They require training data for all classes. We used convolutional neural networks to map the EEG and the stimulus of SSVEP to a latent space. Due to the nonlinear mapping of the network, the latent space is able to obtain the characteristics of the SSVEP signal only by the frequency and phase of the stimulus. Thus, we can obtain a representation of all classes in the latent space and classify them correctly. Since our task shares similarities with the generalized zero-shot learning (GZSL) in computer vision (CV) [1], [14], [15], we borrowed the data division approaches and variable descriptions from it to better describe our work.

In this study, we proposed a novel GZSL model for SSVEP classification. We used 8 unseen classes and 32 seen classes in the 40-target SSVEP system to demonstrate the performance, which is the common ratio (1 : 4) in GZSL missions (e.g., the Animals with attributes (AwA) dataset [14], the Oxford Flowers (FLO) dataset [16] and the ImageNet dataset [17]). In our GZSL model, the SSVEP signal serves as the feature, and the sine wave corresponding to a particular stimulus serves as the semantic. Our model contains three branches. The *Electrodes-Combination-Net* generates the latent space from averaged training data in the training stage. The *Extraction-Net* extracts the SSVEP components from the EEG data and projects them into the latent space. The *Generation-Net* uses the sine wave to generate templates in the latent space. We hypothesize that the SSVEP response of one subject is stable between neighboring stimulus frequencies. Thus, the *Generation-Net* can obtain enough information during the training stage though several classes are unseen. During the test time, we used the correlation coefficients of the outputs of the *Extraction-Net* and *Generation-Net* for classification.

We used two public SSVEP datasets [18], [19] to evaluate our model and compared the performance with the SOTA training-based and training-free methods. We also verified the impact of using different distributions and portions of unseen classes on performance. Since our method did not force the training data and test data to have the same data length, we also experimented with using different data lengths during training and testing. Our study illustrated that the proposed SSVEP classification algorithm can achieve high accuracy even when the training data of several classes are not provided. This result will contribute to the implementation of SSVEP systems with a large number of targets. The previous methods either had complete training for each target or were limited to training-free methods. However, our method can be implemented with only some training data available.

The rest of this paper is organized as follows. Section II introduces several relevant background studies. We present the application of CNN in SSVEP and the related studies of GZSL. In Section III, we described the datasets and our model. We also explained the training process. Section IV presents the experiment results. Section V and VI report the discussion and conclusion.

II. Related Work

In this section, we begin with a brief introduction to classification methods using convolutional neural networks (CNN) in SSVEP. We then present several existing GZSL methods and point out their applications to EEG.

A. Convolutional Neural Network in SSVEP Classification

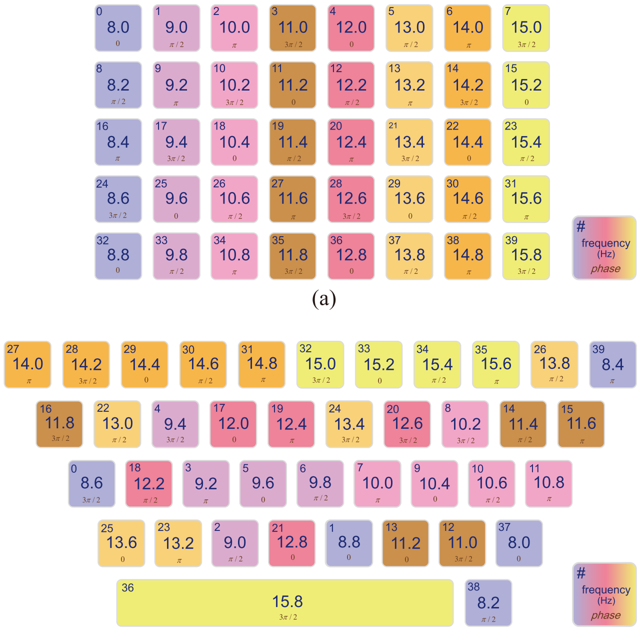
Deep learning (DL) methods require less prior knowledge of the domain and can optimize parameters automatically, thus achieving good results on challenging tasks [20], [21]. CNN is the most commonly used DL method in EEG classification tasks [22]. One of the representative works is EEGNet [23]. However, restricted by the small amount of data and large background noise, not until recently did CNN-based methods have not outperformed spatial filter methods (e.g., TRCA). ConvCA [9], which combines CNN with a correlation layer, performs better than the previous methods on the Benchmark dataset. While a recent study proposed Deep Neural Network for SSVEP (DNN-SSVEP) that claimed to have the SOTA accuracy on the same dataset [10]. However, both methods require training data for each target, which can cause severe fatigue for the user. Several review studies have pointed out that BCI systems should minimize training time to meet practical application requirements [11], [24]

B. Zero-Shot Learning

Lampert et al. pioneered zero-shot learning (ZSL) and proposed the direct attribute prediction (DAP) model for classifying images that did not appear in the training [14]. In ZSL, the dataset is divided into seen and unseen classes, and only seen classes are used for training. The conventional ZSL model uses the search space only containing unseen classes [25]. In contrast, generalized zero-shot learning (GZSL) uses the search space containing all classes for testing, which is more practical [26].

Most of the GZSL tasks use embedding-based models [27], [28], [29] or generative-based models [30], [31], [32]. Both models use semantic information to help transfer knowledge from seen classes to unseen classes. In an embedding-based model, there are three commonly used embedding spaces, including the semantic vector space, the feature vector space, and the latent space [33]. In our work, we used the scheme of latent space embedding since the feature vector having much noise, and the actual signal is hard to characterize in our semantic space. We projected the SSVEP feature and sine wave semantic into the same latent space using two separate network branches. Classification is carried out in the latent space.

The generative-based model learns to generate features using semantics from seen classes. Typically, these models use generative adversarial networks (GAN) [34], [35], [36] or automatic variational encoder (VAE) [37]. In the generative-based model, the generator uses semantic vectors to generate features close to those extracted by the feature extraction network. The classifier is then trained with generated and



(a)

(b)

Fig. 1. Stimulus interface layout. (a) The stimulus layout for the Benchmark dataset. (b) The stimulus layout for the BETA dataset. The different colours divide the adjacent frequencies into eight groups. The number in the top left corner indicates the order of the different targets in the dataset.

real features to distinguish between category and authenticity. Although generative methods were used in our network, the classifier was not trained with the generated data.

C. Generalized Zero-Shot Learning in EEG

Although GZSL has already made a splash in the field of image classification, its implementation in EEG has only recently begun. Hwang et al. proposed EZSL-GAN, which uses word2vec as semantic to classify the EEG signals evoked by different pictures [38]. Duan et al. proposed a method for Motor Imagery classification using GZSL [39]. The EEG features are extracted in their proposed network. Then the projection network uses the average feature vectors of each category as targets and the original features as the input to train. The classification in the test stage uses an outlier detector to distinguish between unseen classes and seen classes in the projected space. This approach can handle the single unseen class conditions, whereas our model needs to distinguish between multiple unseen classes.

III. Methods and Materials

A. Dataset

1) Benchmark Dataset: The Benchmark dataset is proposed by Wang et al. [18]. This dataset includes EEG data recorded from 35 subjects during the SSVEP experiment. The stimulus interface is a matrix of 5 × 8 targets, which is shown schematically in Fig. 1 (a). A total of six blocks of data were recorded. In each block, each target was presented once in random order. The subjects are required to pay attention to the target. In the experiment, there was a 0.5 s cue time before stimulus onset to instruct subjects to observe a specified target. Each target then flash for 5 s at specific frequencies f and phases p using

$$s(m,(f,p)_k)=\frac{1}{2}[1+sin(2\pi f(m/F_r)+p)], k\in C_A. \quad (1)$$

In the formula, m denotes the display frame and $F_r = 60$ Hz is the refresh rate of the display monitor. k is the order of stimulus, C_A is the stimulus set of all classes. The rest-state continues to be recorded for 0.5 s after the stimulus ended. A total 6 s length of data is recorded under a 1000 Hz sampling rate using a 64-electrode 10-20 EEG system, then filtered with a 50 Hz notch filter to remove powerline noise. Finally, the EEG data is downsampled to 250 Hz for storage

2) BETA Dataset: The BETA [19] and Benchmark dataset use the same settings for stimulus frequencies and phases while differing in other respects. Firstly, the BETA dataset uses a different arrangement of stimuli, as shown in Fig. 1 (b). In addition, the BETA dataset recorded data from 70 subjects, 4 blocks each. For the first 15 subjects, the stimulation time is 2 s, while for the rest subjects, the stimulation time is 3 s. The data also contains the 0.5 s rest-state before and after the stimulus. Moreover, the BETA dataset was acquired outside of the laboratory environment and has a lower signal-to-noise ratio (SNR).

B. Preprocessing

We used the data from 0.5 s to 5.5 s for SSVEP classification. A 10-order Butterworth filter with cutoff frequencies from 6 Hz to 90 Hz is applied to filter the data, followed by normalization. We applied a sliding window with a 75% overlapping to obtain more data segments following the approach mentioned in [9].

There are three inputs during the training stage. The *Training EEG Data*, the *Averaged Template*, and the *Sine Template*. The *Training EEG Data* is a single segment produced in the above procedure. The multi-channel *Averaged Template* is the averaged data segment of the same sliding window position and the same target. It is obtained using channel-by-channel averaging of EEG data from different trials of that class. Both the *Training EEG Data* and the *Averaged Template* correspond to the same target in a single input.

The *Sine Template* is the sine and cosine wave corresponding to the stimulus modulation function. In the training-free methods, the frequency and phase of the stimuli are treated as known conditions. And in training-based methods, such as extended-CCA [40], this condition is also used. This assumption of treating frequency and phase information as known semantic information is also reasonable in our study. All settings used to control target flicker are determined by the system at design time, and there is no signal outside these ranges. Thus, for those unseen classes, although we do not know the waveform of their EEG signals in training, we still know the set of their frequencies and phases.

We use the harmonic frequencies in generating the *Sine Template* and maintain the phase information using

$$\begin{aligned}Sine_h(t,(f,p)_k)&=sin(2\pi hf_k(t+\frac{p_k}{2\pi f_k}))\\&=sin(2\pi hf_kt+p_k\times h),\end{aligned}$$

$$\begin{aligned} Cosine_h(t,(f,p)_k) &= cos(2\pi h f_k(t+\frac{p_k}{2\pi f_k})) \\ &= cos(2\pi h f_k t + p_k \times h). \end{aligned} \tag{2}$$

In this formula, $h = 1, 2, \cdots, N_h$ is the order of harmonics. N_h denotes the number of harmonics. All harmonics have the same delay as the fundamental frequency. Both $Sine$ and $Cosine$ waveforms are used in constructing the *Sine Template*. We also applied the sliding window to the *Sine Template*. In a single input, we let the *Sine Template* align with the *Training EEG Data* and the *Averaged Template* using the same sliding window.

During the testing stage, only two inputs, the *Testing EEG Data* and the *Sine Template* are offered to the model. The construction of the *Testing EEG Data* is as same as the *Training EEG Data* whereas the *Testing EEG Data* have not appeared during the training time. The *Testing EEG Data* and the *Sine Template* are also aligned using the same sliding window in the testing stage.

C. The Proposed GZSL Architecture

We assumed that the SSVEP responses can be projected to a common latent space. The collected *EEG data* is a sample in this latent space after adding perturbations. Different *Sine Templates* can be projected to this latent space by one projection function. In this latent space, sample points corresponding to the same stimulus are clustered together. Therefore, once the projection function of *Sine Template* is learned, it is possible to classify sample points of arbitrary stimuli in the latent space. Thus, our model contains three parts. Two of them mapping the *EEG Data* and *Averaged Template* back to latent space (i.e., the *Electrodes-Combination-Net* and the *Extraction-Net*), and another one mapping *Sine Template* into latent space (i.e., the *Generation-Net*). This network performs classification task in latent space using correlation analysis between the generated and acquired sample. The structure of our proposed GZSL model is demonstrated in Fig. 2 (a). Three branches serve different functions. The *Electrodes-Combination-Net* (denoted as $C(\cdot)$) uses the *Averaged Template* to generate the latent vector space. The *Extraction-Net* (denoted as $E(\cdot)$) and *Generation-Net* (denoted as $G(\cdot)$) project the *EEG data* and *Sine Template* into the latent vector space respectively. The input *EEG data* (denoted as $Z_i \in \mathbb{R}^{1\times N_C\times N_S}$) and *Averaged Template* (denoted as $\overline{Z_i} \in \mathbb{R}^{1\times N_C\times N_S}$) have the same dimension of $1 \times N_C \times N_S$. $i \in C_S$ is the class label of stimulus for training, $C_S \subseteq C_A$ is the stimulus set of seen classes. N_C is the number of the recorded electrodes and N_S is the number of sampling points. *Sine Template* (denoted as $Y \in \mathbb{R}^{(2\times N_h)\times N_f\times N_S}$) is composed of multiple sine and cosine templates defined by

$$Y_k = Y_{(f,p)_k} = \begin{bmatrix} Sine_1(t,(f,p)_k) \\ Cosine_1(t,(f,p)_k) \\ \vdots \\ Sine_{N_h}(t,(f,p)_k) \\ Cosine_{N_h}(t,(f,p)_k) \end{bmatrix}, Y_k \in \mathbb{R}^{(2\times N_h)\times N_S}. \tag{3}$$

N_f is the number of all targets in the dataset. Here we used three harmonics in our model. Y is sorted according to its fundamental frequency (i.e., $f_1 < f_2 < \cdots < f_k < \cdots < f_{N_f}$). To prevent confusion, we use k as the subscript for sorted Y. We use i as the subscript for unsorted Y, i corresponds to the class label. The output of *Electrodes-Combination-Net* and *Extraction-Net* are $X \in \mathbb{R}^{1\times 1\times N_S}$ and $T \in \mathbb{R}^{1\times 1\times N_S}$, while the output of the *Generation-Net* is $S \in \mathbb{R}^{1\times N_f\times N_S}$.

$$\begin{aligned} S &= G(Y) \\ &= \begin{bmatrix} G_1([Y_1,\cdots,Y_{N_f/n}]^T) \\ G_2([Y_{N_f/n+1},\cdots,Y_{2\times N_f/n}]^T) \\ \vdots \\ G_n([Y_{(n-1)\times N_f/n+1},\cdots,Y_{N_f}]^T) \end{bmatrix} \\ &= [G(Y_1), G(Y_2),\cdots,G(Y_k),\cdots,G(Y_{N_f})]^T. \end{aligned} \tag{4}$$

n is the number of frequency groups of the *Generation-Net*.

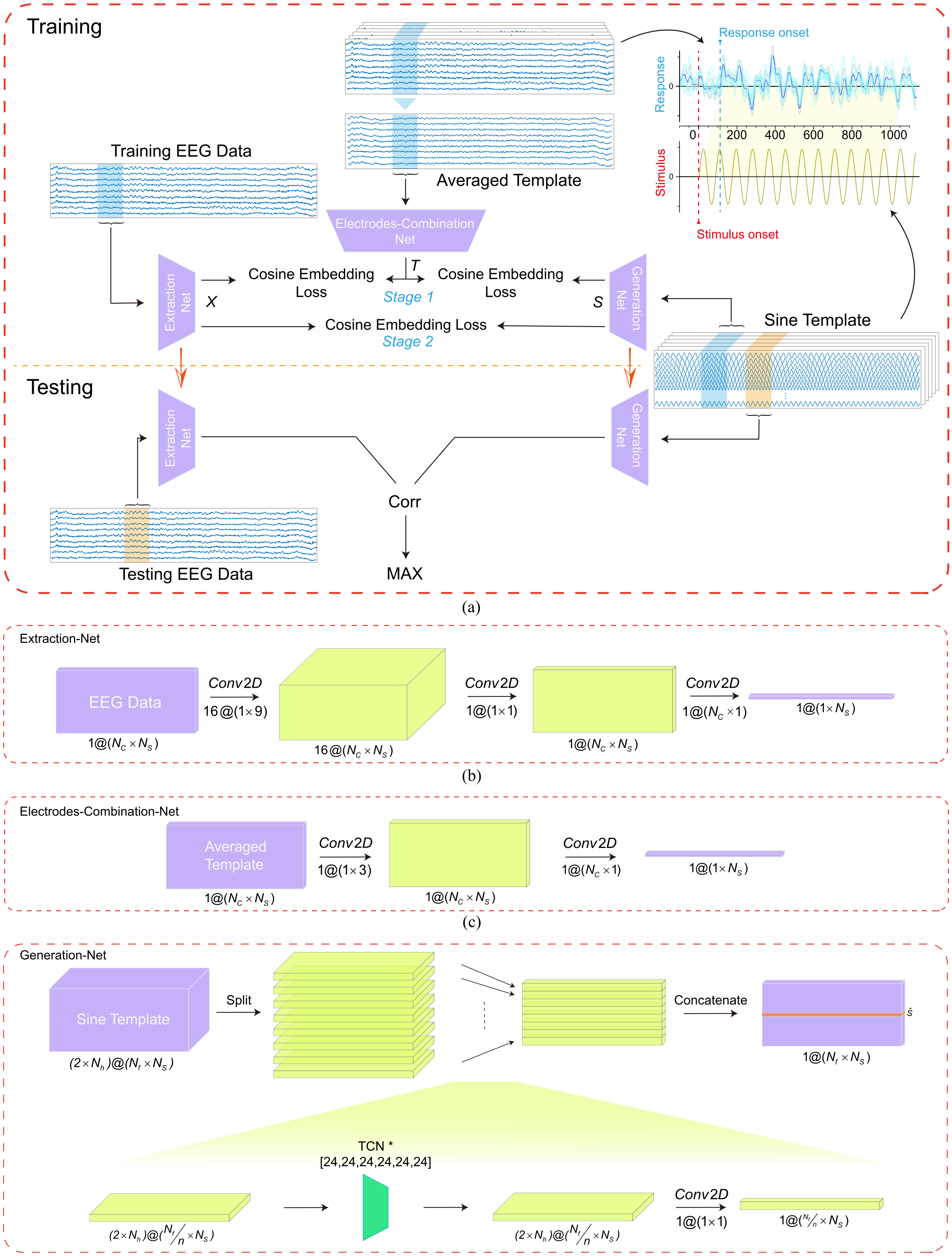
In the testing stage, the network calculates the cosine similarity between the outputs of *Extraction-Net* and *Generation-Net*, and the class corresponding to the maximum correlation is the classification result.

1) Extraction-Net: We used the same network structure for the *EEG data* as for one of the branches in the [9]. The detailed structure of *Extraction-Net* is illustrated in Fig. 2 (b). The first convolution layer serves as the bandpass filter dividing the signal into 16 subbands. The second and third convolution layers perform subband merging and electrode merging. The output of the *Extraction-Net* is denoted as $X \in \mathbb{R}^{1\times 1\times N_S}$.

2) Electrodes-Combination-Net: The structure of the *Electrodes-Combination-Net* is shown in Fig. 2 (c). As the *Averaged Template* is obtained by averaging, it contains little noise, and its processing should be as simple as possible so that the details of the signal are preserved. We used an 1×3 convolution kernel to filter the signals acquired from each electrode separately. Then, the sampled signals from the electrodes are combined to further improve the signal quality. The output is denoted as $T \in \mathbb{R}^{1\times 1\times N_S}$.

3) Generation-Net: This network generates the tensor $S \in \mathbb{R}^{1\times N_f\times N_S}$ in the latent space using the input *Sine Template*. We assumed that the response for stimuli of neighboring frequencies is similar and can be generated using the same network. Therefore, we divided the neighboring frequencies into groups and used different colors to indicate their position in the stimulus interface (see Fig. 1). For the 40-target dataset we used, we divided the frequencies evenly into 8 groups (i.e., $n = 8$). In addition, previous work has shown that there is a visual latency of approximately 100 ms to 200 ms between stimulus onset and response signal. References [19], [41], [42], [43]. This latency is related to the stimulus frequency. The higher the stimulus frequency the more the latency. Since the response is time-locked to a specific stimulus, we can use a specific network to learn the amount of latency and to ensure signal causality. Thus, we need the network to learn the amount of latency.

The temporal convolution network (TCN) [44], which uses causal convolution layers, meets our needs. The output at each moment in the TCN network is only related to the input at the previous moments, thus allowing for latency. Each group of *Sine Template* passes through a separate TCN network.



(a)

(b)

(c)

(d)

Fig. 2. Structure of the proposed GZSL model. (a) The overall structure of the model. We plot the waveform of the SSVEP response at 8 Hz stimulus in the upper right corner. The response is taken from subject 22 in the Benchmark dataset. We have labeled the starting point of the stimulus and response for showing the latency between the two signals. (b) The structure of the *Extraction-Net*. (c) The structure of the *Electrodes-Combination-Net*. (d) The structure of the *Generation-Net*. $n = 8$ is the number of frequency groups. The numbers below the cube represent the shape of tensors. No fully connected layers are used in the network, and only the last layer uses batch normalization. Except for the *TCN** part in *Generation-Net*, each convolution layer shown in the figure uses the linear activation function. Each layer uses zero padding to keep the width of the input and output the same as N_S.

We have made some modifications to the TCN block proposed in [44] (Fig. (3)). Firstly, the TCN proposed in [44] is suitable for the case of one-dimensional tensors, whereas we have modified it to fit the structure of a two-dimensional tensor.

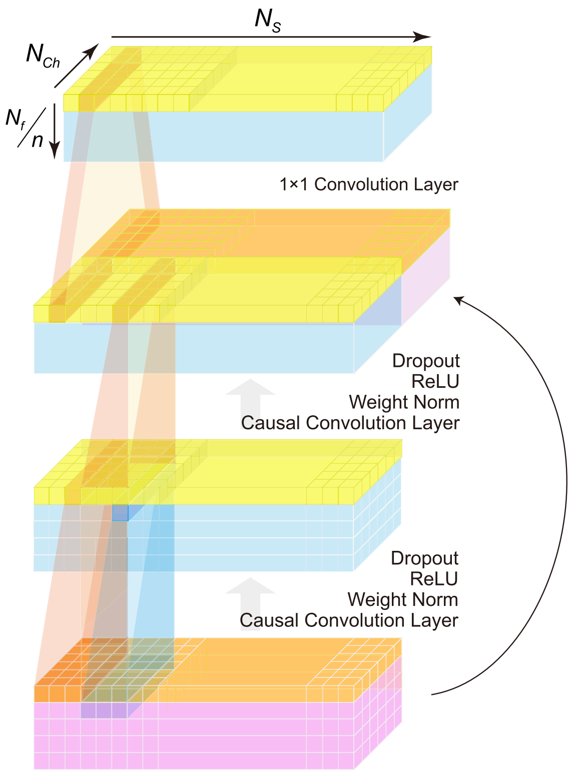


Fig. 3. Structure of one TCN block with dilation = 1. N_{Ch} is the number of channels in the TCN block. The shape of the convolution kernel is marked in the figure. The first two layers are the causal convolution layers. The output at one moment depends only on the input at the moment before it. The last convolution layer is a 1 × 1 convolution kernel for channel merging. We kept the input and output width at N_S by zero padding.

Secondly, we did not directly sum the inputs and outputs by the residuals connection. We connected the input and output on the channel dimension and then passed the output through an additional convolution layer with a 1 × 1 kernel.

We used a total of six TCN blocks in our TCN structure. Each block uses 24 channels (i.e., $N_{Ch} = 24$). We used the same dilation and kernel setting as [44]. For the six TCN blocks, the dilation factor $d = 1, 2, 4, 8, 16, 32$ and the kernel size is 3. Each layer uses ReLU as the activation function and uses a dropout rate of 0.2 for the first two causal convolution layers, the same as in [44].

D. Training Details

We used a total of 30 epochs for training. The first stage takes 20 training epochs, and the second stage takes 10 training epochs.

In the first stage, we trained all three branches. We performed backpropagation to train the *Extraction-Net* by minimizing the cosine embedding loss

$$\mathcal{L}_{XT} = 1 - \frac{E(Z_i)^T C(\overline{Z_i})}{\|E(Z_i)\| \, \|C(\overline{Z_i})\|} = 1 - \frac{X^T T}{\|X\| \, \|T\|}. \quad (5)$$

In this equation, both X and T removes the second dimension of length 1. Since $\overline{Z_i}$ has higher SNR than Z_i, this approach makes it easier for the *Extraction Net* to learn how to reduce noise on EEG data. Also, this prevents the *Electrodes-Combination-Net* from overfitting. In the training stage, $\hat{S} \in \mathbb{R}^{1\times 1\times N_S}$ in S represents the row tensor generated by the *Generation-Net* corresponding to X. The equation for calculating $\hat{S}$ is $\hat{S} = G(Y_i)$.

TABLE I

Mean Running Time per Training Epoch and the Mean Testing Time per Testing Sample on the Benchmark Dataset

Data segment length (s)		0.4	0.6	0.8	1.0
Time per training epoch (s)	Stage 1	32.23	20.08	15.38	12.30
	Stage 2	30.30	19.33	15.03	11.99
Time per testing sample (10^{-4} s)		7.39	9.71	11.15	12.32

We trained both the *Electrodes-Combination-Net* and the *Generation-Net* by minimizing loss function

$$\mathcal{L}_{ST} = 1 - \frac{G(Y_i)^T C(\overline{Z_i})}{\|G(Y_i)\| \, \|C(\overline{Z_i})\|} = 1 - \frac{\hat{S}^T T}{\|\hat{S}\| \, \|T\|}. \quad (6)$$

In this equation, $\hat{S}$ and T removes the second dimension of length 1. This process allows the *Generation-Net* to map *Sine Template* into latent space. Note that in this process, although the latent space corresponding to the unseen classes is also generated by the *Generation-Net*, it does not participate in the backpropagation learning process. Moreover, the latent space generation process of any class in S does not interfere with the generation of latent space of other classes. This is determined by the shape of the convolutional kernel in the *Generation-Net*.

Since the first stage separates the training process for the *Extraction-Net* and the *Generation-Net*, it does not maximize the correlation between $\hat{S}$ and X. In the second stage, we trained the *Generation-Net* only by minimizing the cosine embedding loss of $\hat{S}$ and X

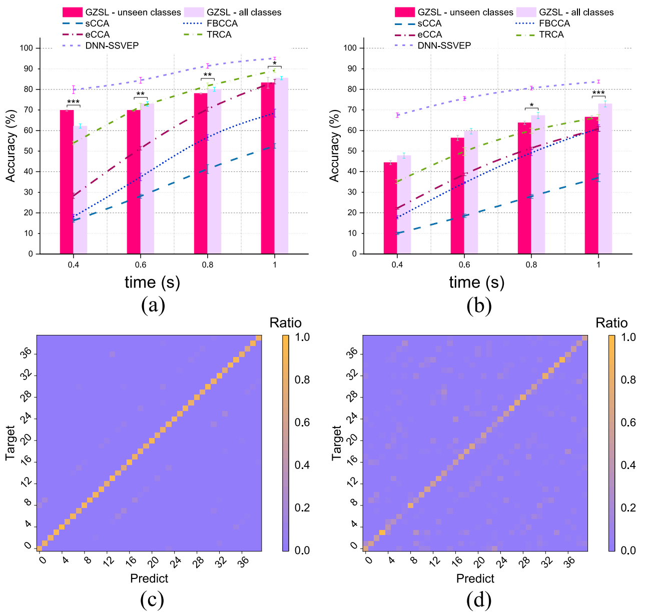
$$\mathcal{L}_{XS} = 1 - \frac{E(Z_i)^T G(Y_i)}{\|E(Z_i)\| \, \|G(Y_i)\|} = 1 - \frac{X^T \hat{S}}{\|X\| \, \|\hat{S}\|}. \quad (7)$$

We used the AdamW optimizer with the initial learning rate set to $1e^{-2}$. The learning rate decays to $1e^{-3}$ and $1e^{-4}$ after the $15th$ and $20th$ epochs. β_1 and β_2 are set to 0.9 and 0.999, $\epsilon = 1e^{-8}$, $\lambda = 1e^{-3}$. The batch size is set to 32.

IV. Result

We evaluated our proposed method on the Benchmark dataset [18] and the BETA dataset [19] using nine electrodes (Pz, PO5, PO3, POz, PO4, PO6, O1, Oz, and O2). The training process takes 30 epoches using an NVIDIA GPU (GeForce RTX 3080 with a memory of 10 GB). The time for training and testing is reported in Table I.

In the Benchmark dataset, we set eight classes (target 2, 7, 12, 17, 22, 27, 32, and 37) as unseen classes (Fig. 7 (a)). In the BETA dataset, we used the classes of the same stimulus pattern as unseen classes (target 1, 4, 7, 15, 18, 26, 29, and 32). Other classes are seen classes in the training



(a) (b)

(c) (d)

Fig. 4. Classification results. The asterisks are generated by paired t-test ($*: p < 0.05$, $** : p < 0.01$, $*** : p < 0.001$). (a) Average classification accuracy of the 35 subjects on the Benchmark dataset. (b) Average classification accuracy of the 70 subjects on the BETA dataset. The error bars indicate the standard errors. (c) The confusion matrix for subject 22 of the Benchmark data set at 0.6 s of stimulation. (d) The confusion matrix for subject 33 of the Benchmark data set at 0.6 s of stimulation.

stage. The network structure was optimized on the Benchmark dataset and maintains the same when testing on the BETA dataset. We observed the classification accuracy for unseen and all classes separately. Unlike the conventional ZSL method, we performed the 40-target classification when verifying the classification accuracy of the unseen classes. With this approach, we can observe the effect of unseen classes on the overall classification accuracy. Since the data is divided into 6 and 4 blocks for the Benchmark dataset and the BETA dataset, we used the leave-one-block-out cross-validation (LOOCV).

The performance of the SOTA training-based and training-free methods serves as the baselines. The Deep Neural Network for SSVEP (DNN-SSVEP) is the SOTA training-based method proposed recently [10]. It combines filterbank analysis with four convolution layers and one fully connected layer. We used the same dataset and the same electrode setting as [10] for performance validation. We chose two other training-based methods for comparison. Extended canonical correlation analysis (eCCA) [40] is closely related to our methods because we both use the sine-cosine wave as the template for correlation coefficient classification. The difference is that eCCA requires all the training data, while we did not. TRCA is another baseline method in training-based SSVEP [8]. Unlike eCCA, TRCA uses user data rather than sine-cosine waves to obtain templates.

FBCCA is one of the SOTA methods used in the training-free scheme [6]. For the FBCCA, we used data after 0.7 s instead of 0.5 s for classification. This allows the FBCCA to avoid interference from visual latency and achieve the highest accuracy. We also set the number of subbands to five. Other settings remain the same as given in the original study [6]. We also included the commonly used training-free method standard CCA (sCCA) [4] for comparison.

The classification accuracy is illustrated in Fig. 4 (a, b). The error bars indicate the standard errors of the classification accuracy for different test blocks. For each stimulus duration, we used paired t-test to analyze the performance differences between our method and other comparison methods. In the Benchmark dataset, the accuracy observed by our GZSL method is 62.19% ($T = 0.4$), 72.97% ($T = 0.6$), 80.05% ($T = 0.8$) and 85.49% ($T = 1.0$). The least significant difference between our GZSL method and other compared method is observed with (1) sCCA ($p = 2.84 \times 10^{-8}$) for $T = 0.8$, (2) FBCCA ($p = 3.17 \times 10^{-7}$) for $T = 0.8$, (3) eCCA ($p = 2.17 \times 10^{-3}$) for $T = 1.0$, (4) TRCA ($p = 3.38 \times 10^{-2}$) for $T = 0.8$, (5) DNN-SSVEP ($p = 1.45 \times 10^{-6}$) for $T = 0.4$. In the BETA dataset, the accuracy observed by our GZSL method is 47.78% ($T = 0.4$), 59.64% ($T = 0.6$), 67.31% ($T = 0.8$) and 72.91% ($T = 1.0$). The least significant difference and corresponding accuracy between our GZSL method and other compared method is observed with (1) sCCA ($p = 1.48 \times 10^{-5}$) for $T = 1.0$, (2) FBCCA ($p = 1.99 \times 10^{-4}$) for $T = 0.4$, (3) eCCA ($p = 2.85 \times 10^{-5}$) for $T = 0.6$, (4) TRCA ($p = 2.49 \times 10^{-4}$) for $T = 0.6$, (5) DNN-SSVEP ($p = 9.10 \times 10^{-5}$) for $T = 0.4$.

Although the training-based methods such as DNN-SSVEP and TRCA could outperform our method in accuracy, they are not available for the classification of unseen classes. It is not appropriate to use these methods when there are unseen classes in the test set. In the validation using the BETA dataset, we did not make any modifications to our model structure, keeping the same as the one used in the Benchmark dataset (unlike the DNN-SSVEP method). It is worth mentioning that the BETA dataset and its data are collected in a non-laboratory environment with lower SNR. It is more difficult to classify accurately on this dataset. We outperformed TRCA and eCCA on the BETA dataset greatly, which shows the high robustness of our method in the face of lower-quality data in a real-world application setting. It is also clear that our method significantly outperforms eCCA on both datasets, which demonstrates the advantage of our method.

On the Benchmark dataset, we selected two subjects with the highest (i.e., subject 22) and lowest (i.e., subject 33) classification accuracies for further evaluation. We trained the network with the last 5 blocks of data and validate the classification results on the remaining one block. The heat map of classification results for each target is illustrated in Fig. 4 (c, d). As can be seen from the figure, the distribution of errors is uniform and the classifier can classify each class equally well.

Since we used the correlation coefficient of the waveform for classification, it is important to generate a signal that is close to the real one in the latent space. Previous methods use the linear combination of sine waves to construct template signals, [4], [6], which assumes that the SSVEP signal contains only periodic waveforms and ignores other signal components. Such an operation is ideal for low complexity applications. Furthermore, the overall visual latency does not have to be

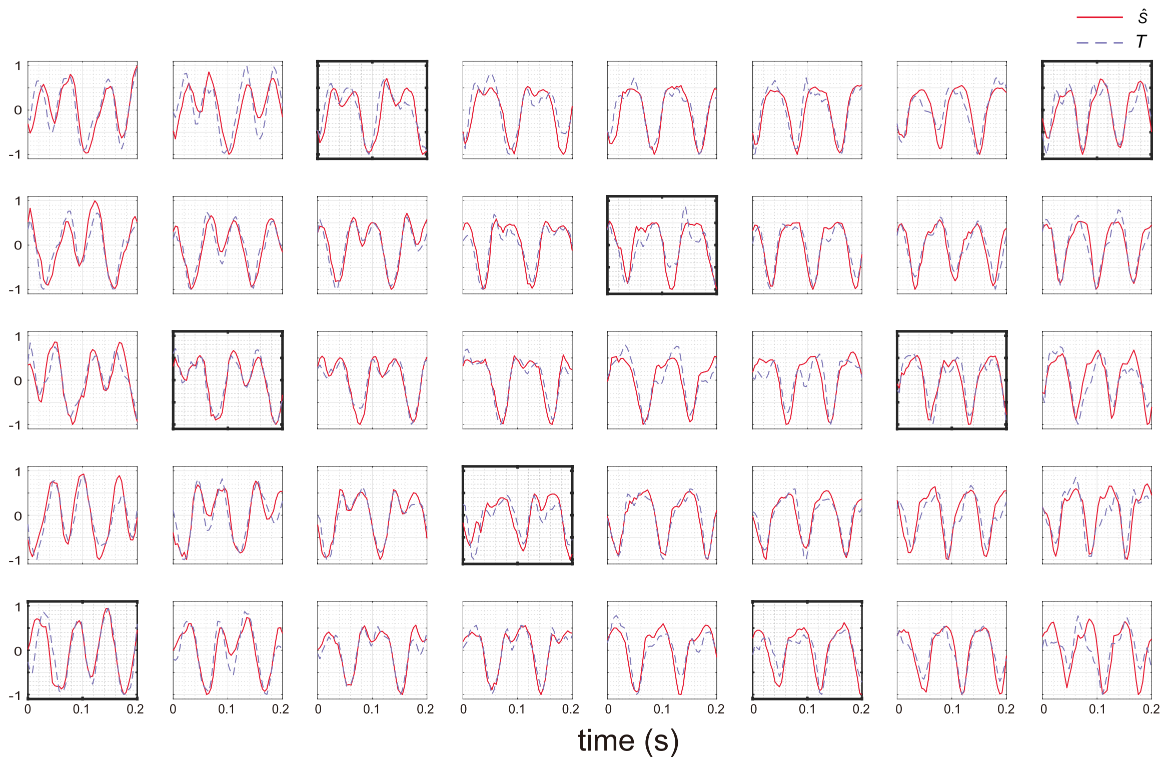


Fig. 5. Comparison of waveforms. Results for subject 22 in the Benchmark dataset. The arrangement of the subplots of the signal is consistent with the arrangement of the stimulus target. The dashed line is **T** and the solid line is $\hat{\mathbf{S}}$ generated by *Sine Template* corresponding to **T**. Plot boxes of unseen classes are indicated using thick lines.

considered due to the auxiliary angle formula

$$a \sin \alpha + b \cos \alpha = \sqrt{a^2 + b^2} \sin(\alpha + \arctan \frac{b}{a}). \tag{8}$$

The arbitrary phases can be expressed by different combination of parameter a and b. In the formula, a and b are the weights of the sine and cosine waves, and $\alpha = 2\pi hft$. Whereas our model used the deep neural network with a nonlinear projection. This model allows for the inclusion of other components that are not part of the stimulus. However, this approach requires that the visual latency is known. Therefore we used the TCN network as the backbone for learning visual latency. We compared the output of the *Generation-Net* and *Electrodes-Combination-Net* using data from subject 22 of the Benchmark dataset as an example. The results are plotted in Fig. 5. As can be seen from the figure, the generated signal $\hat{\mathbf{S}}$ contains detailed information about the signal. In addition, the waveforms of $\hat{\mathbf{S}}$ and $\mathbf{T}$ are well-matched in the time domain, indicating that the *Generation-Net* can learn from the visual latency.

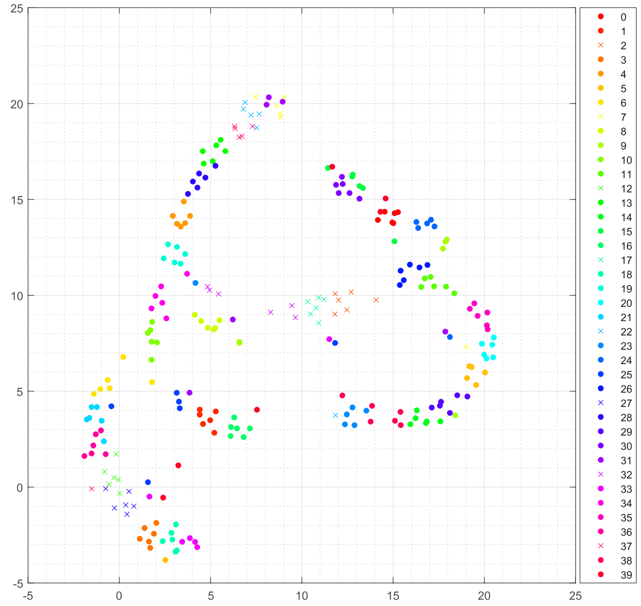


Fig. 6. Dimensionality reduction of the correlation coefficients. Different classes are represented by different colors, where the unseen class is plotted with × and the seen class is plotted with ·.

V. Discussion and Future Work

A. Classification Outputs

The purpose of this study was to show the feasibility of GZSL in SSVEP classification. Therefore, we did not specifically design the classifier and used the correlation coefficient. To verify the feasibility of subsequent improvements in the classifier, we plot the value of the correlation coefficient using t-Distributed Stochastic Neighbor Embedding (t-SNE) [45]. We used the data of subject 22 of the Benchmark data set for testing. The data from 2.3 s to 2.9 s in all data blocks were selected as input *Test EEG Data*. These data totaled 240 items. We recorded the 40-dimensional tensor of output correlation coefficients corresponding to each input. Then we use the t-SNE toolbox in MATLAB to perform dimensionality reduction to the 240 × 40-dimensional matrix. The result is plotted in Fig. 6.

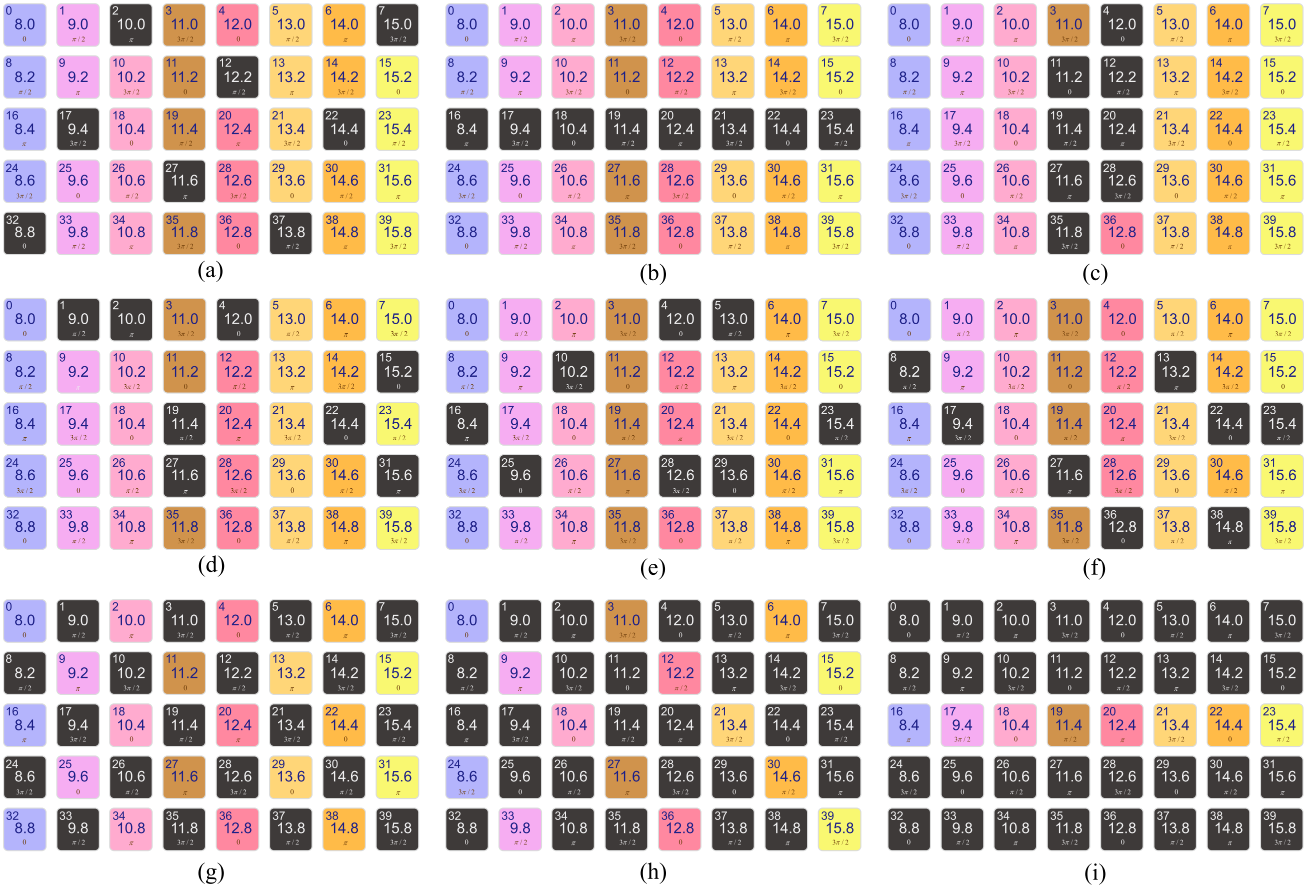


Fig. 7. Distribution of unseen classes. The background color of the unseen classes are marked with a dark color. (a-c) The different distributions when using the 8 unseen classes. In order, they are the *Even Distribution*, the *Grid Distribution*, and the *Block Distribution*. (d-f) Three different sets of random distribution of the unseen classes. (g-i) The distribution diagram with 20, 26, and 32 unseen classes respectively.

As seen in the figure, points that fall into the same category can be well clustered together. This result shows the feasibility of further utilizing different classifiers. Most algorithms use a fully connected (FC) layer as the final classification output, which is also applicable to our work [9], [10]. However, if the FC layer is used directly as the output layer, care needs to be taken to avoid common problems in GZSL. For instance, the network tends to discriminate unseen classes as seen classes. Several studies have been conducted to avoid such aspects [32], [46].

In addition, the Discriminator of generative adversarial networks (GAN) is also suitable for application in the classifier part of our network. There has been a lot of previous work applying GAN to GZSL [38], [47], [48]. The Discriminator can guide the training process of the *Generation-Net* in a better way. In our work, we used the cosine similarity loss to guide the training process of the *Generation-Net*. However, GAN uses a nonlinear mapping to guide the generation process, which was considered to be more effective in previous studies [35], [36]. In addition, since the Discriminator of GAN itself has the function of classification [36], [49], this can also facilitate the implementation of classification.

B. Distribution of Unseen Classes

The distribution of unseen classes is critical to the performance of our model, especially to the *Generation-Net*. Firstly, the different distributions determine the amount of data that each subnet of the *Generation-Net* can use for training. We divided the data for each subnet according to the neighboring frequencies. When the unseen classes are centrally distributed, the amount of training data for certain subnets could greatly reduced. In addition, if the knowledge learned by a subnet from the seen classes can not represent the unseen classes, this will also affect the performance. One situation is that it is hard to characterize other classes in the set with data far from the center.

We used the 0.6 s data segment of all 35 subjects from the Benchmark dataset for validation. Each time we used 8 unseen classes. The distribution is shown in Fig. 7 (a-c). The data distribution in the Fig. 7 (a) is the one we used to test the system performance in Section III of this paper. We performed the LOOCV and illustrate the result in Fig. 8.

As seen from the figure, the *Grid Distribution* has the best performance in the classification of unseen classes. A possible explanation is that for one subnet, the frequency of the unseen class is in the middle of all frequencies, which makes the data of the unseen class easy to characterize. This is also consistent with our assumptions. The visual circuits through which stimuli of neighboring frequencies pass are similar and can use the same parameters to express. On the contrary, the classification accuracy is significantly lower in the case of *Block Distribution*. The two subnets involving unseen classes

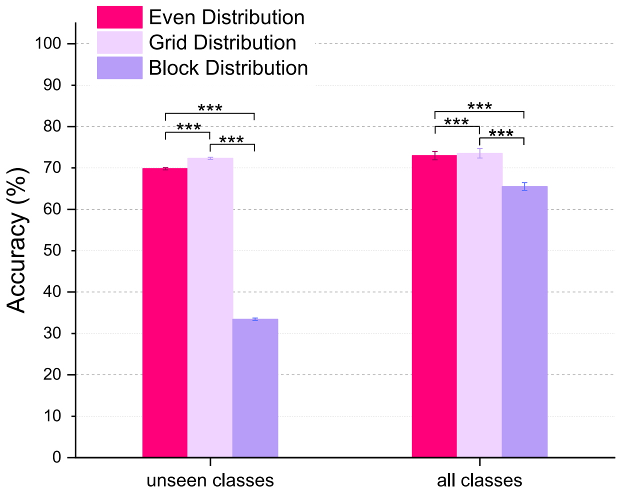


Fig. 8. Classification accuracy using different distributions. We used a data length of 0.6 s. Error bars indicate the standard errors of the standard errors. The asterisks are generated by paired t-test ($* : p < 0.05$, $** : p < 0.01$, $* * * : p < 0.001$).

obtained insufficient training data to fit. Moreover, the training data obtained by these two subnets deviate significantly from those of the unseen class.

To further test the performance, we also experimented when unseen classes were randomly distributed using 0.6 s segment. We performed a total of three sets of experiments, and the distribution of unseen classes are shown in Fig. 7 (d-f). The accuracies of these three sets were (d) 73.18 ± 1.10%, (e) 73.31 ± 1.16%, and (f) 73.17 ± 1.05% in all classes, respectively. For the unseen classes, their accuracy were (d) 70.38 ± 0.23%, (e) 70.36 ± 0.24%, and (f) 70.45 ± 0.25%. The one-way ANOVA analysis indicated no significant difference ($p = 0.29$ and $p = 0.24$). We compared the performance on the three random distributions by paired t-test. The accuracy of unseen classes and all classes are all significantly lower than the grid distribution ($p < 0.05$). They are also significantly higher than the block distribution ($p < 0.001$).

This result can guide the selection of unseen classes in the GZSL for SSVEP. First, the frequencies of the unseen class should be surrounded by the frequencies of the seen class. We cannot have all classes corresponding to a certain subnet as unseen classes. Otherwise, this would make the subnet impossible to train. Even if we only have a small number of seen classes, they should be enough to characterize the unseen classes. Second, the number of frequency groups is also important. Dividing more groups will lead to more accurate classification while limiting the amount of data obtained by each subnet. Whereas dividing fewer groups helps each subnet to get more training data. We made more discussion on the number of subnetworks in the following.

C. Different Number of Unseen Classes

We tested the performance of the proposed model in the case of more unseen classes. In designing the experiments, we considered the previous discoveries and used them to arrange the distribution of unseen classes. The selection of unseen classes is shown in Fig. 7 (g-i). We tested the classification accuracy of the system for all classes using 20, 26, and 32 unseen classes using 0.6 s data segment. The classification accuracy of all classes are presented in Table II. In the table, the highest accuracy for each portion is shown in bold font. We also recorded the standard error of the accuracy of different subjects.

TABLE II
USING DIFFERENT PORTION OF UNSEEN CLASSES. P-VALUE IS OBTAINED BY PAIRED T-TEST BETWEEN DIFFERENT NUMBER OF SUBNETS

	8-subnet		4-subnet		
unseen/seen classes	acc (%)	std.	acc (%)	std.	p-value
8/32	**72.97**	**1.12**	72.39	1.04	1.43×10^{-4}
20/20	**72.15**	**1.15**	71.12	0.97	9.01×10^{-5}
26/14	66.82	0.92	**67.78**	**0.86**	4.58×10^{-5}
32/8	55.72	0.63	**63.63**	**0.64**	6.48×10^{-9}
FBCCA	37.46	1.84	37.46	1.84	-

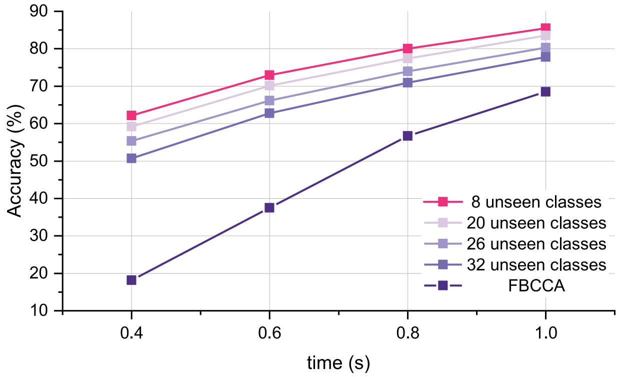


Fig. 9. Mean classification accuracy using different number of unseen classes. We used the dataset to illustrate the test results. We used 8-subnet for 8 and 20 unseen classes, and 4-subnet for 26 and 32 unseen classes.

When the number of unseen classes increases, the overall accuracy decreases. However, even with only 8 seen classes, our model still outperformed FBCCA. In addition, this result also shows the importance of the division of subnets. When using the *8-subnet* in the 32 unseen classes condition, the amount of data obtained by each subnet is small, thus significantly lowering the classification accuracy.

We then divided the stimulus frequencies into *4-subnet*, each group being 8.0 to 9.8 Hz, 10.0 to 11.8 Hz, 12.0 to 13.8 Hz, and 14.0 to 15.8 Hz. As can be seen from the table, *4-subnet* has less performance degradation then *8-subnet* when using less seen classes. However, when there are enough seen classes, using more subnets results in higher accuracy. This result leads to the trade-offs in our GZSL design. We used this setting to calculate the accuracy corresponding to each stimulus time in these cases and plotted in Fig. 9. Here, we used 8-subnet for 8 and 20 unseen classes, and 4-subnet for 26 and 32 unseen classes.

D. Different Data Lengths in Training and Testing

Since our model uses the correlation coefficients for classification, it does not require the training and test data to have the same length. In the training data pre-processing, a shorter data segment length can lead to more data segments [9]. Usually, more training data will result in better performance [50], [51], [52], [53]. Therefore, we tried to use shorter data segments for training and longer data segments for testing. The result

TABLE III

ACCURACY (%) USING DIFFERENT LENGTHS OF TRAINING DATA AND TEST DATA ON THE BENCHMARK DATASET

train/test Subject	0.6/0.8	0.8/0.8	0.6/1.0	1.0/1.0
S1	78.62	78.98	86.25	86.51
S2	84.24	84.88	88.33	89.52
S3	81.98	82.83	89.13	88.24
S4	88.02	88.17	91.54	91.03
S5	92.43	92.36	95.77	95.90
S6	88.50	88.33	92.37	92.44
S7	76.50	78.10	84.26	84.01
S8	56.93	57.81	65.51	66.19
S9	62.90	62.55	69.29	69.71
S10	82.10	83.33	88.72	88.46
S11	59.40	60.50	67.02	68.14
S12	88.40	88.07	93.33	92.98
S13	74.31	73.57	81.57	80.96
S14	89.55	90.12	94.33	93.85
S15	74.33	74.31	83.11	83.62
S16	78.21	78.93	85.10	84.26
S17	87.14	87.50	93.11	93.14
S18	59.45	59.52	66.67	66.38
S19	58.19	58.02	65.42	65.45
S20	77.60	77.86	85.99	85.10
S21	82.38	82.45	88.69	87.98
S22	93.21	93.57	96.22	96.19
S23	66.17	66.90	72.37	72.69
S24	89.45	90.10	93.14	92.66
S25	79.95	80.69	87.47	86.67
S26	88.38	88.38	92.69	93.37
S27	81.31	81.71	88.14	87.37
S28	80.95	81.17	85.67	87.34
S29	54.17	54.17	62.60	61.67
S30	83.05	83.19	88.53	88.40
S31	91.64	91.48	94.84	94.42
S32	85.67	85.52	92.12	91.76
S33	50.05	51.14	59.49	58.65
S34	81.33	81.90	88.08	87.69
S35	84.57	84.90	89.55	89.62
mean	78.03	78.37	84.18	84.07
std.	11.86	11.79	10.59	10.55
p-value	0.9058		0.9639	

is plotted in Table III. In Table III, 0.6/0.8 means the model was trained with the 0.6 s data segment and tested with the 0.8 s data segment. P-value is obtained by paired t-test.

As illustrated in the table, using different data segment lengths during training and testing has no significant impact on the results. Thus, the performance gains cannot be obtained by relying on reducing data length to increase training data volume. The reason for this situation we believe is caused by the use of overlapping cuts to acquire data. In this way, the total amount of information provided to the network does not change. However, there was a significant accuracy difference ($p = 1.26 \times 10^{-20}$) between the two groups of 0.6/0.8 and 0.6/1.0 which both used a time segment of 0.6 s for training. This is consistent with previous findings that higher accuracy can be achieved by longer stimuli [4], [6], [8], [9], [10], [40].

E. Future Work

This study developde a method of SSVEP-based BCI that only requires data from some classes for training. To enable further practical application of SSVEP-based BCI, this system can be further investigated in an integrated system. The existing methods requires the provider of the SSVEP system must be fully aware of the stimulation method of the future application before the training data is collected. This makes it challenge to design SSVEP systems that can integrate multiple applications. A disabled person who needs to use the SSVEP system may not only need to control a wheelchair [5], [54], but may also need to type [18] or even operate a robotic arm [55]. Without an integrated system, the user must provide training data for each system separately. This increases the training burden on the user. Our method can be further investigated in such kind of system and evaluated the perforance.

VI. CONCLUSION

We proposed a novel GZSL scheme for SSVEP classification and achieved a high classification accuracy for both unseen and seen classes in this work. Our method accomplished 89.9% of the accuracy of the SOTA training-based method. The three branches in our network served as signal extraction, latent space construction, and SSVEP signal generation. We tested the accuracy when using different distributions of unseen classes and gave suggestions for building the SSVEP classifier with the GZSL scheme in the future. We also applied this model to a less seen class situation. Even with only eight seen classes, we still exceed the SOTA training-free method in the 40-target classification. Thus, our work provided a new scheme for implementing a user-friendly multi-target SSVEP classifier in the future.

REFERENCES

[1] R. Abiri, S. Borhani, E. W. Sellers, Y. Jiang, and X. Zhao, "A comprehensive review of EEG-based brain-computer interface paradigms," *J. Neural Eng.*, vol. 16, Jan. 2019, Art. no. 011001.

[2] M. Li, D. He, C. Li, and S. Qi, "Brain–computer interface speller based on steady-state visual evoked potential: A review focusing on the stimulus paradigm and performance," *Brain Sci.*, vol. 11, no. 4, p. 450, Apr. 2021.

[3] A. Kawala-Sterniuk et al., "Summary of over fifty years with brain-computer interfaces—A review," *Brain Sci.*, vol. 11, no. 1, p. 43, Jan. 2021.

[4] Z. Lin, C. Zhang, W. Wu, and X. Gao, "Frequency recognition based on canonical correlation analysis for SSVEP-based BCIS," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 6, pp. 1172–1176, Jun. 2007.

[5] J. Xie, X. Wu, P. Fang, G. Li, G. Cao, and T. Xue, "The performance evaluation of SSVEP-BCI actuated wheelchair with parameter setting of time-window length and stimulation layout," in *Proc. IEEE Int. Workshop Metrology Ind. 4.0 (IoT)*, Jun. 2020, pp. 581–585.

[6] X. Chen, Y. Wang, S. Gao, T.-P. Jung, and X. Gao, "Filter bank canonical correlation analysis for implementing a high-speed SSVEP-based brain–computer interface," *J. Neural Eng.*, vol. 12, no. 4, Aug. 2015, Art. no. 046008.

[7] L. Xu, M. Xu, T.-P. Jung, and D. Ming, "Review of brain encoding and decoding mechanisms for EEG-based brain-computer interface," *Cognit. Neurodyn.*, vol. 15, pp. 1–16, Aug. 2021.

[8] M. Nakanishi, Y. Wang, X. Chen, Y. Wang, X. Gao, and T.-P. Jung, "Enhancing detection of SSVEPs for a high-speed brain speller using task-related component analysis," *IEEE Trans. Biomed. Eng.*, vol. 65, no. 1, pp. 104–112, Jan. 2018.

[9] Y. Li, J. Xiang, and T. Kesavadas, "Convolutional correlation analysis for enhancing the performance of SSVEP-based brain-computer interface," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 12, pp. 2681–2690, Dec. 2020.

[10] O. B. Guney, M. Oblokulov, and H. Ozkan, "A deep neural network for SSVEP-based brain-computer interfaces," *IEEE Trans. Biomed. Eng.*, vol. 69, no. 2, pp. 932–944, Feb. 2022.

[11] R. Zerafa, T. Camilleri, O. Falzon, and K. P. Camilleri, "To train or not to train? A survey on training of feature extraction methods for SSVEP-based BCIs," *J. Neural Eng.*, vol. 15, no. 5, Oct. 2018, Art. no. 051001.

[12] C. Han, G. Xu, J. Xie, M. Li, S. Zhang, and A. Luo, "An eighty-target high-speed Chinese BCI speller," in *Proc. 39th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2017, pp. 1652–1655.

[13] Y. Chen, C. Yang, X. Ye, X. Chen, Y. Wang, and X. Gao, "Implementing a calibration-free SSVEP-based BCI system with 160 targets," *J. Neural Eng.*, vol. 18, Jun. 2021, Art. no. 046094.

[14] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 951–958.

[15] Y. Fu, T. Xiang, Y.-G. Jiang, X. Xue, L. Sigal, and S. Gong, "Recent advances in zero-shot recognition: Toward data-efficient understanding of visual content," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 112–125, Jan. 2018.

[16] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proc. 6th Indian Conf. Comput. Vis., Graph. Image Process.*, Dec. 2008, pp. 722–729.

[17] M. Rohrbach, M. Stark, and B. Schiele, "Evaluating knowledge transfer and zero-shot learning in a large-scale setting," in *Proc. CVPR*, Jun. 2011, pp. 1641–1648.

[18] Y. Wang, X. Chen, X. Gao, and S. Gao, "A benchmark dataset for SSVEP-based brain-computer interfaces," *IEEE Trans. Neural Syst. Rehabil. Eng.* vol. 25, no. 10, pp. 1746–1752, Nov. 2016.

[19] B. Liu, X. Huang, Y. Wang, X. Chen, and X. Gao, "BETA: A large benchmark database toward SSVEP-BCI application," *Frontiers Neurosci.*, vol. 14, p. 627, Jun. 2020.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–9.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assisted Intervent*. Cham, Switzerland: Springer, 2015, pp. 234–241.

[22] A. Craik, Y. He, and J. L. Contreras-Vidal, "Deep learning for electroencephalogram (EEG) classification tasks: A review," *J. Neural Eng.*, vol. 16, no. 3, Jun. 2019, Art. no. 031001.

[23] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "EEGNet: A compact convolutional neural network for EEG-based brain–computer interfaces," *J. Neural Eng.*, vol. 15, no. 5, Oct. 2018, Art. no. 056013.

[24] R. Abiri, S. Borhani, E. W. Sellers, Y. Jiang, and X. Zhao, "A comprehensive review of EEG-based brain–computer interface paradigms," *J. Neural Eng.*, vol. 16, no. 1, Feb. 2019, Art. no. 011001.

[25] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2927–2936.

[26] W.-L. Chao, S. Changpinyo, B. Gong, and F. Sha, "An empirical study and analysis of generalized zero-shot learning for object recognition in the wild," in *Computer Vision ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 52–68.

[27] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1778–1785.

[28] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, vol. 2. Red Hook, NY, USA, Dec. 2013, pp. 3111–3119.

[29] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 453–465, Mar. 2013.

[30] Y. Zhu, M. Elhoseiny, B. Liu, X. Peng, and A. Elgammal, "A generative adversarial approach for zero-shot learning from noisy texts," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1004–1013.

[31] R. Gao et al., "Zero-VAE-GAN: Generating unseen features for generalized and transductive zero-shot learning," *IEEE Trans. Image Process.*, vol. 29, pp. 3665–3680, 2020.

[32] Y. Ye, Y. He, T. Pan, J. Li, and H. T. Shen, "Alleviating domain shift via discriminative learning for generalized zero-shot learning," *IEEE Trans. Multimedia*, vol. 24, pp. 1325–1337, 2022.

[33] F. Pourpanah et al., "A review of generalized zero-shot learning methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jul. 18, 2022, doi: 10.1109/TPAMI.2022.3191696.

[34] I. Goodfellow et al., "Generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 3, Jun. 2014, pp. 53–65.

[35] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, *arXiv:1411.1784*.

[36] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 2180–2188.

[37] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. 2nd Int. Conf. Learn. Represent., (ICLR)*, Banff, AB, Canada, Apr. 2014, pp. 1–14.

[38] S. Hwang, K. Hong, G. Son, and H. Byun, "EZSL-GAN: EEG-based zero-shot learning approach using a generative adversarial network," in *Proc. 7th Int. Winter Conf. Brain-Comput. Interface (BCI)*, Feb. 2019, pp. 1–4.

[39] L. Duan et al., "Zero-shot learning for EEG classification in motor imagery-based BCI system," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 11, pp. 2411–2419, Nov. 2020.

[40] X. Chen, Y. Wang, M. Nakanishi, X. Gao, T.-P. Jung, and S. Gao, "High-speed spelling with a noninvasive brain–computer interface," *Proc. Nat. Acad. Sci. USA*, vol. 112, no. 44, pp. E6058–E6067, 2015.

[41] B. Wittevrongel et al., "High-gamma oscillations precede visual steady-state responses: A human electrocorticography study," *Hum. Brain Mapping*, vol. 41, no. 18, pp. 5341–5355, Dec. 2020.

[42] P. Krolak-Salmon et al., "Human lateral geniculate nucleus and visual cortex respond to screen flicker," *Ann. Neurol.*, vol. 53, no. 1, pp. 73–80, Jan. 2003.

[43] T. Tsoneva, G. Garcia-Molina, and P. Desain, "SSVEP phase synchronies and propagation during repetitive visual stimulation at high frequencies," *Sci. Rep.*, vol. 11, no. 1, pp. 1–13, Mar. 2021.

[44] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.

[45] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[46] Z. Han, Z. Fu, S. Chen, and J. Yang, "Contrastive embedding for generalized zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2371–2381.

[47] S. Palazzo, C. Spampinato, I. Kavasidis, D. Giordano, and M. Shah, "Generative adversarial networks conditioned by brain signals," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3430–3438.

[48] J. Liu, L. Fu, H. Zhang, Q. Ye, W. Yang, and L. Liu, "Learning discriminative and representative feature with cascade GAN for generalized zero-shot learning," *Knowl.-Based Syst.*, vol. 236, Jan. 2022, Art. no. 107780.

[49] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.

[50] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *IEEE Intell. Syst.*, vol. 24, no. 2, pp. 8–12, Mar. 2009.

[51] R. Zerafa, T. Camilleri, O. Falzon, and K. P. Camilleri, "To train or not to train? A survey on training of feature extraction methods for SSVEP-based BCIs," *J. Neural Eng.*, vol. 15, no. 5, Oct. 2018, Art. no. 051001.

[52] R. L. Figueroa, Q. Zeng-Treitler, S. Kandula, and L. H. Ngo, "Predicting sample size required for classification performance," *BMC Med. Informat. Decis. Making*, vol. 12, no. 1, pp. 1–10, Dec. 2012.

[53] X. Chen, C. Li, A. Liu, M. J. McKeown, R. Qian, and Z. J. Wang, "Toward open-world electroencephalogram decoding via deep learning: A comprehensive survey," *IEEE Signal Process. Mag.*, vol. 39, no. 2, pp. 117–134, Mar. 2022.

[54] V. Sakkalis, M. Krana, C. Farmaki, C. Bourazanis, D. Gaitatzis, and M. Pediaditis, "Augmented reality driven steady-state visual evoked potentials for wheelchair navigation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 2960–2969, 2022.

[55] X. Chen, B. Zhao, Y. Wang, and X. Gao, "Combination of high-frequency SSVEP-based BCI and computer vision for controlling a robotic arm," *J. Neural Eng.*, vol. 16, no. 2, Apr. 2019, Art. no. 026012.