Semi-supervised Deep Learning in Motor Imagery-Based Brain-Computer Interfaces with Stacked Variational Autoencoder

Semi-supervised Deep Learning in Motor Imagery-Based Brain-Computer Interfaces with Stacked Variational Autoencoder

Junjian Chen, Zhuliang Yu and Zhenghui Gu

College of Automation Science and Engineering, South China University of Technology, Guangzhou, Guangdong, 510640, China
Email: 201720116570@mail.scut.edu.cn; zlyu@scut.edu.cn; zhgu@scut.edu.cn

Abstract. Recently, deep learning methods have contributed to the development of motor imagery (MI) based brain-computer interface (BCI) research. However, these methods typically focused on supervised deep learning with the labelled data and failed to learn from the unlabelled data, where additional information may be critical for performance improvement in MI decoding. To address this problem, we propose a semi-supervised deep learning method based on the stacked variational autoencoder (SVAE) for MI decoding, where the input to the network is an envelope representation of EEG signal. Under the framework of SVAE, the labelled training data and unlabelled test data can be trained collaboratively. Experimental evaluation on the BCI IV 2a dataset reveals that SVAE outperforms competing methods and it also yields state-of-the-art performance in decoding MI tasks. Hence, the proposed method is a promising tool in the research of the MI-based BCI system.

1. Introduction

Motor imagery (MI) has attracted increasing attention in the research of non-invasive brain-computer interfaces (BCIs) due to its advantage of no external stimulus required [1]. In the MI-based BCI system, the subject is asked to perform MI tasks on cue to trigger the event-related patterns in specific brain areas [2]. Since these patterns may vary dramatically among subjects, intensive training is required for each subject [3]. Besides, the recorded electroencephalography (EEG) signals have high dimensionality and low signal-to-noise ratio, which makes it challenging to decode the MI EEG correctly. Therefore, efficient feature extraction and correct classification are two major challenges in the MI-based BCI system.

Amongst the representative feature extraction methods, common spatial pattern (CSP) algorithm [4] and its extensions [5-7], were used frequently as spatial filters to capture the ERD/ERS patterns of MI EEG signals, and then the energies of spatially filtered channels were taken as features for the follow-up classification. In the classification part, many classical classifiers, such as Bayesian classifier and support vector machine (SVM), can be used to classify these energy features [6].

In recent years, deep learning methods have achieved huge success in many fields [8], yielding superior performance to the traditional hand-craft methods. Meantime, the development of deep learning has made a progressive influence on BCI research and provided the possibility of a performance breakthrough for MI decoding. An et al. [9] proposed a boosted deep belief networks (DBNs) structure for MI decoding, demonstrating the better performance than SVM when using the same frequency representation of EEG signals via fast Fourier transformation (FFT). Yang et al. [10]

proposed a deep model by combining the augmented CSP (ACSP) and convolutional neural network (CNN), which could learn the discriminative features without relying on hand-craft extraction. Sakhavi et al. [11] designed a CNN architecture for MI decoding, where the input to the network kept temporal representation. Zhang et al. [12] proposed a hybrid deep learning scheme by combining CNN and long short-term memory (LSTM) model to decode MI EEG.

However, the aforementioned deep learning methods [9-12] performed MI decoding in a supervised manner, i.e., only labelled trials were used for training, which ignored the useful information hidden in the available unlabelled trials. Additionally, collecting the labelled EEG recordings for fully supervised learning is time-consuming and expensive, but it is relatively easier to collect a large number of unlabelled EEG recordings. To overcome this problem, this paper focus on the semi-supervised deep learning for MI decoding by using the stacked variational autoencoder (SVAE) [13].

The rest of this paper is arranged as follows. Section 2 introduces the proposed method SVAE. Section 3 describes the details of dataset and experimental settings, and then gives the results and discussion. Section 4 summarizes the paper.

2. Methodology

2.1. Data Transformation

Before deep learning for MI decoding, it is necessary to transform the original EEG signals into an appropriate representation. By referring to Ref. [11], we introduced an envelope representation of the EEG signals for the subsequent MI decoding. The data transformation process mainly consists of two steps. Firstly, the spatially filtered EEG signals with temporal representation can be obtained by modifying the filter bank common spatial pattern (FBCSP) [14] method. Secondly, an envelope representation of these spatially filtered EEG signals is extracted via the Hilbert transform. Since the envelope representation is of low-frequency nature, it can be down-sampled to lower the dimensionality of EEG data.

2.2. Stacked Variational Autoencoder

In variational autoencoders, neural networks are used as probabilistic encoders or decoders to build the probability distributions to approximate the true distributions of the feature data. Based on the stacked variational autoencoder (SVAE) proposed in Ref. [13], it is feasible to implement semi-supervised deep learning for MI decoding. The probabilistic graph model of SVAE is shown in figure 1, which includes observed variable $\boldsymbol{x}$, latent variable $\boldsymbol{z}_1$, latent variable $\boldsymbol{z}_2$, and class variable y. The generative model of SVAE is as follows:

$$\begin{aligned} p_\theta(y) &= Cat(y \mid \boldsymbol{\pi}) \\ p_\theta(\boldsymbol{z}_2) &= N(\boldsymbol{z}_2 \mid \mathbf{0}, \mathbf{I}) \\ p_\theta(\boldsymbol{z}_1 \mid \boldsymbol{z}_2, y) &= N\left(\boldsymbol{z}_1 \mid \boldsymbol{\mu}_\theta(\boldsymbol{z}_2, y), diag\left(\boldsymbol{\sigma}_\theta^2(\boldsymbol{z}_2, y)\right)\right) \\ p_\theta(\boldsymbol{x} \mid \boldsymbol{z}_1) &= N\left(\boldsymbol{x} \mid \boldsymbol{\mu}_\theta(\boldsymbol{z}_1), diag\left(\boldsymbol{\sigma}_\theta^2(\boldsymbol{z}_1)\right)\right) \end{aligned} \tag{1}$$

where $N(\cdot)$ is the normal distribution, $Cat(\cdot)$ is the multinomial distribution, and $diag(\cdot)$ is the diagonal matrix. For the inference model, if class variable y is observed, and thereby the inference model of SVAE is as follows:

$$\begin{aligned} q_\phi(\boldsymbol{z}_1 \mid \boldsymbol{x}) &= N\left(\boldsymbol{z}_1 \mid \boldsymbol{\mu}_\phi(\boldsymbol{x}), diag\left(\boldsymbol{\sigma}_\phi^2(\boldsymbol{x})\right)\right) \\ q_\phi(\boldsymbol{z}_2 \mid \boldsymbol{z}_1, y) &= N\left(\boldsymbol{z}_2 \mid \boldsymbol{\mu}_\phi(\boldsymbol{z}_1, y), diag\left(\boldsymbol{\sigma}_\phi^2(\boldsymbol{z}_1, y)\right)\right) \end{aligned} \tag{2}$$

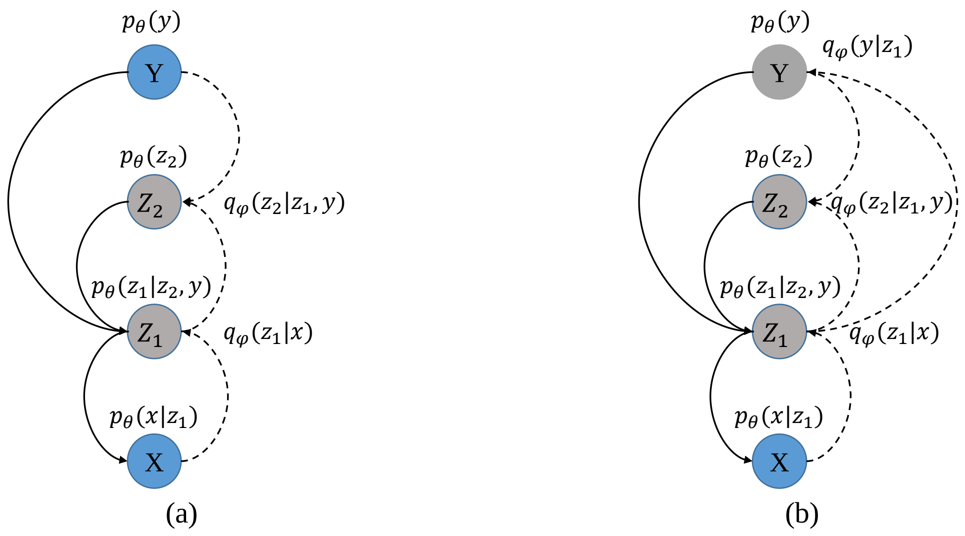

Figure 1. The probabilistic graph model of SVAE, where solid lines with arrows denote the generative model and dotted lines with arrows denote the inference model: (a) Class variable y is observed; (b) class variable y is missing.

The corresponding evidence lower bound (ELBO) on the margin likelihood of the labelled data is as follows:

$$\begin{aligned} L_{\theta,\phi}(\boldsymbol{x},y) &= E_{q_\phi(\boldsymbol{z}_1,\boldsymbol{z}_2|\boldsymbol{x},y)}\left[\log\frac{p_\theta(y)\cdot p_\theta(\boldsymbol{z}_2)\cdot p_\theta(\boldsymbol{z}_1\mid\boldsymbol{z}_2,y)\cdot p_\theta(\boldsymbol{x}\mid\boldsymbol{z}_1)}{q_\phi(\boldsymbol{z}_1\mid\boldsymbol{x})\cdot q_\phi(\boldsymbol{z}_2\mid\boldsymbol{z}_1,y)}\right] \\ &= E_{q_\phi(\boldsymbol{z}_1|\boldsymbol{x})}\left\{E_{q_\phi(\boldsymbol{z}_2|\boldsymbol{z}_1,y)}\left[\log\frac{p_\theta(y)\cdot p_\theta(\boldsymbol{z}_2)\cdot p_\theta(\boldsymbol{z}_1\mid\boldsymbol{z}_2,y)}{q_\phi(\boldsymbol{z}_2\mid\boldsymbol{z}_1,y)}\right]+\log\frac{p_\theta(\boldsymbol{x}\mid\boldsymbol{z}_1)}{q_\phi(\boldsymbol{z}_1\mid\boldsymbol{x})}\right\} \\ &= E_{q_\phi(\boldsymbol{z}_1|\boldsymbol{x})}\left[K_{\theta,\phi}(\boldsymbol{z}_1,y)+\log\frac{p_\theta(\boldsymbol{x}\mid\boldsymbol{z}_1)}{q_\phi(\boldsymbol{z}_1\mid\boldsymbol{x})}\right] \end{aligned} \tag{3}$$

For the inference model, if the class variable y is missing, and thereby the inference model of SVAE is as follows:

$$\begin{aligned} q_\phi(\boldsymbol{z}_1\mid\boldsymbol{x}) &= N\left(\boldsymbol{z}_1\mid\boldsymbol{\mu}_\phi(\boldsymbol{x}),diag\left(\boldsymbol{\sigma}_\phi^2(\boldsymbol{x})\right)\right) \\ q_\phi(y\mid\boldsymbol{z}_1) &= Cat\left(y\mid\boldsymbol{\pi}_\phi(\boldsymbol{z}_1)\right) \\ q_\phi(\boldsymbol{z}_2\mid\boldsymbol{z}_1,y) &= N\left(\boldsymbol{z}_2\mid\boldsymbol{\mu}_\phi(\boldsymbol{z}_1,y),diag\left(\boldsymbol{\sigma}_\phi^2(\boldsymbol{z}_1,y)\right)\right) \end{aligned} \tag{4}$$

The corresponding ELBO on the margin likelihood of the unlabelled data is as follows:

$$\begin{aligned} U_{\theta,\phi}(\boldsymbol{x}) &= E_{q_\phi(\boldsymbol{z}_1,\boldsymbol{z}_2,y|\boldsymbol{x})}\left[\log\frac{p_\theta(y)\cdot p_\theta(\boldsymbol{z}_2)\cdot p_\theta(\boldsymbol{z}_1\mid\boldsymbol{z}_2,y)\cdot p_\theta(\boldsymbol{x}\mid\boldsymbol{z}_1)}{q_\phi(\boldsymbol{z}_1\mid\boldsymbol{x})\cdot q_\phi(y\mid\boldsymbol{z}_1)\cdot q_\phi(\boldsymbol{z}_2\mid\boldsymbol{z}_1,y)}\right] \\ &= E_{q_\phi(\boldsymbol{z}_1|\boldsymbol{x})}\left\{\sum\nolimits_y\left[q_\phi(y\mid\boldsymbol{z}_1)\cdot\left(K_{\theta,\phi}(\boldsymbol{z}_1,y)-\log q_\phi(y\mid\boldsymbol{z}_1)\right)\right]+\log\frac{p_\theta(\boldsymbol{x}\mid\boldsymbol{z}_1)}{q_\phi(\boldsymbol{z}_1\mid\boldsymbol{x})}\right\} \end{aligned} \tag{5}$$

In summary, the optimization objective (ELBO) for the entire dataset is:

$$J=\sum\nolimits_{(\boldsymbol{x},y)\sim\tilde{p}_l}L_{\theta,\phi}(\boldsymbol{x},y)+\sum\nolimits_{\boldsymbol{x}\sim\tilde{p}_u}U_{\theta,\phi}(\boldsymbol{x}) \tag{6}$$

However, the label predictive distribution $q_{\phi}(y|\boldsymbol{z}_1)$ appears only to the second term related to the unlabelled data. To enable the distribution $q_{\phi}(y|\boldsymbol{z}_1)$ to learn from the labelled data, we add an auxiliary item for the labelled data, extending the optimization objective as follows:

$$\tilde{J} = J + \lambda \bullet \sum_{(x,y)\sim\tilde{p}_l} \left\{ E_{q_\phi(z_1|x)} \left[-\log q_\phi \left(y \mid z_1 \right) \right] \right\} \tag{7}$$

where λ is a hyper-parameter controlling the relative weight between generative learning and discriminative learning.

2.3. Devised Network Architecture

Based on the SVAE [13], we devised a network architecture for MI decoding. The detailed configuration of encoder structure for variables $\boldsymbol{z}_1$, $\boldsymbol{z}_2$and y are shown in tables 1-3 respectively, while the detailed configuration of decoder structure for variables $\boldsymbol{z}_1$ and $\boldsymbol{x}$ are shown in tables 4-5 respectively. For ease of practical application, the variable y is converted to the one-hot encoding representation. Note that the encoder structure for variable $\boldsymbol{z}_1$ and the decoder structure for variable $\boldsymbol{x}$ are not strictly symmetrical, where the decoder for variable $\boldsymbol{x}$ adopts the batch normalization (BN) [15] module, but the encoder for variable $\boldsymbol{z}_1$ does not. The reason is that the decoder of variable $\boldsymbol{x}$ contains multiple hidden layers, and the inputs to this decoder are the samples of latent variable $\boldsymbol{z}_1$, which can be generated on a large scale by Monte Carlo sampling method, so it is necessary to use the BN module to accelerate the convergence of the model.

Table 1. The configuration of encoder structure for variable $\boldsymbol{z}_1$.

Step	Module	Kernel Size/Stride	Output Size	Function
1	Input	—	1 @ 32×70	Input $\boldsymbol{x}$
2	Convolution	1×7 / 1×3	16 @ 32×22	—
3	ReLU	—	16 @ 32×22	—
4	Convolution	1×3 / 1×3	32 @ 32×8	—
5	ReLU	—	32 @ 32×8	—
6	Convolution	32×1 / 1×1	32 @ 1×8	Calculate the mean of $\boldsymbol{z}_1$
7	Flatten	—	256	Calculate the mean of $\boldsymbol{z}_1$
6	Convolution	32×1 / 1×1	32 @ 1×8	Calculate the std[a] of $\boldsymbol{z}_1$
7	Flatten	—	256	Calculate the std of $\boldsymbol{z}_1$
8	Softplus	—	256	Calculate the std of $\boldsymbol{z}_1$

[a] "std" is the abbreviation of standard deviation.

Table 2. The configuration of encoder structure for variable $\boldsymbol{z}_2$.

Step	Module	Output Size	Function
1	Input	256	Input $\boldsymbol{z}_1$
1	Input	4	Input y
2	Concat	260	Concatenate $\boldsymbol{z}_1$ and y
3	Fully connect	50	Calculate the mean of $\boldsymbol{z}_2$
3	Fully connect	50	Calculate the std of $\boldsymbol{z}_2$
4	Softplus	50	Calculate the std of $\boldsymbol{z}_2$

Table 3. The configuration of encoder structure for variable y.

Step	Module	Output Size	Function
1	Input	256	Input $\boldsymbol{z}_1$
2	Fully connect	4	Calculate y
3	Softplus	4	Calculate y

Table 4. The configuration of decoder structure for variable $\boldsymbol{z}_1$.

Step	Module	Output Size	Function
1	Input	50	Input $\boldsymbol{z}_2$
1	Input	4	Input y
2	Concat	54	Concatenate $\boldsymbol{z}_2$ and y
3	Fully connect	256	Calculate the mean of $\boldsymbol{z}_1$
3	Fully connect	256	Calculate the std of $\boldsymbol{z}_1$
4	Softplus	256	Calculate the std of $\boldsymbol{z}_1$

Table 5. The configuration of decoder structure for variable $\boldsymbol{x}$.

Step	Module	Kernel Size/Stride	Output Size	Function
1	Input	—	256	Input $\boldsymbol{z}_1$
2	Shape transformation	—	32 @ 1×8	—
3	Deconvolution	32×1 / 1×1	32 @ 32×8	—
4	Batch normalization	—	32 @ 32×8	—
5	ReLU	—	32 @ 32×8	—
6	Deconvolution	1×3 / 1×3	16 @ 32×22	—
7	Batch normalization		16 @ 32×22	—
8	ReLU	—	16 @ 32×22	—
9	Deconvolution	1×7 / 1×3	1 @ 32×70	Calculate the mean of $\boldsymbol{x}$
10	Softplus	—	1 @ 32×70	Calculate the mean of $\boldsymbol{x}$
9	Deconvolution	1×7 / 1×3	1 @ 32×70	Calculate the std of $\boldsymbol{x}$
10	Softplus	—	1 @ 32×70	Calculate the std of $\boldsymbol{x}$

3. Experiments and Results

3.1. Data Description

In this paper, the BCI IV 2a dataset [16] is adopted to verify the proposed method for MI decoding. The dataset includes four-class MI tasks (left hand, right hand, feet, and tongue) acquired from nine subjects. The MI EEG data were recorded with twenty-two EEG electrodes at a sampling rate of 250Hz. Two sessions of data were collected from each subject, where one was for training, and the other was for testing. Each session has 288 trials, and each trial has a 6-second length. In this paper, the time interval of single-trial EEG data between 2.5s and 6s is used. For each subject, 70% of training data (200 trials) were randomly chosen for training and the remainder (88 trials) were for validation.

3.2. Experiment Setup

During data transformation, the filter bank in the FBCSP algorithm comprised 12 band-pass filters with the frequency range of $[i, i+2]$ Hz and $[j, j+5]$ Hz, where $i = 6,8,10,12$ and $j = 14,17,20,23,26,29,32,35$. Besides, for two-class MI tasks, a total of 4 pairs of spatially filtered channels were selected within 12 frequency bands via MIBIF [14] algorithm, where two pairs of spatially filtered channels were provided in each frequency band for selection. Since BCI IV 2a dataset included four-class MI tasks, the one-versus-rest (OVR) strategy was appointed in the FBCSP algorithm, which led to a total of 32 spatially filtered channels. Then, the envelope representation of the spatially filtered EEG was extracted via the Hilbert transform. Because the envelope signal had a cutoff frequency of 5Hz, which meant a sampling frequency of 10Hz is sufficient for the signal according to Nyquist sampling theorem, we reduced the sampling rate from 250Hz to 20Hz and hence a total of 70 sampled points were yielded for each channel in the 3.5-second interval.

To enable the proposed method to perform MI decoding in a semi-supervised learning manner, we used 200 labelled training trials and 288 unlabelled test trials to train the SVAE for each subject. During the training process, the Adam optimizer [17] were used with the learning rate of 0.001, betas of (0.9, 0.999), weight penalty of 0.005, and the batch size of 64. Additionally, based on Monte Carlo posterior approximation, the number of particles used to form the ELBO estimator was set to 51.

3.3. Results and Discussion
To verify classification performance of the proposed method SVAE, the competing methods for comparison are as follows:

(1) CNN: the CNN was composed of the encoders of variable $\boldsymbol{z}_1$ and y in SVAE sequentially, where network layers used to calculate the standard deviation of variable $\boldsymbol{z}_1$ were removed.

(2) FBCSP-NBPW: the FBCSP algorithm followed by the naive Bayesian Parzen window (NBPW) classifier [14].

(3) CSP-NBPW: the CSP algorithm followed by the NBPW classifier [14].

Moreover, the proposed method was also compared with the top 3 winner methods [18] on BCI IV 2a dataset. Note that all the competing methods performed MI decoding in a supervised manner, *i.e.*, only the labelled trials were used for training. In this paper, the kappa coefficient [19] was chosen as the performance metric for BCI IV 2a dataset.

Table 6 summarizes the classification kappa derived by different methods on the test data from BCI IV 2a dataset. The SVAE achieves the highest mean kappa of 0.630, while the competing methods like CNN and FBCSP-NBPW achieve the mean kappa of 0.612 and 0.573 respectively. The confusion matrices for different methods are presented in figure 2, where the confusion matrix for SVAE is relatively larger on the main diagonal and smaller on the other sides. In addition, table 7 shows the paired T-test results among different methods, where all p values were adjusted by the false discovery rate correction for multiple comparisons.

Compared with the baseline methods FBCSP-NBPW and CSP-NBPW, the deep learning method CNN increases the classification kappa by 6.81% and 26.97%, respectively. The paired T-test results of CNN versus FBCSP-NBPW ($p \leq 0.05$), and CNN versus CSP-NBPW ($p \leq 0.005$) are also significant. Moreover, the CNN yields superior classification performance than the top 3 winner methods on BCI IV 2a dataset. These numerical results confirm that the deep learning method can significantly improve the classification performance of MI tasks, and hence it is a promising tool in MI decoding.

Table 6. Comparison of classification kappa on the test data from BCI IV 2a dataset.

Method	Mean Kappa	Subject								
		A01	A02	A03	A04	A05	A06	A07	A08	A09
SVAE	0.630	0.778	0.523	0.838	0.556	0.352	0.361	0.796	0.741	0.727
CNN	0.612	0.764	0.528	0.819	0.537	0.356	0.301	0.778	0.727	0.669
FBCSP-NBPW	0.573	0.755	0.449	0.801	0.505	0.273	0.296	0.792	0.671	0.616
CSP-NBPW	0.482	0.574	0.366	0.676	0.454	0.130	0.231	0.616	0.593	0.699
1st winner	0.570	0.680	0.420	0.750	0.480	0.400	o.270	0.770	0.750	0.610
2nd winner	0.520	0.690	0.340	0.710	0.440	0.160	0.210	0.660	0.730	0.690
3rd winner	0.310	0.380	0.180	0.480	0.330	0.070	0.140	0.290	0.490	0.440

Table 7. Paired T-test for the results of table 6.

	SVAE	CNN
CNN	*(0.024)	—
FBCSP-NBPW	†(1.3×10^{-3})	*(0.017)
CSP-NBPW	††(2.1×10^{-4})	†(1.1×10^{-3})
1st winner	*(0.017)	*(0.043)
2nd winner	†(1.1×10^{-3})	†(0.004)
3rd winner	††(7.3×10^{-5})	††(9.9×10^{-5})

Note: ~ nonsignificant, * $p \leq 0.05$, ** $p \leq 0.01$, † $p \leq 0.005$, †† $p \leq 0.001$

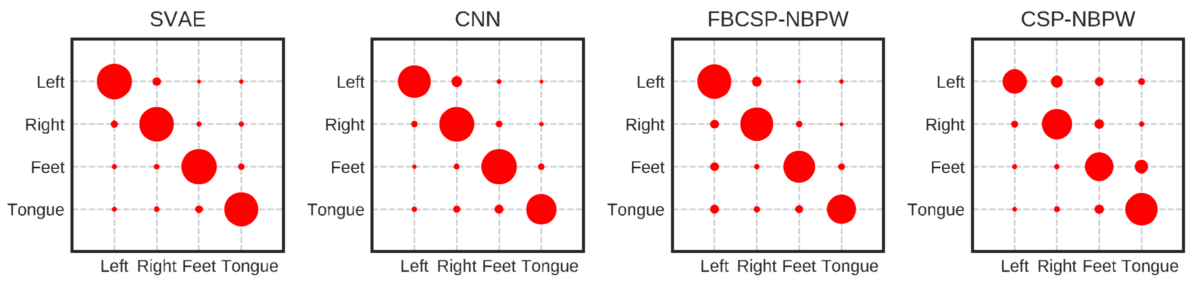

Figure 2. Confusion matrices for different methods on the BCI IV 2a dataset, where the size of the circle indicates the value of the corresponding element.

To study the effect of the information hidden in the unlabelled test data on the classification performance of MI tasks, we compared the proposed semi-supervised learning method SVAE and the supervised learning method CNN. As shown in tables 6-7, The SVAE increases 2.94% higher classification kappa than the CNN, and the paired T-test result of SVAE versus CNN ($p \leq 0.05$) is significant, reflecting the effectiveness of SVAE for MI decoding. A reasonable explanation is that during the semi-supervised learning process, the SVAE absorbs the additional useful information from the unlabelled test trials, which indeed contributes to the performance improvement of MI decoding. Therefore, the semi-supervised deep learning method SVAE is also promising in the application of MI-based BCI system.

4. Conclusion

In summary, the information of the unlabelled test trials is important to improve the classification performance in MI tasks. Under the devised framework based on SVAE, the labelled training trials and unlabelled test trials can be trained collaboratively, and hence the additional useful information from the unlabelled test trials is further exploited to improve the classification performance of the deep model. Experimental results on the BCI IV 2a dataset show the cutting-edge classification performance of the proposed method. As a promising tool for MI decoding, SVAE has a great potential for the research and application of MI-based BCI system.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China 61836003 and the Fundamental Research Funds for the Central Universities D2192260.

References

[1] Xie X, Yu Z L, Gu Z, Zhang J, Cen L and Li Y 2018 *IEEE Trans. Neural Syst. Rehabil. Eng.* **26** 698-708.
[2] Nam C S, Jeon Y, Kim Y J, Lee I and Park K 2011 *Clin. Neurophysiol.* **122** 567-577.

[3] Wolpaw J R, Birbaumer N, Heetderks W J, McFarland D J, Pekham P H, Schalk G, Dochin E, Quatrano L A, Robinson C J and Vaughan T M 2000 *IEEE Trans. Rehab. Eng.* **8** 164-173.
[4] Blankertz B, Tomioka R, Lemm S, Kawanabe M and Muller K R 2008 *IEEE Signal Process. Mag.* **25** 41-56.
[5] Novi Q, Guan C, Dat T H and Xue P 2007 *3rd Int. IEEE/EMBS Conf. Neural Eng.* ed Akay M (New York, USA: IEEE) pp 204-207.
[6] Ang K K, Chin Z Y, Zhang H and Guan C 2008 *Proc. Int. Jt. Conf. Neural Networks* (New York, USA: IEEE) pp 2390-97.
[7] Lotte F and Guan C 2011 *IEEE Trans. Biomed. Eng.* **58** 355-362.
[8] LeCun Y, Bengio Y and Hinton G 2015 *Nature* **521** 436-444.
[9] An X, Kuang D, Guo X, Zhao Y and He L 2014 *10th Int. Conf. Intelligent Computing* vol 8590 ed Huang D S, Han K *et al.* (Berlin, Germany: Springer Verlag) pp 203-210.
[10] Yang H, Sakhavi S, Ang K K and Guan C 2015 *37th Conf. Proc. IEEE Eng. Med. Biol. Soc.* ed Patton J (New York, USA: IEEE) pp 2620-23.
[11] Sakhavi S, Guan C and Yan S 2018 *IEEE Trans. Neural Netw. Learn. Syst.* **29** 5619-29.
[12] Zhang R, Zong Q, Dou L and Zhao X 2019 *J. Neural Eng.* **16** 066004.
[13] Kingma D P, Rezende D J, Mohamed S and Welling M 2014 *28th Adv. Neural Inf. Process. Syst.* vol 4 ed Ghahramani Z, Welling M et al. (La Jolla, California, USA: Neural Information Processing Systems Foundation) pp 3581-89.
[14] Ang K K, Chin Z Y, Wang C, Guan C and Zhang H 2012 *Front. Neurosci.* **6** 39.
[15] Ioffe S and Szegedy C 2015 *32nd Proc. Int. Conf. Machine Learning* ed Bach F R and Blei D M (Austin, Texas, USA: International Machine Learning Society) pp 448-456.
[16] Brunner C, Leeb R, Muller-Putz G R, Schlogl A and Pfurtscheller G *BCI Competition 2008-Graz Data Set A* http://www.bbci.de/competition/iv.
[17] Kingma D P and Ba J 2014 *arXiv:1412.6980.*
[18] Tangermann M, Mueller K R, Aertsen A, Birbaumer N, Braun C, Brunner C, Leeb R, Mehring C, Miller K J, Mueller-Putz G R, et al. 2012 *Front. Neurosci.* **6** 00055.
[19] Carletta J 1996 *Comput. Linguist.* **22** 249-254.