On the Vulnerability of CNN Classifiers in EEG-Based BCIs

Xiao Zhang and Dongrui Wu, *Senior Member, IEEE*

***Abstract*— Deep learning has been successfully used in numerous applications because of its outstanding performance and the ability to avoid manual feature engineering. One such application is electroencephalogram (EEG)-based brain-computer interface (BCI), where multiple convolutional neural network (CNN) models have been proposed for EEG classification. However, it has been found that deep learning models can be easily fooled with adversarial examples, which are normal examples with small deliberate perturbations. This paper proposes an unsupervised fast gradient sign method (UFGSM) to attack three popular CNN classifiers in BCIs, and demonstrates its effectiveness. We also verify the transferability of adversarial examples in BCIs, which means we can perform attacks even without knowing the architecture and parameters of the target models, or the datasets they were trained on. To the best of our knowledge, this is the first study on the vulnerability of CNN classifiers in EEG-based BCIs, and hopefully will trigger more attention on the security of BCI systems.**

***Index Terms*— Electroencephalogram, brain-computer interfaces, convolutional neural networks, adversarial examples.**

I. Introduction

A BRAIN-COMPUTER interface (BCI) is a communication pathway between the human brain and a computer [13]. Electroencephalogram (EEG), which measures the brain signal from the scalp, is the most widely used input signal in BCIs, due to its simplicity and low cost [25]. There are different paradigms in using EEG signals in BCIs, e.g., P300 evoked potentials [10], [33], [40], [43], motor imagery (MI) [29], steady-state visual evoked potential (SSVEP) [47], drowsiness/reaction time estimation [41], [42], [44], etc.

As shown in Fig. 1, a BCI system usually consists of four parts: *signal acquisition*, *signal preprocessing*, *machine learning*, and *control action*. The machine learning block usually includes feature extraction and classification/regression if traditional machine learning algorithms are applied. However, feature extraction and classification/regression can also be seamlessly integrated into a single deep learning model.

Manuscript received September 5, 2018; revised November 25, 2018 and February 24, 2019; accepted March 30, 2019. Date of publication April 2, 2019; date of current version May 9, 2019. This work was supported by the National Natural Science Foundation of China under Grant 61873321. *(Corresponding author: Dongrui Wu.)*

The authors are with the Key Laboratory of Image Processing and Intelligent Control, School of Artificial Intelligence and Automation, Ministry of Education, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: xiao_zhang@hust.edu.cn; drwu@hust.edu.cn).

Digital Object Identifier 10.1109/TNSRE.2019.2908955

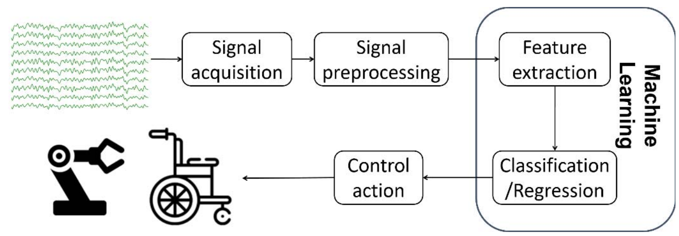


Fig. 1. The procedure of a BCI system when traditional machine learning algorithms are used. Manual feature extraction is not necessary if deep learning is employed.

Deep learning has achieved state-of-the-art performance in various applications, without the need of manual feature extraction. Recently, multiple deep learning models, particularly those based on convolutional neural networks (CNNs), have also been proposed for EEG classification in BCIs. Lawhern et al. (2016) [19] proposed EEGNet, which demonstrated outstanding performance in several BCI applications. Schirrmeister et al. (2017) [31] designed a deep CNN model (DeepCNN) and a shallow CNN model (ShallowCNN) to perform end-to-end EEG decoding. There were also some works converting EEG signals to images and then classifying them with deep learning models [4], [35], [37]. This paper focuses on the CNN models that accept the raw EEG signal as the input, more specifically, EEGNet, DeepCNN, and ShallowCNN. A CNN model using the spectrogram as the input is briefly discussed in Section III-J.

Albeit their outstanding performance, deep learning models are vulnerable to adversarial attacks. In such attacks, deliberately designed small perturbations, many of which may be hard to notice even by human, are added to normal examples to fool the deep learning model and cause dramatic performance degradation. This phenomena was first investigated by Szegedy et al. in 2013 [34] and soon received great attention. Goodfellow et al. (2014) [12] successfully confused a deep learning model to misclassify a panda into a gibbon. Kurakin et al. (2016) [18] found that deep learning systems might even make mistakes on printed photos of adversarial examples. Brown et al. (2017) [6] made an adversarial patch which was able to confuse deep learning models when attached on a picture. Athalye et al. (2017) [2] built an adversarial 3D-printed turtle which was classified as a riffle at every viewpoint with randomly sampled poses. Additionally, adversarial examples have also been used to attack speech recognition systems, e.g., a piece of speech, which is almost the same as a normal one but with a small adversarial perturbation, could be transcribed into any phrase the attacker chooses [8].

Adversarial examples could significantly damage deep learning models, which have been an indispensable component in computer vision, automatic driving, natural language processing, speech recognition, etc. To our knowledge, the vulnerability of deep learning models in EEG-based BCIs has not been investigated yet, but it is critical and urgent. For example, EEG-based BCIs could be used to control wheelchairs or exoskeleton for the disabled [20], where adversarial attacks could make the wheelchair or exoskeleton malfunction. The consequence could range from merely user confusion and frustration, to significantly reducing the user's quality of life, and even to hurting the user by driving him/her into danger on purpose. In clinical applications of BCIs in awareness evaluation/detection for disorder of consciousness patients [20], adversarial attacks could lead to misdiagnosis.

According to how much the attacker can get access to the target model, attacks can be categorized into three types:

1) *White-box attacks*, which assume that the attacker has access to all information of the target model, including its architecture and parameters. Most of the white-box attacks are based on some optimization strategies or gradient strategies, such as L-BFGS [34], DeepFool [24], the C&W method [7], the fast gradient sign method (FGSM) [12], the basic iterative method [18], etc.
2) *Black-box attacks*, which assume the attacker knows neither the architecture nor the parameters of the target model, but can observe its responses to inputs. Papernot et al. (2016) [27] developed a black-box attack approach which can be used to generate adversarial examples by interacting with the target model and training a substitute model. Su et al. (2017) [32] successfully fooled three different models by changing just one pixel of an image. Brendel et al. (2017) [5] proposed a black-box attack approach that starts from a large adversarial perturbation and then tries to reduce the perturbation while staying adversarial.
3) *Gray-box attacks*, which assume the attacker knows some but not all information about the target model, e.g., the training data that the target model is tuned on, but not its architecture and parameters.

In order to better compare the application scenarios of the three attack types, we summarize their characteristics in Table I. '−' means that whether the information is available or not will not affect the attack strategy, since it will not be used in the attack. It is clear that we need to know less and less information about the target model when going from white-box attack to gray-box attack and then to black-box attack. This makes the attack more and more practical, but we would also expect that knowing less information about the target model may affect the attack performance. This paper considers all three types of attacks in EEG-based BCIs.

TABLE I
SUMMARY OF THE THREE ATTACK TYPES

Target model information	White-Box	Gray-Box	Black-Box
Know its architecture	✓	×	×
Know its parameters $\boldsymbol{\theta}$	✓	×	×
Know its training data	−	✓	×
Can observe its response	−	−	✓

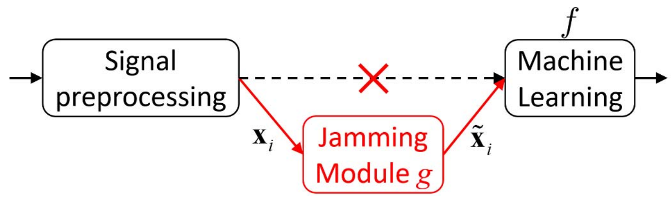


Fig. 2. Our proposed attack framework: inject a jamming module between signal preprocessing and machine learning to generate adversarial examples.

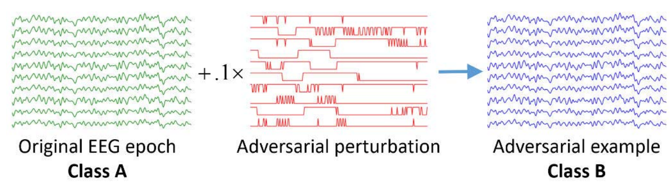


Fig. 3. A normal EEG epoch and its adversarial example. The two inputs could be classified into different classes, although they are almost identical.

According to its purpose, an attack can also be regarded as a *target attack*, which forces a model to classify an adversarial example into a specific class, or a *nontarget attack*, which only forces a model to misclassify the adversarial examples.

This paper aims at exploring the vulnerability of CNN classifiers under nontarget adversarial examples in EEG-based BCIs. We propose an attack framework which converts a normal EEG epoch into an adversarial example by simply injecting a jamming module before machine learning to perform adversarial perturbation, as shown in Fig. 2. We then propose an unsupervised fast gradient sign method (UFGSM), an unsupervised version of FGSM [12], to design the adversarial perturbation. The adversarial perturbation could be so weak that it is hardly noticeable, as shown in Fig. 3, but can effectively fool a CNN classifier. We consider three attack scenarios – white-box attack, gray-box attack, and black-box attack – separately. For each scenario, we provide the attack strategy to craft adversarial examples, and the corresponding experimental results. We show that our approaches can work in most cases and can significantly reduce the classification accuracy of the target model. To our knowledge, this is the first study on the vulnerability of CNN classifiers in EEG-based BCIs. It exposes an important security problem in BCI, and hopefully will lead to the design of safer BCIs.

The remainder of the paper is organized as follows: Section II proposes the strategies we use to attack the CNN classifiers. Section III presents the details of the experiments and the results on white-box attack, gray-box attack, and black-box attack. Section IV draws conclusion and points out some future research directions.

II. Strategies to Attack BCI Systems

This section introduces our strategies to attack EEG-based BCI systems when CNN is used as the classifier. We assume the attacker is able to invade a BCI system and inject a jamming module between signal preprocessing and machine learning, as shown in Fig. 2. This is possible in practice, as many BCI systems transmit preprocessed EEG signals to a computer, a smart phone, or the cloud, for feature extraction and classification/regression. The attacker could attach the jamming module to the EEG headset signal transmitter, or to the receiver at the other side, to perform adversarial perturbation.

The jamming module needs to satisfy two requirements: 1) the adversarial perturbation it generates should be so small that it is hardly detectable; and, 2) the adversarial example can effectively fool the CNN classifier. We propose UFGSM to construct it.

Let

$$\mathbf{x}_i = \begin{bmatrix} \mathbf{x}_i(1,1), & \cdots, & \mathbf{x}_i(1,T) \\ \vdots & \ddots & \vdots \\ \mathbf{x}_i(C,1), & \cdots, & \mathbf{x}_i(C,T) \end{bmatrix} \tag{1}$$

be the i-th raw EEG epoch, where C is the number of EEG channels and T the number of the time domain samples. Let y_i be the true class label associated with $\mathbf{x}_i$, $f(\mathbf{x}_i)$ the mapping from $\mathbf{x}_i$ to y_i used in the target CNN model, $\widetilde{\mathbf{x}}_i = g(\mathbf{x}_i)$ the adversarial perturbation generated by the jamming module g. Then, g needs to satisfy:

$$|\widetilde{\mathbf{x}}_i(c,t) - \mathbf{x}_i(c,t)| \leqslant \epsilon, \quad \forall c \in [1, C], \ t \in [1, T] \tag{2}$$

$$f(\widetilde{\mathbf{x}}_i) \neq y_i \tag{3}$$

Equation (2) ensures that the perturbation is no larger than a predefined threshold ϵ, and (3) requires the generated adversarial example should be misclassified by the target model f. Note that (2) must hold for every $c \in [1, C]$ and $t \in [1, T]$, but it may be impossible for every $\widetilde{\mathbf{x}}_i$ to satisfy (3), especially when ϵ is small. We evaluate the performance of the jamming module g by measuring the *accuracy* of the target model on the adversarial examples. A lower accuracy of the target model under adversarial attack indicates a better performance of the jamming model g, i.e., the more $\mathbf{x}_i$ satisfy (3), the better the performance of g is.

Next we describe how we construct the jamming module g. We first introduce FGSM, one of the most well-known adversarial example generators, and then our proposed UFGSM, which extends FGSM to unsupervised scenarios.

A. FGSM

FGSM was proposed by Goodfellow et al. (2014) [12] and soon became a benchmark attack approach. Let f be the target deep learning model, $\boldsymbol{\theta}$ be its parameters, and J be the loss function in training f. The main idea of FGSM is to find an optimal max-norm perturbation η constrained by α to maximize J. The perturbation can be calculated as:

$$\eta = \alpha \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y_i)) \tag{4}$$

And hence the jamming module g can be written as:

$$g(\mathbf{x}_i) = \mathbf{x}_i + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y_i)) \tag{5}$$

The requirement in (2) holds as long as $\alpha \leq \epsilon$.

Let $\alpha = \epsilon$ so that we can perturb $\mathbf{x}_i$ at the maximum extent. Then, g can be re-expressed as:

$$g(\mathbf{x}_i) = \mathbf{x}_i + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y_i)) \tag{6}$$

FGSM is an effective attack approach since it only requires calculating the gradients once instead of multiple times such as in the basic iterative method [18].

B. White-Box Attack

In a white-box attack, the attacker knows the architecture and parameters $\boldsymbol{\theta}$ of the target model f. It may represent the scenario that a BCI system designer wants to evaluate the worst case performance of the system under attack, since usually white-box attacks cause more damages to the classifier than gray-box or black-box attacks. The designer then uses all information he/she knows about the classifier to attack it. It may also represent scenarios that the target model of a BCI system is somehow leaked/hacked, or the target model is publicly available.

Knowing the architecture and parameters $\boldsymbol{\theta}$ of the target model f is not enough for FGSM, because it needs to know the true label y_i of $\mathbf{x}_i$ in order to generate the adversarial perturbation. Next we propose UFGSM, an unsupervised FGSM, to cope with this problem.

UFGSM replaces the label y_i by $y'_i = f(\mathbf{x}_i)$, i.e., the estimated label from the target model. Consequently, g in UFGSM can be rewritten as:

$$g(\mathbf{x}_i) = \mathbf{x}_i + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y'_i)) \tag{7}$$

y'_i approaches y_i when the accuracy of f is high, and hence the performance of UFGSM approaches FGSM. However, as will be demonstrated in the next section, our proposed UFGSM is still effective even when y'_i is quite different from y_i.

The pseudocode of our proposed UFGSM for white-box attacks is shown in Algorithm 1.

Algorithm 1 Our Proposed UFGSM for White-Box Attacks

Input: f, the target model; $\boldsymbol{\theta}$, the parameters of f; J, loss function of the target model; ϵ, the upper bound of perturbation; $\mathbf{x}_i$, a normal EEG epoch.
Output: $\widetilde{\mathbf{x}}_i$, an adversarial EEG epoch.

Calculate $y'_i = f(\mathbf{x}_i)$;
Calculate $\widetilde{\mathbf{x}}_i = \mathbf{x}_i + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y'_i))$.

return $\widetilde{\mathbf{x}}_i$

Algorithm 2 Our Proposed UFGSM for Gray-Box Attacks

Input: D, training data of the target model; J, loss function of the substitute model; ϵ, the upper bound of perturbation; $\mathbf{x}_i$, a normal EEG epoch.
Output: $\widetilde{\mathbf{x}}_i$, an adversarial EEG epoch.

Train a substitute model f' from D, using loss function J;
Calculate $y'_i = f'(\mathbf{x}_i)$;
Calculate $\widetilde{\mathbf{x}}_i = \mathbf{x}_i + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y'_i))$, where $\boldsymbol{\theta}$ encodes the parameters of f'.

return $\widetilde{\mathbf{x}}_i$

C. Gray-Box Attack

UFGSM does not need the true labels when generating adversarial examples, but it assumes that we know the parameters of the target model f, which is still challenging in practice. This requirement can be eliminated by utilizing the transferability of adversarial examples, which is the basis of both gray-box and black-box attacks.

The transferability of adversarial examples was first observed by Szegedy et al. (2013) [34], which may be the most dangerous property of adversarial examples. It denotes an intriguing phenomenon that adversarial examples generated by one deep learning model can also, with high probability, fool another model even the two models are different. This property has been used to attack deep learning models, e.g., Papernot et al. (2016) [27] attacked deep learning systems by training a substitute model with only queried information.

To better secure deep learning systems, a lot of studies have been done to understand the transferability of adversarial examples. Papernot et al. (2016) [28], Liu et al. (2016) [21] and Tramer et al. (2017) [39] all attributed this property to some kind of similarity between the models. However, Wu et al. (2018) [45] questioned these explanations since they found that the transferability of adversarial examples is not symmetric, which does not satisfy the definition of similarity.

Although more theoretical research is needed to understand both the adversarial example and its transferability property, this does not hinder us from using them in gray-box attack. Assume we have access to the training dataset used to construct the target model f, e.g., we know that f was trained using some public databases. The basic idea of gray-box attack is to train our own model f' to replace the target model f in UFGSM, so that we can get rid of the dependency on the target model f.

The pseudocode of UFGSM for gray-box attacks is shown in Algorithm 2.

D. Black-Box Attack

Gray-box attack assumes the attacker has access to the training data of the target model, e.g., the target model is trained on some classic public datasets. An even more challenging situation is black-box attack, in which the attacker has access to neither the parameters of the target model nor its training data. Instead, the attacker can only interact with the target model and observe its output for an input. One example is to attack a commercial proprietary EEG-based BCI system. The attacker can buy such a system and observe its responses, but does not have access to the parameters or the training data of the target model.

Algorithm 3 Our Proposed UFGSM for Black-Box Attacks

Input: f, the target model; J, loss function of the substitute model; λ, the parameter to control the step to generate the new training dataset; N, the maximum number of iterations; ϵ, the upper bound of perturbation; $\mathbf{x}_i$, a normal EEG epoch.
Output: $\widetilde{\mathbf{x}}_i$, an adversarial EEG epoch.

Construct a set of unlabeled EEG epochs S;
Pass S through f to generate a training dataset D;
Train a substitute model f' from D, using loss function J;
for $n = 1$ *to* N **do**
  $\Delta S = \{\mathbf{x} + \lambda \cdot \text{sign}(\nabla_{\mathbf{x}} J(\boldsymbol{\theta}, \mathbf{x}, y)) : (\mathbf{x}, y) \in D\}$, where $\boldsymbol{\theta}$ encodes the parameters of f';
  $\Delta D = \{(\mathbf{x}_i, f(\mathbf{x}_i))\}_{\mathbf{x}_i \in \Delta S}$;
  $D \leftarrow D \bigcup \Delta D$;
  Train a substitute model f' from D, using loss function J;
end

Calculate $y'_i = f'(\mathbf{x}_i)$;
Calculate $\widetilde{\mathbf{x}}_i = \mathbf{x}_i + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_i} J(\boldsymbol{\theta}, \mathbf{x}_i, y'_i))$, where $\boldsymbol{\theta}$ encodes the parameters of f'.

return $\widetilde{\mathbf{x}}_i$

Papernot et al. [27] proposed an approach to perform black-box attacks. A similar idea is used in this paper. We record the inputs and outputs of the target model to train a substitute model f', which is then used in UFGSM to generate adversarial examples, as shown in Algorithm 3. Note that the way we augment the training set is different from the original one in [27]. In [27], the new training set was constructed by calculating the Jacobian matrix corresponding to the labels assigned to the inputs, whereas we use the loss computed from the inputs instead.

III. Experiments and Results

This section validates the performances of the three attack strategies. Three EEG datasets and three CNN models were used.

A. The Three EEG Datasets

The following three EEG datasets were used in our experiments:

1) P300 Evoked Potentials (P300): The P300 dataset for binary-classification, collected from four disabled subjects and four healthy ones, was first introduced in [15]. In the experiment, a subjects faced a laptop on which six images were flashed randomly to elicit P300 responses. The goal was to classify whether the image is target or non-target. The EEG data were recorded from 32 channels at 2048Hz. We bandpass filtered the data to 1-40Hz and downsampled them to 256Hz. Then we extracted EEG epochs in [0,1]s after each image onset, normalized them using $\frac{x-\text{mean}(x)}{10}$, and truncated the resulting values into [-5, 5], as the input to the CNN classifiers. Each subject had about 3,300 epochs.

2) Feedback Error-Related Negativity (ERN): The ERN dataset [23] was used in a Kaggle competition[1] for two-class classification. It was collected from 26 subjects and partitioned into training set (16 subjects) and test set (10 subjects). We only used the 16 subjects in the training set as we do not have access to the test set. The 56-channel EEG data had been downsampled to 200Hz. We bandpass filtered them to 1-40Hz, extracted EEG epochs between [0,1.3]s, and standardized them using z-score normalization. Each subject had 340 epochs.

3) Motor Imagery (MI): The MI dataset is Dataset 2A[2] in BCI Competition IV [36]. It was collected from nine subjects and consisted of four classes (imagined movements of the left hand, right hand, feet, and tongue). The 22-channel EEG signals were sampled at 128Hz. As in [19], we bandpass filtered the data to 4-40Hz, and standardized them using an exponential moving average window with a decay factor of 0.999. Each subject had 576 epochs, 144 in each class.

B. The Three CNN Models

The following three different CNN models were used in our experiments:

1) EEGNet: EEGNet [19] is a compact CNN architecture with only about 1000 parameters (the number may change slightly according to the nature of the task). EEGNet contains an input block, two convolutional blocks and a classification block. To reduce the number of model parameters, it replaces the traditional convolution operation with a depthwise separable convolution, which is the most important block in Xception [9].

2) DeepCNN: Compared with EEGNet, DeepCNN [31] is deeper and hence has much more parameters. It consists of four convolutional blocks and a classification block. The first convolutional block is specially designed to handle EEG inputs and the other three are standard ones.

3) ShallowCNN: ShallowCNN [31] is a shallow version of DeepCNN, inspired by filter bank common spatial patterns [1]. Its first block is similar to the first convolutional block of DeepCNN, but with a larger kernel, a different activation function, and a different pooling approach.

C. Training Procedure and Performance Measures

The first two datasets have high class imbalance. To accommodate this, in training we applied a weight to each class, which was the inverse of its number of examples in the training set. We used the Adam optimizer [16], and cross

[1]https://www.kaggle.com/c/inria-bci-challenge

[2]http://www.bbci.de/competition/iv/

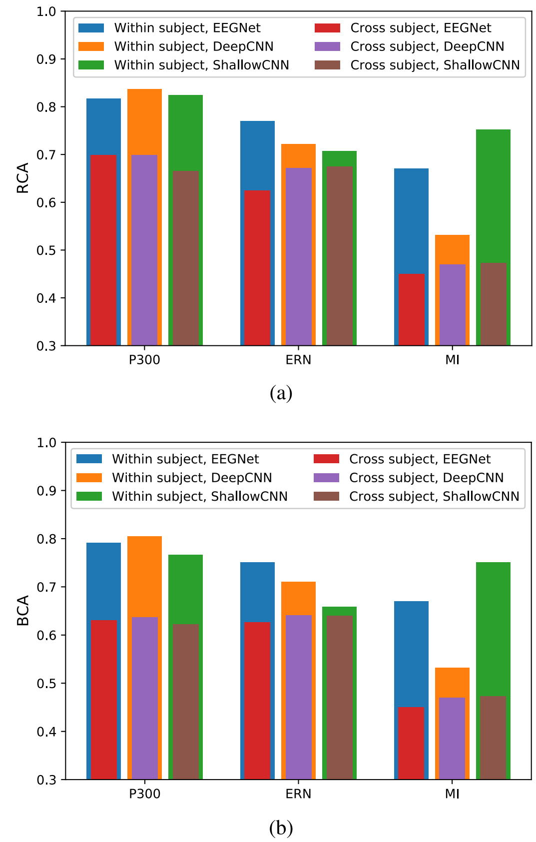


Fig. 4. Baseline classification accuracies of the three CNN classifiers on different datasets. (a) RCAs; (b) BCAs.

entropy as our loss function. Early stopping was used to reduce overfitting.

The test set still had class imbalance, which resembled the practical application scenario. We employed two metrics to evaluate the test performance:

1) *Raw classification accuracy* (RCA), which is the ratio of the total number of correctly classified test examples to the total number of test examples.
2) *Balanced classification accuracy* (BCA) [40], which is the average of the individual RCAs of different classes.

D. Baseline Performances on Clean EEG Data

We first evaluated the baseline performances of the three CNN models.

1) Within-Subject Experiments: For each individual subject, epochs were shuffled and divided into 80% training and 20% test. We further randomly sampled 25% epochs from the training set as our validation set in early stopping. We calculated the mean RCAs and BCAs from all subjects as the performance measures.

2) Cross-Subject Experiments: For each dataset, leave-one-subject-out cross-validation was performed, and the mean RCAs and BCAs were calculated. Epochs from all subjects in the training set were mixed, shuffled, and divided into 75% training and 25% validation.

The baseline results are shown in Fig. 4 and Table II, and the corresponding models were regarded as our target models.

TABLE II

RCAs/BCAs of Different CNN Classifiers in White-Box and Gray-Box Attacks on the Three Datasets ($\epsilon = 0.1/0.1/0.05$ for P300/ERN/MI)

Experimental Setup	Dataset	Target Model f	Baselines		White-Box Attack	Substitute Model f' in Gray-Box Attack		
			Clean Data	Noisy Data		EEGNet	DeepCNN	ShallowCNN
Within-Subject	P300	EEGNet	.8168/.7915	.8156/.7896	.1947/.2204	.2342/.2486	.2832/.3017	.6753/.6538
		DeepCNN	.8371/.8049	.8350/.8006	.2047/.2276	.3865/.3717	.3065/.3118	.7110/.6730
		ShallowCNN	.8243/.7668	.8264/.7623	.1868/.2420	.6764/.6178	.6388/.5889	.3238/.3335
	ERN	EEGNet	.7702/.7513	.7693/.7435	.2353/.2519	.4283/.4070	.5377/.5268	.6994/.6697
		DeepCNN	.7224/.7100	.7233/.7113	.3539/.3549	.5864/.5689	.5294/.4983	.6719/.6483
		ShallowCNN	.7077/.6589	.7068/.6577	.3162/.3450	.6811/.6279	.6535/.6062	.4770/.4855
	MI	EEGNet	.6705/.6698	.6552/.6551	.1561/.1570	.2337/.2312	.4684/.4759	.3554/.3593
		DeepCNN	.5316/.5316	.5250/.5248	.2586/.2609	.5067/.5067	.3736/.3803	.3554/.3589
		ShallowCNN	.7519/.7505	.7213/.7212	.1705/.1728	.6130/.6179	.5575/.5636	.2289/.2296
Cross-Subject	P300	EEGNet	.6985/.6306	.6978/.6295	.3085/.3786	.3635/.4092	.4666/.4729	.5351/.5308
		DeepCNN	.6992/.6366	.6982/.6345	.3095/.3666	.4096/.4500	.3739/.4115	.4711/.4882
		ShallowCNN	.6659/.6225	.6660/.6230	.3346/.3783	.4429/.4799	.4439/.4694	.3509/.4013
	ERN	EEGNet	.6250/.6266	.6272/.6309	.3904/.3823	.3561/.3917	.3263/.3533	.5241/.5455
		DeepCNN	.6719/.6404	.6792/.6434	.3281/.3595	.3450/.3897	.3254/.3464	.5379/.5186
		ShallowCNN	.6754/.6394	.6783/.6391	.3246/.3606	.5555/.5533	.5438/.5125	.3379/.3702
	MI	EEGNet	.4500/.4500	.4460/.4460	.2369/.2369	.2531/.2531	.2834/.2834	.2785/.2785
		DeepCNN	.4695/.4695	.4655/.4655	.2550/.2550	.3536/.3536	.2865/.2865	.2948/.2948
		ShallowCNN	.4734/.4734	.4660/.4660	.2610/.2610	.3520/.3520	.3009/.3009	.2658/.2658

Note that the RCAs and BCAs on the MI dataset were considerably lower, because MI was 4-class classification, whereas P300 and ERN were 2-class classification. For all datasets and all classifiers, RCAs and BCAs of within-class experiments were higher than their counterparts in cross-subject experiments, which is reasonable, because individual differences cause inconsistency among examples from different subjects.

E. Baseline Performances Under Random Noise

Before attacking these three target models using our proposed approaches, we corrupted the clean EEG data with random noise η_0 to check if that can significantly deteriorate the classification performances. If so, then we do not need to use a sophisticated approach to construct the adversarial examples: just adding random noise is enough.

The random noise was designed to be:

$$\eta_0 = \epsilon \cdot \mathrm{sign}\,(\mathcal{N}(0, 1)) \tag{8}$$

i.e., η_0 is either $-\epsilon$ or ϵ, so that its amplitude resembles that of the adversarial perturbations in (2). Although the EEG in all three datasets had similar standard deviations, empirically we found that the CNN classifiers trained on the MI dataset were more sensitive to noise than those on P300 or ERN. So, we set $\epsilon = 0.1$ for P300 and ERN, and $\epsilon = 0.05$ for MI in the experiments.

The results are shown in Table II. The classification accuracies on the noisy EEG data were comparable with, and sometimes even slightly better than, those on the clean EEG data, suggesting that adding random noise did not have a significant influence on the target models. In other words, adversarial perturbations cannot be implemented by simple random noise; instead, they must be deliberately designed.

F. White-Box Attack Performance

In a white-box attack, we know the target model exactly, including its architecture and parameters. Then, we can use

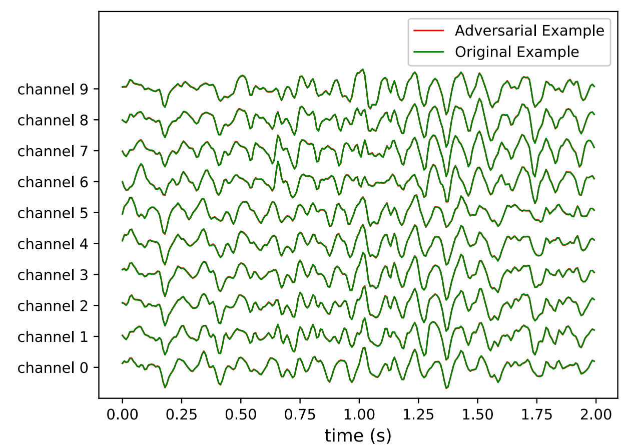


Fig. 5. An example of the EEG epoch before and after adversarial perturbation (MI dataset). $\epsilon = 0.05$.

UFGSM in Algorithm 1 to attack the target model. The results are shown in Table II. Clearly, there were significant performance deteriorations in all cases, and in most cases the classification accuracies after attack were even lower than random guess. Interestingly, though UFGSM is based on the assumption that the target model should have high accuracy so that we can replace the true class labels with the predicted ones, Table II shows that significant damages could also be made even when the target model has low accuracy, e.g., on the MI dataset.

An example of the EEG epoch before and after adversarial perturbation is shown in Fig. 5. The perturbation was so small that it is barely visible, and hence difficult to detect.

In summary, our results showed that the three CNN classifiers can all be easily fooled with tiny adversarial perturbations generated by the proposed UFGSM in Algorithm 1, when the attacker has full knowledge of the target model.

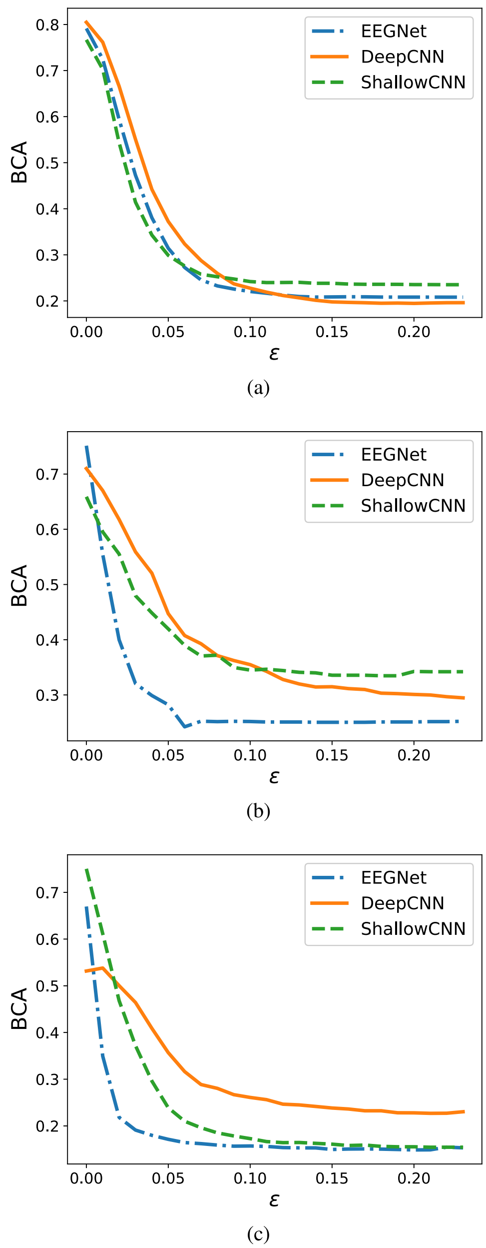


Fig. 6. BCAs of the target model after within-subject white-box attack, with respect to different ϵ. a) P300 dataset; b) ERN dataset; and, c) MI dataset.

$\epsilon = 0.1$ for P300/ERN and $\epsilon = 0.05$ for MI were used in the above experiments. Since ϵ is an important parameter in Algorithm 1, we also evaluated the performance of white-box attack with respect to different values of ϵ. The results are shown in Fig. 6. In all cases, the post-attack accuracy first decreased rapidly as ϵ increased, and then converged to a low value.

G. Gray-Box Attack Performance

Gray-box attack considers a more practical scenario than white-box attack: instead of knowing the architecture and parameters of the target model, here we only know its training data. In gray-box attack, we train a substitute model f' on the same training data, and use it in Algorithm 2 to generate adversarial examples. This subsection verifies the effectiveness of gray-box attack. Again, we set $\epsilon = 0.1/0.1/0.05$ for P300/ERN/MI, respectively.

Assume the target model is EEGNet, but the attacker does not know. The attacker randomly picks a model architecture, e.g., DeepCNN, and trains it using the known training data. This model then becomes the substitute model f' and is used in Algorithm 2 to generate adversarial examples. When different target models and different substitute models are used, the attacking performances are shown in the last part of Table II for the three datasets. We can observe that:

1) The RCAs and BCAs after gray-box attacks were lower than the corresponding baseline accuracies, suggesting the effectiveness of the proposed gray-box attack approach.
2) The RCAs and BCAs after gray-box attacks were generally higher than the corresponding accuracies after white-box attacks, especially when the attacker did not guess the architecture of the target model right, suggesting that knowing more target model information (white-box attack) can lead to more effective attacks.
3) In gray-box attacks, when the attacker guessed the right architecture of the target model, the attack performance was generally better than when he/she guessed the wrong architecture.

H. Black-Box Attack Performance

This subsection considers a hasher but most practical scenario: the attacker knows neither the parameters nor the training set of the target model.

To simulate such a scenario, we partitioned *8/16/9* subjects in the P300/ERN/MI dataset into two groups: *7/14/7* subjects in Group A, and the remaining *1/2/2* in Group B. We assume that the CNN classifier in the EEG-based BCI system was trained on Group A. The attacker, who belongs to Group B, bought such a system and would like to collect some data from himself/herself, train a substitute model f' using Algorithm 3, and then attack the CNN classifier. It's important to note that we used 80% epochs in Group A for training the target model f (among which 75% were used for tuning the parameters, and 25% for validation in early stopping), and the remaining 20% epochs in Group A for testing f and f'. Before training on the P300 and ERN datasets, to balance the classes of our initial dataset, we randomly downsampled the majority class according to the labels that the target model predicted at the first time.

$\lambda = 0.5$ and $N = 2$ (Algorithm 3) were used in our experiments. We only performed black-box attack on the mixed epochs from all the subjects in the training set, since it was too time-consuming to train cross-subject models or within-subject model for each subject. The baseline and black-box attack results are shown in Table III. Note that the baseline results were slightly different from those in Table II, because here we only used a subset of the subjects to train the baseline models, whereas previously we used all subjects. After black-box attack, the RCAs and BCAs of all target models decreased, suggesting that our proposed black-box

TABLE III

MIXED-SUBJECT RCAS/BCAS BEFORE AND AFTER BLACK-BOX ATTACK ON THE THREE DATASETS ($\epsilon = 0.1/0.1/0.05$ FOR P300/ERN/MI)

Dataset	Target Model f	Baselines		Substitute Model f'		
		Clean	Noisy	EEGNet	DeepCNN	ShallowCNN
P300	EEGNet	.7570/.7179	.7531/.7156	.3955/.4188	.5244/.5506	.5212/.5559
	DeepCNN	.7713/.7404	.7747/.7430	.3957/.4081	.3589/.4297	.4617/.4806
	ShallowCNN	.7336/.7163	.7375/.7186	.6315/.6189	.5113/.5505	.4118/.4301
ERN	EEGNet	.7665/.7614	.7687/.7640	.3113/.3288	.4893/.5024	.7207/.7006
	DeepCNN	.7719/.7455	.7740/.7519	.3134/.3644	.3049/.3963	.6972/.7113
	ShallowCNN	.7495/.7399	.7367/.7238	.4478/.3860	.3977/.3906	.4691/.5715
MI	EEGNet	.5603/.5565	.5345/.5324	.2352/.2343	.2586/.2571	.3005/.2988
	DeepCNN	.5222/.5189	.5135/.5108	.4446/.4446	.4483/.4475	.4384/.4353
	ShallowCNN	.6201/.6201	.6133/.6128	.5394/.5411	.5062/.5045	.4433/.4466

TABLE IV

SNRS (dB) OF NOISY EXAMPLES (NORMAL EXAMPLES PLUS RANDOM NOISE) AND ADVERSARIAL EXAMPLES. $\epsilon = 0.1/0.1/0.05$ FOR P300/ERN/MI

Dataset	Noisy examples	Adversarial examples		
		EEGNet	DeepCNN	ShallowCNN
P300	20.43	20.43	20.50	20.53
ERN	20.26	20.26	20.39	20.31
MI	25.50	25.50	25.57	25.60

attack strategy was effective. Generally, when the substitute model f' had the same structure as the target model f, e.g., both were EEGNet, the attack was most effective. This is intuitive.

I. Characteristics of the Perturbations

To better understand the characteristics of adversarial perturbations, this subsection studies the signal-to-noise ratio (SNR) of the adversarial examples, and the spectrogram of the perturbations.

We had no clue of the SNR of the normal epochs, so we had to assume that they contained very little noise. To compute the SNR of the noisy epochs [the random noise was generated using (8)], we treated the normal epochs as the clean signals, and the noise in (8) as the noise. To compute the SNR of the adversarial examples, we treated the normal epochs as the clean signals, and the adversarial perturbations as noise. The SNRs are shown in Table IV. In all three datasets, the SNRs of the noisy examples were roughly the same as those in the adversarial examples, which is intuitive, since they were controlled by the same parameter ϵ in our experiments. With the same amount of noise, the deliberately generated perturbations can significantly degrade the performances of the CNN classifiers, whereas random noise cannot, suggesting again the effectiveness of our proposed algorithms.

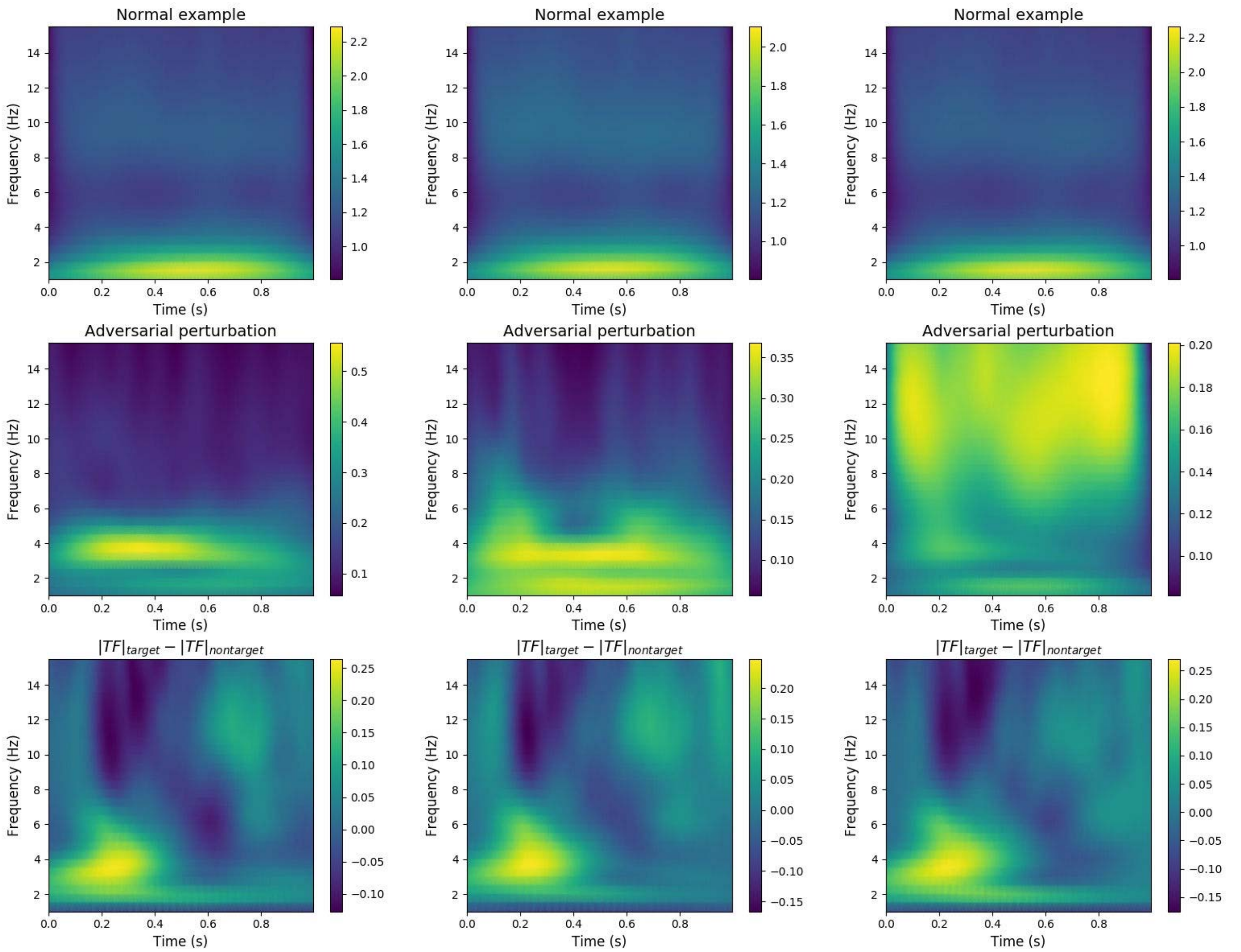
Next we analyze the spectrogram of the adversarial examples. Consider within-subject white-box attacks on the P300 dataset. For each classifier we partitioned the misclassified adversarial examples into two groups. Group 1 consisted of non-target examples whose adversarial examples were classified as targets, and Group 2 consisted of target examples whose adversarial examples were classified as non-targets. We then computed the spectrograms of all such examples using wavelet decomposition, the mean spectrogram of each group, and the difference of the two group means. The results are shown in the first and third row of Fig. 7 for the three classifiers. They were very similar to each other, in terms of their patterns and ranges. We could observe a clear peak around 0.2s for all three classifiers, maybe corresponding to the onset of P300 responses.

We then computed the mean spectrogram of the adversarial perturbations. The results are shown in the second row of Fig. 7 (note that their scales were much smaller than those in the first row). The patterns and ranges are now noticeably different. For EEGNet, the energy of the perturbations was concentrated in a small region, i.e., $[0, 0.8]$s and $[3, 5]$Hz, whereas that for DeepCNN was a little more scattered, and that for ShallowCNN was almost uniformly distributed in the entire time-frequency domain. These results suggest that the perturbations from the three CNN classifiers had dramatically different shapes, maybe specific to the characteristics of the classifiers. They also explain the within-subject gray-box attack results on the P300 dataset in Table II: the perturbations generated from EEGNet and DeepCNN were similar, and hence EEGNet (DeepCNN) as a substitute model could effectively attack DeepCNN (EEGNet). However, the perturbations generated from EEGNet/DeepCNN and ShallowCNN were dramatically different, and hence EEGNet/DeepCNN were less effective in attacking ShallowCNN, and vice versa.

J. Additional Attacks

To increase the robustness of a P300-based BCI system, a CNN classifier may be applied to the synchronized average of multiple epochs, instead of a single epoch. It's interesting to know if this averaging strategy can help defend adversarial attacks.

As mentioned in Section III-A, the P300 dataset was collected from eight subjects. Each subject completed four recording sessions, and each session consisted of six runs, one for each of the six images. The number of epochs of each run was about 120 as each image was flashed about 20 times. We constructed an averaged epoch as the average of 10 epochs from the same image. Thus, two averaged epochs were obtained from each image, and for each subject, $4 \times 6 \times 2 = 48$ averaged target (P300) epochs were obtained. Similarly, we obtained 248 averaged non-target epochs for



(a) (b) (c)

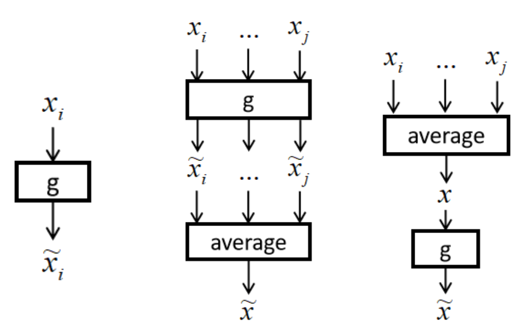
Fig. 7. Mean spectrogram of the normal examples (top row), mean spectrogram of the corresponding perturbations (middle row), and mean spectrogram difference between target and non-target normal examples (bottom row), in within-subject white-box attack on the P300 dataset. Note that the scales are different. The channel was randomly chosen. (a) EEGNet; (b) DeepCNN; and, (c) ShallowCNN.

each subject. These epochs were shuffled and divided into 80% training and 20% test for each subject.

We then compared the following three white-box attacks ($\epsilon = 0.1$) in within-subject experiments:

1) *Perturbation on each single epoch* (PSE), in which an adversarial example was generated for each single (unaveraged) epoch, as shown in Fig. 8(a). This was also the main attack considered before this subsection.
2) *Averaged adversarial examples* (AAE), in which an adversarial example was generated for each of the 10 single epochs, as in PSE, and then their average was taken, as shown in Fig. 8(b).
3) *Perturbation on the averaged epochs* (PAE), in which an adversarial example was generated directly on each averaged epoch, as shown in Fig. 8(c).

To better explore the transferability of adversarial examples, we also tested a traditional approach,[3] xDAWN + RG, which won the Kaggle BCI competition in 2015 and was also tested in [19]. xDAWN+RG used xDAWN spatial filtering [30], Riemannian geometry [3] and ElasticNet to classify the epochs.



(a) (b) (c)

Fig. 8. Three approaches to generate adversarial examples on the synchronized-averaging epochs. (a) PSE (no average); (b) AAE (attack, then average); and, (c) PAE (average, then attack).

[3]https://github.com/alexandrebarachant/bci-challenge-ner-2015

TABLE V
WITHIN-SUBJECT RCAS/BCAS BEFORE AND AFTER WHITE-BOX ATTACKS ON THE AVERAGED P300 EPOCHS. $\epsilon = 0.1$

models	Baseline				Adversarial examples		
	Single	Noisy	Average	Noisy	PSE	AAE	PAE
EEGNet	.7824/.7184	.7818/.7200	.9569/.9677	.9461/.9443	.1272/.2441	.2459/.3001	.0431/.0323
DeepCNN	.7725/.7375	.7712/.7383	.9332/.9265	.9224/.9202	.2371/.3494	.3169/.3329	.0819/.0826
ShallowCNN	.2477/.5291	.2417/.5273	.8685/.8393	.8556/.8357	.8491/.6147	.5465/.5421	.1315/.1607
xDAWN+RG	.8145/.5697	.8000/.5654	.9569/.8934	.9289/.7775	Attacked by EEGNet		.7931/.6417
					Attacked by DeepCNN		.7177/.5501
					Attacked by ShallowCNN		.8664/.6964

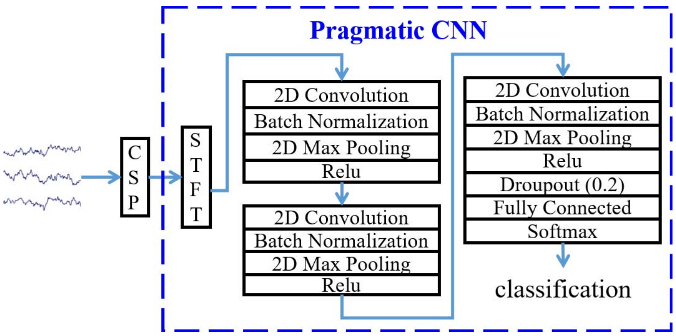


Fig. 9. The CNN classification pipeline on the MI dataset, when EEG spectrogram is used as the input feature.

TABLE VI
WITHIN-SUBJECT RCAS/BCAS BEFORE AND AFTER WHITE-BOX ATTACKS ON PRAGMATICCNN FOR THE MI DATASET. $\epsilon = 0.05$

Subject	Clean Examples	Noisy Examples	Adversarial Examples
s1	.8103/.8012	.8103/.7958	.1638/.1754
s2	.6552/.6422	.6638/.6508	.1810/.1898
s3	.8621/.8525	.8017/.8006	.1379/.1466
s4	.6810/.6781	.6724/.6722	.1983/.2023
s5	.5862/.5946	.5690/.5727	.2672/.2609
s6	.5259/.5272	.5517/.5532	.2241/.2250
s7	.8448/.8491	.8103/.8194	.1810/.1752
s8	.8448/.8328	.7759/.7568	.2414/.2535
s9	.8707/.8673	.8448/.8411	.2414/.2299
Average	.7423/.7383	.7222/.7181	.2040/.2065

We attacked it using PAE adversarial examples generated by different CNN models.

The results are shown in Table V.[4] All four approaches had improved performance when trained and tested on the averaged epochs, suggesting the rationale to take the synchronized average (of course, this also has the side effect of reducing the speed of the BCI system). For each CNN model, all three attack approaches were effective, but the attack was most effective when the adversarial examples were generated on the averaged epochs (i.e., PAE). The second part of Table V shows that adversarial examples generated by CNN models could be used to attack xDAWN + RG, but not as effective as in attacking the CNN models. However, this does not mean that traditional machine learning approaches are example from adversarial attacks [26]. They may require different attack strategies.

In all previous experiments we used raw EEG signals as the input to the CNN models. However, the spectrograms are also frequently used in MI based BCIs. Next, we study whether our attack strategies can still work for the spectrogram input. The CNN classification pipeline is shown in Fig. 9, where common spatial pattern (CSP) filtering [17] was used to reduce the number of EEG channels from 22 to 8, short-time Fourier transform (STFT) was used to convert EEG signals into spectrograms, and PragmatricCNN [37] was used as the classifier (EEGNet, DeepCNN and ShallowCNN cannot work on the spectrograms). Because both CSP and STFT are differentiable operations, we can compute the gradients over the whole pipeline to find the adversarial perturbations on the raw EEG time series.

The within-subject white-box attack results on the MI dataset are shown in Table VI. Clearly, the attack was very successful, suggesting that simply extracting the spectrogram as the input features cannot effectively defend adversarial attacks.

IV. CONCLUSION AND FUTURE RESEARCH

This paper investigates the vulnerability of CNN classifiers in EEG-based BCI systems. We generate adversarial examples by injecting a jamming module before a CNN classifier to fool it. Three scenarios – white-box attack, gray-box attack, and black-box attack – were considered, and separate attack strategies were proposed for each of them. Experiments on three EEG datasets and three CNN classifiers demonstrated the effectiveness of our proposed strategies, i.e., the vulnerability of CNN classifiers in EEG-based BCIs.

Our future research will:

1) Study the vulnerability of traditional machine learning approaches in BCIs. As shown in Section III-J, the adversarial examples generated from CNN models may not transfer well to traditional machine learning models, and hence new attack strategies are needed.
2) Investigate attack strategies to other components of the BCI machine learning model. Fig. 10 shows that attacks can target different components of a machine learning model. This paper only considered adversarial examples, targeted at the test input. Attacks to the training data,

[4] The baseline single epoch attack results were different from those in Table II, especially for ShallowCNN, because here the CNN classifiers were trained on the averaged epochs, and then applied to the single epochs, whereas in Table II the CNN classifiers were trained directly on the single epochs.

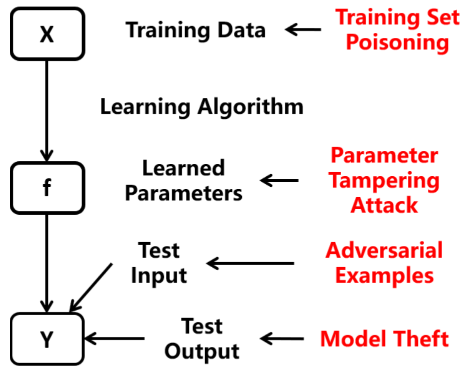


Fig. 10. Attack strategies to different components of a machine learning model.

learned model parameters, and the test output, will also be considered.

3) Study strategies to defend adversarial attacks on EEG-based BCIs. Multiple defense approaches, e.g., adversarial training [12], defensive distillation [28], ensemble adversarial training [38], and so on [14], [22], [46], have been proposed for other application domains. Unfortunately, there has not existed a universal defense approach because it is still unclear theoretically why adversarial examples occur in deep learning. Goodfellow et al. (2014) [12] believed that adversarial examples exist because of the linearity of deep learning models. Gilmer et al. (2018) [11] argued that adversarial examples occur as a result of the high dimensional geometry of the data manifold. We will investigate the root cause of adversarial examples in EEG classification/regression, and hence develop effective defense strategies for safer BCIs.

References

[1] K. K. Ang, Z. Y. Chin, H. Zhang, and C. Guan, "Filter bank common spatial pattern (FBCSP) in brain-computer interface," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Hong Kong, Jun. 2008, pp. 2390–2397.
[2] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proc. 35th Int. Conf. Mach. Learn.*, Stockholm, Sweden, Jul. 2018, pp. 284–293.
[3] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass brain-computer interface classification by riemannian geometry," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 4, pp. 920–928, Apr. 2012.
[4] P. Bashivan, I. Rish, M. Yeasin, and N. Codella, "Learning representations from EEG with deep recurrent-convolutional neural networks," in *Proc. Int. Conf. Learn. Represent.*, San Juan, Puerto Rico, May 2016.
[5] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, May 2018.
[6] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. (2017). "Adversarial patch." [Online]. Available: http://arxiv.org/abs/1712.09665
[7] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*. San Jose, CA, USA, May 2017, pp. 39–57.
[8] N. Carlini and D. A. Wagner. (2018). "Audio adversarial examples: Targeted attacks on speech-to-text." [Online]. Available: https://arxiv.org/abs/1801.01944
[9] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 1800–1807.
[10] L. A. Farwell and E. Donchin, "Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalogr. Clin. NeuroPhysiol.*, vol. 70, no. 6, pp. 510–523, 1988.
[11] J. Gilmer *et al.* (2018). "Adversarial spheres." [Online]. Available: https://arxiv.org/abs/1801.02774
[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, San Diego, CA, USA, May 2015.
[13] B. Graimann, B. Allison, and G. Pfurtscheller, *Brain-Computer Interfaces: A Gentle Introduction*. Berlin, Germany: Springer, 2009, pp. 1–27.
[14] C. Guo, M. Rana, M. Cisse, and L. van der Maaten. (2017). "Countering adversarial images using input transformations." [Online]. Available: https://arxiv.org/abs/1711.00117
[15] U. Hoffmann, J.-M. Vesin, T. Ebrahimi, and K. Diserens, "An efficient P300-based brain-computer interface for disabled subjects," *J. Neurosci. Methods*, vol. 167, no. 1, pp. 115–125, 2008.
[16] D. P. Kingma and L. J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, Apr. 2014.
[17] Z. J. Koles, M. S. Lazar, and S. Z. Zhou, "Spatial patterns underlying population differences in the background EEG," *Brain Topography*, vol. 2, no. 4, pp. 275–284, Jun. 1990.
[18] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proc. Int. Conf. Learn. Represent.*, Toulon, France, Apr. 2017.
[19] 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, 2018, Art. no. 056013.
[20] Y. Li *et al.*, "Multimodal BCIs: Target detection, multidimensional control and awareness evaluation in patients with disorder of consciousness," *Proc. IEEE*, vol. 104, no. 2, pp. 332–352, Feb. 2016.
[21] Y. Liu, X. Chen, C. Liu, and D. Song. (2016). "Delving into transferable adversarial examples and black-box attacks." [Online]. Available: https://arxiv.org/abs/1611.02770
[22] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. (2017). "Towards deep learning models resistant to adversarial attacks." [Online]. Available: https://arxiv.org/abs/1706.06083
[23] P. Margaux, M. Emmanuel, D. Sébastien, B. Olivier, and M. Jérémie, "Objective and subjective evaluation of online error correction during P300-based spelling," *Adv. Hum.-Comput. Interact.*, vol. 2012, Jan. 2012, Art. no. 4.
[24] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 2574–2582.
[25] L. F. Nicolas-Alonso and J. Gomez-Gil, "Brain computer interfaces, a review," *Sensors*, vol. 12, no. 2, pp. 1211–1279, 2012.
[26] N. Papernot, P. McDaniel, and I. Goodfellow. (2016). "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples." [Online]. Available: https://arxiv.org/abs/1605.07277
[27] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. Asia Conf. Comput. Commun. Secur.*, Abu Dhabi, UAE, Apr. 2017, pp. 506–519.
[28] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Jose, CA, USA, May 2016, pp. 582–597.
[29] G. Pfurtscheller and C. Neuper, "Motor imagery and direct brain-computer communication," *Proc. IEEE*, vol. 89, no. 7, pp. 1123–1134, Jul. 2001.
[30] B. Rivet, A. Souloumiac, V. Attina, and G. Gibert, "xDAWN algorithm to enhance evoked potentials: Application to brain–computer interface," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 8, pp. 2035–2043, Aug. 2009.
[31] R. T. Schirrmeister *et al.*, "Deep learning with convolutional neural networks for EEG decoding and visualization," *Hum. Brain Mapping*, vol. 38, no. 11, pp. 5391–5420, 2017.
[32] J. Su, D. V. Vargas, and S. Kouichi. (2017). "One pixel attack for fooling deep neural networks." [Online]. Available: https://arxiv.org/abs/1710.08864
[33] S. Sutton, M. Braren, J. Zubin, and E. R. John, "Evoked-potential correlates of stimulus uncertainty," *Science*, vol. 150, no. 3700, pp. 1187–1188, 1965.
[34] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent.*, Banff, AB, Canada, Apr. 2014.

[35] Y. R. Tabar and U. Halici, "A novel deep learning approach for classification of EEG motor imagery signals," *J. Neural Eng.*, vol. 14, no. 1, 2017, Art. no. 016003.
[36] M. Tangermann *et al.*, "Review of the BCI competition IV," *Frontiers Neurosci.*, vol. 6, p. 55, Jul. 2012.
[37] Z. Tayeb *et al.*, "Validating deep neural networks for online decoding of motor imagery movements from EEG signals," *Sensors*, vol. 19, no. 1, p. 210, Jan. 2019.
[38] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, May 2018.
[39] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. (2017). "The space of transferable adversarial examples." [Online]. Available: https://arxiv.org/abs/1704.03453
[40] D. Wu, "Online and offline domain adaptation for reducing BCI calibration effort," *IEEE Trans. Human-Mach. Syst.*, vol. 47, no. 4, pp. 550–563, Aug. 2017.
[41] D. Wu, J.-T. King, C.-H. Chuang, C.-T. Lin, and T.-P. Jung, "Spatial filtering for EEG-based regression problems in brain-computer interface (BCI)," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 2, pp. 771–781, Apr. 2018.
[42] D. Wu, V. J. Lawhern, S. Gordon, B. J. Lance, and C.-T. Lin, "Driver drowsiness estimation from EEG signals using online weighted adaptation regularization for regression (OwARR)," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1522–1535, Dec. 2017.
[43] D. Wu, V. J. Lawhern, W. D. Hairston, and B. J. Lance, "Switching EEG headsets made easy: Reducing offline calibration effort using active weighted adaptation regularization," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 11, pp. 1125–1137, Nov. 2016.
[44] D. Wu, V. J. Lawhern, B. J. Lance, S. Gordon, T.-P. Jung, and C.-T. Lin, "EEG-based user reaction time estimation using riemannian geometry features," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 11, pp. 2157–2168, Nov. 2017.
[45] L. Wu, Z. Zhu, C. Tai, and E. Weinan. (2018). "Understanding and enhancing the transferability of adversarial examples." [Online]. Available: http://arXiv.org/abs/1802.09707
[46] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, May 2018.
[47] D. Zhu, J. Bieger, G. Garcia Molina, and R. M. Aarts, "A survey of stimulation methods used in SSVEP-based BCIs," *Comput. Intell. Neurosci.*, vol. 2010 Jan. 2010, Art. no. 702357.