International Journal of Service and Knowledge Management
International Institute of Applied Informatics
2024, Vol. 8, No. 2, IJSKM813

Adversarial Attacks for Time Series Classification using Partial Perturbations

Jun Teraoka [*], Keiichi Tamura [*]

Abstract

Adversarial attacks using adversarial examples have recently become a significant threat that intentionally misleads deep-learning models beyond human recognition. Adversarial examples have primarily been studied in the field of image recognition; however, they have recently been applied in other fields, including time series data classification. To generate adversarial examples, small perturbations unrecognizable by humans are typically added to all the data regions. However, adding perturbations to the entire time series data results in time series data that are clearly manipulated for time series classification. In this case, adversarial attacks are immediately apparent to humans and do not pose a significant threat. This study shows that unidentifiable adversarial examples of time series can be identified as adversarial examples in time series data classification by adopting partial perturbations. The fast gradient sign method (FGSM) and projected gradient descent (PGD) attack methods, which are originally proposed for generating adversarial examples of image data, are applied to time series data classification models. In this study, partial-FGSM and partial-PGD attacks are proposed which utilize only a part of the perturbations to generate fewer unreliable adversarial examples of time series data that are easily recognized as adversarial examples. To evaluate partial-FGSM and partial-PGD attacks, the 2 Class-Based-Detecting adversarial detection method is employed, as its effectiveness for protecting adversarial attacks against time series classification has been proven. The performance is evaluated, and the results show that attacks are possible with a small degradation in attack performance for some datasets, even if the perturbation ratio is 1/10.

Keywords: Adversarial Examples, Time Series Data, Deep Learning, Security

1 Introduction

Deep learning is a machine learning method that reproduces the mechanism of human neurons, and it has become an indispensable high-performance technology for various applications including, image recognition, speech recognition, natural language processing, and time series data analysis [1]. However, deep learning is vulnerable to attacks that employ adversarial examples, a threat with a highly probability of causing model misclassifications [2][3]. Adversarial examples are data created by adding extremely small perturbations to input data that are imperceptible to

[*] Graduate School of Information Sciences, Hiroshima City University, Hiroshima, Japan

humans, intended to mislead the output of a deep learning model. Adversarial attacks using adversarial examples have been extensively studied in the context of image recognition [4][5][6][7]. Recently, researchers have analyzed the threat with regard to other deep learning applications, including speech recognition [8], natural language processing [9], and time series data classification [10][11][12][13]. The vulnerability of this model poses a threat when deep learning is applied to an actual system.

This study focuses on adversarial attacks using adversarial examples for time series data classification, which is the task of predicting class labels for unclassified time series data. Adversarial examples for time series data classification can be generated in the same manner as those for image recognition [10]. In addition, studies have evaluated attack performance and methods for defense against adversarial examples [11][12][13]. Time series data are common in several contexts, including sensor observations, biometric signals, and motion data. As time series data classification is an important task, the identification of adversarial examples of time series data is essential in order to mitigate the risk their potential exploitation poses to sensing-centric applications.

Methods for generating adversarial examples that are commonly used in image recognition can be used to easily generate manipulated time series data. In the case of image data, perturbing the entire dataset does not facilitate the easy determination of whether perturbations have been introduced because the noise is minute and difficult to detect. However, in the case of time series data classification, the addition of perturbations generates adversarial examples that are easy to detect, including perturbations that are perceptible to humans. This is because noise added to pixels in image data cannot be recognized because it is a small change in tint, but noise added to time series data can be easily recognized by comparing it with normal data.

This study demonstrates that adversarial examples for time series data classification generated by methods commonly used in image recognition can be easily identified. Additionally, if partial perturbations are used to generate adversarial examples, then adversarial examples that are seemingly indistinguishable from adversarial examples that are perturbed can be generated [14]. In our previous study [14], only one model of the attack target was used, and the evaluation of individual data was not sufficiently demonstrated. In this study, another model was added as an attack target, and its attack capability was demonstrated via evaluation experiments using individual datasets. To the best of our knowledge, no previous studies have focused on the perceptibility of these attacks to humans. The effectiveness of using partial perturbations as a practical and potentially threatening attack method is evaluated.

The remainder of this paper is organized as follows: Section 2 describes the assumptions introduced in this study. Section 3 discusses adversarial examples of timeseries data. Section 4 describes adversarial attacks in the context of time series data classification using partial perturbations. Section 5 presents the experimental evaluation. Section 6 concludes the paper.

2 Preliminaries

This section outlines the prerequisite knowledge for adversarial attacks using adversarial examples. First, the prerequisite knowledge of the attacker and attack types are described as the attack settings. Next, typical methods for generating adversarial examples are presented. Finally, defense methods against adversarial attacks that use adversarial examples for time series data classification are described.

2.1 Attack Setting

2.1.1 Adversary's Knowledge

Adversarial attacks using adversarial examples are classified into two categories based on the adversary's knowledge: white-box and black-box attacks. This classification is established because the way adversarial examples are generated depends on prerequisite knowledge.

White-box attacks assume that the adversary possesses all information regarding the target model, such as the training data, model structure, hyperparameters, and network weights. It is relatively easy to create adversarial examples that result in erroneous outputs if the model structure and weight parameters are known. Therefore, these adversarial examples represent a more significant threat to the model than those generated via black-box attacks.

Conversely, black-box attacks assume that the adversary has no information regarding the target model and knows only the input and output. Although most attack methods are white-box attacks, the black-box assumption is applicable. This is because studies have demonstrated that an adversary example generated for one model is valid for another with different training data and structures [15].

2.1.2 Attack Categories

There are two categories of attack objectives: targeted and non-targeted attacks.

Targeted attacks aim to direct the prediction of a model toward a specific class. In general, adversarial examples can be generated by reducing the loss of a targeted class during an attack. For example, a targeted attack manipulates data classified under classification label 1 into data classified under classification label 2. In general, the adversarial examples generated for this type of attack constitute a significant threat to real-world applications.

Non-targeted attacks aim to mislead the predictions of a model, regardless of the classes. In general, adversarial examples can be generated by increasing the losses of the correct answer classes during attacks. Typically, an adversarial example that is more powerful than the target attack can be generated.

2.2 Attack Methods

2.2.1 FGSM

The FGSM is a typical adversarial example generation method proposed by Goodfellow et al. [3]. In deep learning, the gradient of the loss function is used to update the weights of the network to ensure that the loss decreases. However, the FGSM does not update the weights of the network but varies the input data such that the loss increases. Using the FGSM, a hostile sample $\tilde{x}$ can be expressed as follows:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \epsilon sign\left(\nabla_x J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right), \tag{1}$$

where $\boldsymbol{x}$ denotes the input data, ϵ denotes a parameter for adjusting the perturbation magnitude, $\boldsymbol{\theta}$ denotes a parameter of the model, y denotes the correct label for $\boldsymbol{x}$, and $J(\boldsymbol{\theta}, \boldsymbol{x}, y)$ denotes the loss function.

Although the FGSM is proposed as an attack for image classification models, it can be applied to time series data classification models by using time series data instead of image data for $\boldsymbol{x}$. Whereas images are perturbed pixel-by-pixel, and time series data can be perturbed for each element of the series to generate adversarial examples.

2.2.2 Projected Gradient Descent (PGD)

PGD is an effective attack method proposed by Madry et al. [16]. Whereas the FGSM only applies perturbation to the input data once, PGD generates an adversarial example by using the FGSM repeatedly with a step size of α. The PGD is expressed as follows:

$$\tilde{\boldsymbol{x}}_{t+1} = Clip_{(\tilde{x}_t+\epsilon, \tilde{x}_t-\epsilon)}\left(\tilde{\boldsymbol{x}}_t + \alpha sign\left(\nabla_{\tilde{x}_t} J(\boldsymbol{\theta}, \tilde{\boldsymbol{x}}_t, y)\right)\right), \tag{2}$$

where $Clip_{(\tilde{x}_t+\epsilon, \tilde{x}_t-\epsilon)}$ is a transformation process that ensures the magnitude of the perturbation does not exceed ϵ. FGSM and PGD methods are inherently white-box and non-targeted attack methods, but they can be extrapolated to black-box or targeted attacks.

2.3 Defense Methods

The implementation of defense mechanisms against adversarial attacks using adversarial examples can be approached in two ways: 1) adversarial training, which improves the robustness of the system by including adversarial examples in the training data [3][6][16] and 2) adversarial detection, which detects adversarial examples based on differences in the model behavior or input data characteristics [7][12][13]. Compared with other methods, adversarial training can achieve greater robustness against various types of adversarial attacks using adversarial examples. However, this requires the modification of the target model, decreases the classification accuracy of the original samples, and can cause overfitting. Adversarial detection is disadvantageous when compared with adversarial training because its performance is easily affected by the type of attacker and adversarial example; however, it does not degrade the classification accuracy.

The 2 Class-Based-Detecting (2CB) adversarial detection method was used in this study [13]. This method is based on differences in the characteristics of the input data. This approach constructs a deep learning model for detection that performs 2 class classification if the sample is an adversarial example, in addition to the model used for prediction.

3 Adversarial Examples of Time Series

In this section, we discuss the characteristics of time series adversarial examples. An adversarial example of image data is extremely difficult for humans to identify, as shown in Figure 1 [17]. Time series data from the UCR time series classification archive [18] show the motion data of gun pointing, with the horizontal and vertical axes representing the time and position, respectively, as shown in Figure 2. The human eye can easily distinguish an adversarial example of time series data if the data are smooth, even with slight perturbations, as shown in Figure 2. As in the FGSM, imposing perturbations of a certain magnitude on the entire sample can cause discriminable changes that are specific to adversarial examples and oscillate at a certain amplitude. Thus, in cases involving time series data, adversarial examples generated using methods such as the FGSM and PGD, which apply perturbations to the entire dataset, generate data that appear unnatural, even under slight perturbations.

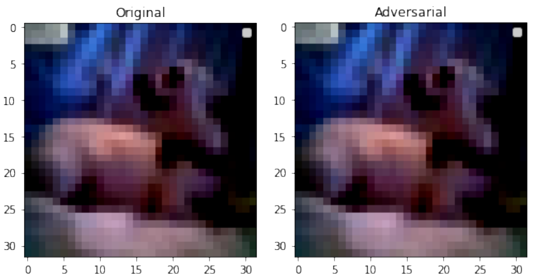


Figure 1: Image adversarial example

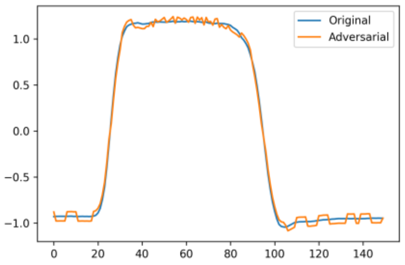


Figure 2: Time series adversarial example

4 Attacks with Partial Perturbations

Two attacks with partial perturbations are proposed in this section: partial-FGSM and partial-PGD. These attacks utilize only some perturbations to generate fewer unnatural adversarial examples for time series data.

4.1 Partial-FGSM

The partial-FGSM uses only part of the perturbation generated by the FGSM to generate an adversarial example, which is expressed as follows:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \epsilon\, \boldsymbol{m} \circ sign\big(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\big), \tag{3}$$

where $\boldsymbol{m}$ denotes a mask matrix or vector that determines the part of the perturbation $\epsilon sign\big(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\big)$ that is added to the original sample of the same size as the input data $\boldsymbol{x}$ and takes on a value of either 0 or 1.

4.2 Partial-PGD

The partial-PGD generates an adversarial example by repeating the partial-FGSM multiple times with a step size of α. As with the PGD, the perturbation is clipped by $Clip_{(\tilde{\boldsymbol{x}}_t+\epsilon,\tilde{\boldsymbol{x}}_t-\epsilon)}$ such that the magnitude of the perturbation is not greater than ϵ owing to repetition. The same $\boldsymbol{m}$ was used for all the steps. The partial-PGD is expressed as follows:

$$\tilde{\boldsymbol{x}}_{t+1} = Clip_{(\tilde{\boldsymbol{x}}_t+\epsilon,\tilde{\boldsymbol{x}}_t-\epsilon)}\left(\tilde{\boldsymbol{x}}_t + \alpha\boldsymbol{m} \circ sign\left(\nabla_{\tilde{\boldsymbol{x}}_t} J(\boldsymbol{\theta}, \tilde{\boldsymbol{x}}_t, y)\right)\right) \tag{4}$$

where $\boldsymbol{m}$ denotes a mask matrix or vector that determines the part of the perturbation $\epsilon sign\left(\nabla_{\tilde{\boldsymbol{x}}_t} J(\boldsymbol{\theta}, \tilde{\boldsymbol{x}}_t, y)\right)$ that is added to the original sample of the same size as the input data $\boldsymbol{x}$ and takes on a value of either 0 or 1.

4.3 Creating Partial Perturbation

The $\boldsymbol{m}$ that determines the interval to be partially perturbed results in the best attack performance. As a simple example of determining $\boldsymbol{m}$ for a single zone, the data can be partitioned into s equal zones when the perturbation range is $1/s$ of the data size, and the best zone can be determined by applying perturbation to each zone. Here, $\boldsymbol{m}$ can be searched easily in multiple zones by repeatedly adding perturbations to $1/s$ random points.

5 Experiment

The experimentally evaluated performances of the two attacks are presented in this section. First, the experimental setup is described. Next, adversarial examples of time series data generated using the existing and proposed methods are compared. Finally, the attack performance and effectiveness of 2CB detection as a defense method against adversarial examples are evaluated.

5.1 Experimental Setup

5.1.1 Dataset

The UCR time-series classification archive (2018) dataset, which is a benchmark for time series data classification problems, was used in this study [18]. The UCR time series classification archive was categorized into old and new archives, which contained 85 and 43 datasets, respectively. Only old archives were used in this experiment. The UCR time-series classification archive provides separate training and test sets. The data were shuffled in each set; however, the default division was not changed.

5.1.2 Model

The fully convolutional network (FCN) and residual network (ResNet), which are typical time-series data classification models using deep learning, were used in this study [19].

The FCN comprises three convolution layers, each with batch normalization and a rectified linear unit (ReLU) activation function. Furthermore, the FCN replaces the typical final fully convolutional layer with a global average pooling (GAP) layer. Meanwhile, ResNet comprises three residual blocks, a GAP layer, and a softmax classifier. Each residual block comprises three convolutions, each with batch normalization and ReLU activation functions. Unlike the FCN, ResNet features linear shortcuts between the residual blocks, thus rendering it difficult for the gradient to vanish.

The model parameters and source code for the experiment were obtained from [20]. Table 1 lists the architecture and optimization hyperparameters of each model.

Table 1: Architecture and optimization hyperparameters for each model

Methods	Layers	Conv	Epochs	Batch	Learning rate	Decay
FCN	5	3	2000	16	0.001	0.0
ResNet	11	9	1500	16	0.001	0.0

5.2 Generating Adversarial Examples

The existing FGSM and PGD methods as well as the proposed partial-FGSM and partial-PGD methods were used to generate adversarial examples. For all attack methods, the perturbation magnitude was set to 0.1. The number of iterations was set to 40 for both the PGD and partial -PGD methods. In this experiment, the perturbation ranges of the partial-FGSM and partial-PGD methods were set to 1/10 of the data length. The data were partitioned into ten equal-length zones, and the zone with the highest misclassification probability for each dataset was adopted. The experiment was conducted using CleverHans, which is a library of adversarial examples [21]. Parameters other than those for the FGSM and PGD were obtained from the default parameters of CleverHans.

Figures 3 and 4 show adversarial examples from the OliveOil dataset generated from the UCR time series classification archives. These graphs represent the food spectrograms of olive oil, where the horizontal and vertical axes represent time and frequency, respectively. Figure 3 presents a comparison of adversarial examples based on the FGSM and partial-FGSM. Figure 4 shows adversarial examples of PGD and partial-PGD. Noise can be easily identified for the FGSM and PGD methods that apply all perturbations, as shown in Figures 3 and 4, respectively. However, the noise for the partial-FGSM and partial-PGD methods could not be easily identified because most of the data remained unchanged from the original data.

Figures 5(a) and 5(b) show graphs comparing the values of the mean absolute error of the original data and the adversarial example for each of the 85 datasets in the UCR based on FGSM, partial-FGSM, PGD, and partial-PGD. The mean absolute error (MAE) values for the partial-FGSM were smaller than those for the FGSM, and the overall MAE values for the PGD were smaller than those for the partial-PGD, thus rendering identification difficult when partial perturbations are applied, as shown in Figures 3 and 4.

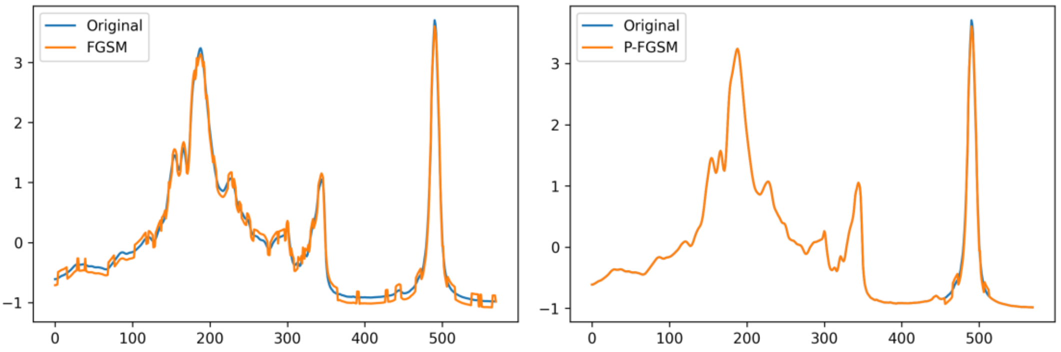


Figure 3: Adversarial example for FGSM (left) and partial-FGSM (right)

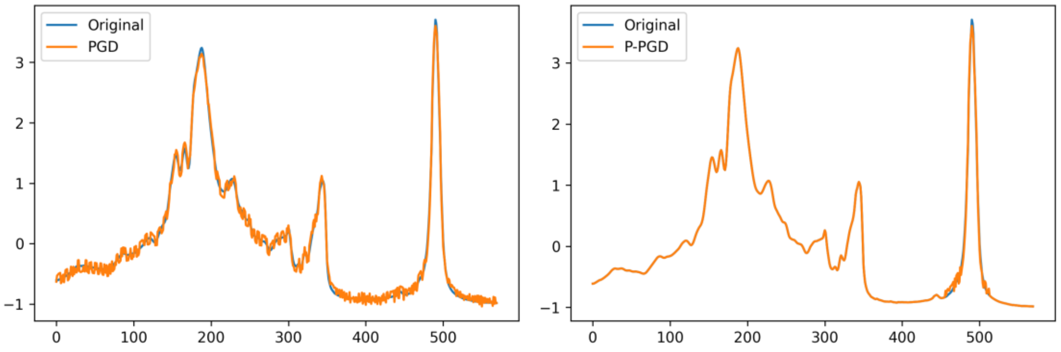
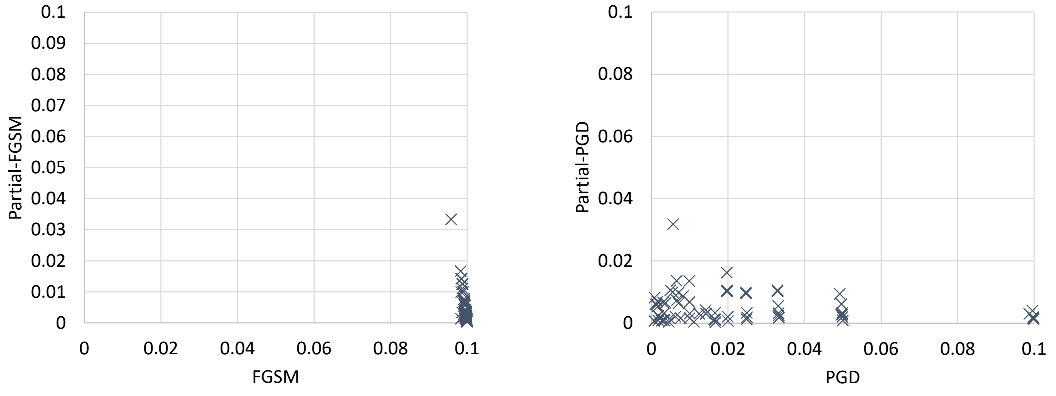


Figure 4: Adversarial example for PGD (left) and partial-PGD (right)



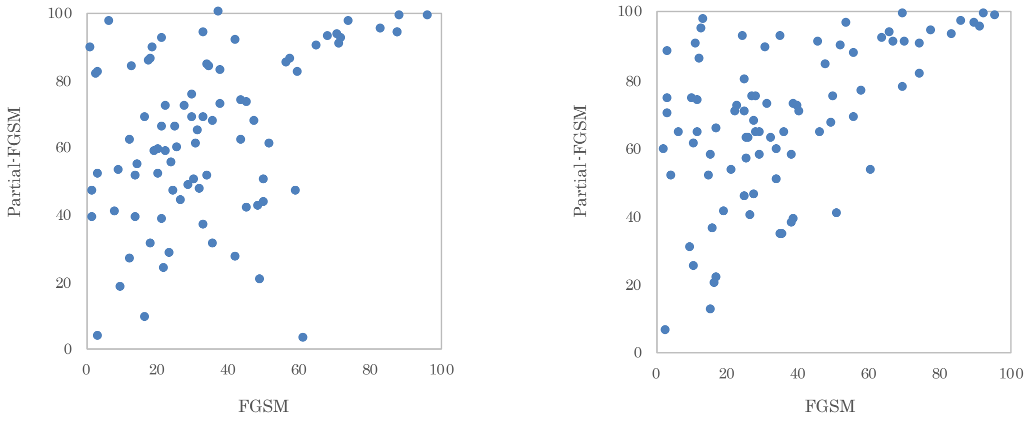
(a) MAEs of FGSM and partial-FGSM

(b) MAEs of PGD and partial-PGD

Figure 5: Comparison of MAEs

5.3 Performance of Attacks

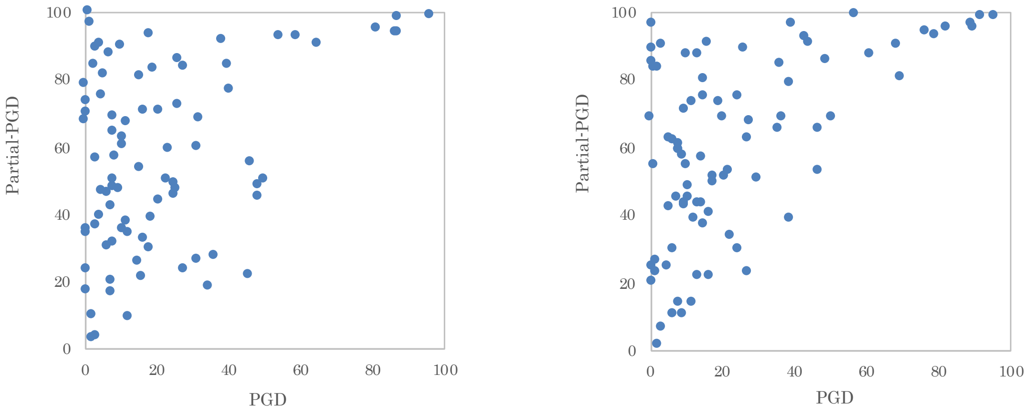
The accuracies of each adversarial example generated with all perturbations and those generated with partial perturbations for all datasets were compared. The classification accuracy for each dataset using the adversarial input model is plotted in Figures 6 and 7. Adversarial inputs with all perturbations are on the horizontal axis, and those with partial perturbations are on the vertical axis. A comparison between the FGSM and partial-FGSM methods is shown in Figure 6, while a comparison between the PGD and partial-PGD methods is shown in Figure 7. The attack performances of the adversarial examples with all perturbations were higher for most datasets, as shown in Figures 5 and 6. However, the detailed results indicate that the attack performance of the partial perturbation was not sufficiently low and could be ignored for several datasets, as listed in Tables 2 and 3. The number of datasets for which the accuracy against partial perturbations (denoted by P-FGM and P-PGD in Tables 2 and 3) was less than half of that against the original data (denoted by ORG in Tables 2 and 3) was 23 for FCN and 15 for ResNet. This poses a significant threat to the majority of systems. In some datasets, such as OliveOil and Strawberry, the partial perturbation achieved the same attack performance as the entire perturbation.



(a) FCN

(b) ResNet

Figure 6: Accuracy comparison between FGSM and partial-FGSM (%) for: (a) FCN and (b) ResNet



(a) FCN

(b) ResNet

Figure 7: Accuracy comparison between PGD and partial-PGD (%) for: (a) FCN and (b) ResNet

Table 2: Accuracy of each dataset (%). (ORG denotes original samples, P-FGM denotes partial-FGSM, and P-PGD denotes partial-PGD) (1/2)

Dataset	**FCN**					**ResNet**				
	ORG	**FGSM**	**PGD**	**P-FGM**	**P-PGD**	**ORG**	**FGSM**	**PGD**	**P-FGM**	**P-PGD**
Adiac	83.89	1.79	0.77	46.88	34.02	85.42	3.33	1.28	74.22	54.69
ArrowHead	84.57	37.71	10.86	72.66	60.16	83.43	25.71	9.71	62.50	43.43
Beef	73.33	33.33	3.33	36.67	36.67	73.33	26.67	13.33	40.00	43.33
BeetleFly	95.00	30.00	5.00	75.00	75.00	85.00	25.00	15.00	80.00	80.00
BirdChicken	90.00	65.00	10.00	90.00	90.00	90.00	50.00	15.00	75.00	75.00
CBF	99.56	88.56	86.89	98.63	98.63	99.22	92.67	92.00	99.02	98.44
Car	90.00	21.67	6.67	38.33	30.00	95.00	21.67	5.00	53.33	25.00
ChlorineConcentration	81.93	14.06	12.14	39.06	34.38	85.13	10.70	8.05	25.00	14.06
CinCECGTorso	82.97	24.93	6.38	65.63	46.35	84.13	25.44	7.90	70.31	45.31
Coffee	100.00	3.57	0.00	82.14	78.57	100.00	53.57	39.29	96.43	96.43
Computers	80.40	43.60	20.80	73.44	70.83	80.80	38.80	20.40	72.40	68.75
CricketX	80.51	22.31	8.21	71.88	68.75	78.72	31.28	9.74	72.56	70.83
CricketY	77.18	16.67	8.21	68.49	64.06	81.79	22.31	6.67	70.31	62.11
CricketZ	79.74	21.28	10.51	65.63	62.50	81.28	29.23	8.21	64.06	59.38
DiatomSizeReduction	30.07	42.16	36.28	27.34	27.34	54.90	35.29	6.54	34.38	29.69
DistalPhalanxOutlineAgeGroup	71.94	24.46	20.86	46.88	43.75	75.54	29.50	20.86	57.81	51.56
DistalPhalanxOutlineCorrect	75.36	30.44	25.00	50.00	48.91	78.26	34.06	21.74	59.38	53.13
DistalPhalanxTW	68.35	22.30	8.63	58.27	56.84	68.35	15.83	10.79	57.55	48.44
ECG200	89.00	38.00	19.00	82.81	82.81	89.00	48.00	36.00	84.38	84.38
ECG5000	94.11	72.11	38.11	92.04	91.78	93.44	70.22	61.09	90.62	87.62
ECGFiveDays	98.14	34.03	2.56	84.38	84.38	94.43	11.03	5.58	60.94	62.50
Earthquakes	73.38	33.09	31.66	68.35	68.35	74.10	40.29	36.69	70.31	68.75
ElectricDevices	69.69	45.46	31.35	41.41	26.56	72.22	51.06	27.31	40.63	23.44
FaceAll	95.03	68.11	54.08	92.88	92.54	82.54	74.32	69.76	81.43	80.89
FaceFour	92.05	18.18	6.82	85.94	87.50	95.45	65.91	43.18	93.75	92.19
FacesUCR	94.59	70.63	59.22	92.97	92.97	95.56	83.76	79.37	92.97	92.97
FiftyWords	64.18	9.45	7.91	53.13	50.00	70.55	12.09	8.35	64.06	59.38
Fish	96.00	12.57	0.57	61.72	35.16	98.86	12.57	0.00	85.94	68.75
FordA	91.67	51.59	46.29	60.94	55.06	93.56	45.91	13.41	90.63	87.20
FordB	78.64	47.65	22.72	67.19	50.00	82.35	28.40	19.26	75.00	73.44
GunPoint	100.00	18.67	5.33	89.06	81.25	99.33	52.00	1.33	89.84	83.59
Ham	68.57	31.43	31.43	64.76	60.00	72.38	27.62	27.62	67.62	67.62
HandOutlines	88.11	35.95	34.32	31.25	18.38	93.24	35.95	6.76	34.38	10.94
Haptics	47.73	18.51	15.91	30.73	21.35	51.30	19.16	13.31	41.15	22.27
Herring	51.56	59.38	48.44	46.88	48.44	56.25	60.94	46.88	53.13	53.13
InlineSkate	33.82	9.64	7.27	17.97	16.41	39.09	17.27	11.82	21.88	14.06
InsectWingbeatSound	39.29	12.48	7.58	26.56	20.31	47.78	16.36	15.05	35.94	37.50
ItalyPowerDemand	95.92	87.85	86.30	93.75	93.75	95.92	91.45	89.70	95.31	95.31
LargeKitchenAppliances	89.07	59.47	15.73	82.03	80.94	89.60	69.87	38.93	77.34	78.91
Lightning2	75.41	27.87	26.23	72.13	72.13	75.41	55.74	50.82	68.85	68.85
Lightning7	82.19	30.14	16.44	68.75	70.31	83.56	27.40	24.66	75.00	75.00
Mallat	96.46	17.57	3.28	85.55	56.64	97.27	24.39	2.60	92.29	83.38
Meat	45.00	50.00	48.33	43.33	45.00	95.00	3.33	1.67	70.00	23.33
MedicalImages	79.87	35.66	11.58	67.19	67.19	75.26	49.34	35.79	67.19	65.63
MiddlePhalanxOutlineAgeGroup	56.49	23.38	27.92	28.13	23.44	57.79	38.31	24.68	37.50	29.69
MiddlePhalanxOutlineCorrect	82.13	20.28	18.21	51.56	29.69	82.82	25.77	16.84	56.25	40.63
MiddlePhalanxTW	50.65	22.08	14.94	23.44	25.78	48.05	9.74	16.88	30.47	21.88
MoteStrain	92.65	71.57	64.86	90.63	90.63	92.49	74.60	68.45	90.10	90.10

Table 3: Accuracy of each dataset (%). (ORG denotes original samples, P-FGM denotes partial-FGSM, and P-PGD denotes partial-PGD) (2/2)

Dataset	**FCN**					**ResNet**				
	ORG	**FGSM**	**PGD**	**P-FGM**	**P-PGD**	**ORG**	**FGSM**	**PGD**	**P-FGM**	**P-PGD**
NonInvasiveFetalECGThorax1	95.93	2.09	2.09	39.06	9.72	95.22	4.73	0.56	51.56	25.00
NonInvasiveFetalECGThorax2	95.32	3.46	0.46	51.56	17.03	95.01	2.49	0.61	59.38	20.31
OSULeaf	98.35	13.22	0.00	83.59	67.97	97.93	13.22	0.83	94.53	89.06
OliveOil	76.67	3.33	3.33	3.33	3.33	80.00	16.67	3.33	20.00	6.67
PhalangesOutlinesCorrect	81.00	34.15	18.53	51.43	38.70	82.52	38.23	17.60	57.81	51.30
Phoneme	32.44	16.61	12.40	9.38	9.38	32.07	15.40	9.28	12.50	10.94
Plane	100.00	37.14	0.95	100.00	100.00	100.00	69.52	57.14	99.05	99.05
ProximalPhalanxOutlineAgeGroup	83.90	48.78	7.32	42.19	42.19	82.93	32.68	10.24	62.50	54.69
ProximalPhalanxOutlineCorrect	89.35	14.09	10.65	51.04	35.40	90.38	23.02	9.97	71.82	42.96
ProximalPhalanxTW	76.59	23.90	4.39	55.12	39.06	76.10	6.83	5.37	64.06	42.19
RefrigerationDevices	50.93	32.27	25.33	47.47	47.20	50.93	34.13	30.13	50.40	50.67
ScreenType	61.87	28.80	25.07	48.44	45.31	63.20	28.00	22.40	46.09	33.59
ShapeletSim	61.67	48.89	45.56	20.31	21.88	88.33	11.67	11.67	73.44	73.44
ShapesAll	89.50	3.17	0.67	81.25	70.17	91.50	3.67	0.83	87.95	85.07
SmallKitchenAppliances	78.40	45.60	15.47	72.80	53.33	78.93	40.00	17.60	72.00	49.87
SonyAIBORobotSurface1	96.34	83.03	81.20	94.87	94.87	97.17	77.87	76.54	94.27	94.27
SonyAIBORobotSurface2	97.17	87.83	87.09	93.75	93.75	98.22	90.14	89.19	96.35	96.35
StarLightCurves	96.87	57.72	40.21	85.94	76.56	97.26	57.94	46.90	76.56	65.63
Strawberry	97.57	61.08	2.43	3.13	3.13	97.84	2.70	2.16	6.25	1.56
SwedishLeaf	97.12	21.60	3.20	92.00	89.44	95.84	35.04	16.32	92.19	90.63
Symbols	94.87	42.01	4.52	91.74	90.18	91.86	31.26	10.15	89.06	87.19
SyntheticControl	98.67	96.33	96.00	98.67	98.67	99.67	95.67	95.67	98.67	98.67
ToeSegmentation1	96.49	33.33	17.98	93.75	93.23	95.61	67.11	44.30	90.63	90.63
ToeSegmentation2	93.08	34.62	27.69	83.59	83.59	89.23	56.15	49.23	87.50	85.94
Trace	100.00	74.00	26.00	96.88	85.94	100.00	64.00	26.00	92.00	89.06
TwoLeadECG	100.00	6.85	1.84	96.88	96.88	99.91	11.50	3.25	90.10	90.10
TwoPatterns	86.95	56.65	39.65	84.67	84.25	99.23	86.03	82.30	96.88	95.31
UwaveGestureLibraryAll	81.91	14.66	0.87	54.69	23.44	84.51	17.09	1.87	65.63	26.56
UwaveGestureLibraryX	75.60	25.94	4.83	59.38	46.88	76.49	26.41	8.29	62.50	60.94
UwaveGestureLibraryY	63.90	26.91	11.64	43.75	37.50	65.49	25.29	12.48	45.31	39.06
UwaveGestureLibraryZ	72.53	20.66	9.58	58.85	47.40	74.85	28.50	9.07	64.06	57.81
Wafer	99.74	1.10	0.41	89.06	73.44	99.76	13.60	0.93	97.32	96.65
Wine	50.00	50.00	50.00	50.00	50.00	61.11	38.89	38.89	38.89	38.89
WordSynonyms	55.96	8.31	8.15	40.63	31.25	62.54	15.05	10.66	51.56	45.31
Worms	77.92	19.48	7.79	58.44	48.05	81.82	10.39	14.29	74.03	57.14
WormsTwoClass	76.62	31.17	23.38	60.94	59.38	72.73	36.36	27.27	64.06	62.50
Yoga	83.63	43.80	16.43	61.72	32.81	85.47	46.43	14.53	64.06	43.61

5.4 Effectiveness of 2CB detection

The effectiveness of the 2CB detection method was evaluated using adversarial examples with partial perturbations. The same structure was used for the detection and prediction models. Adversarial examples generated using FGSM were used as training data for the adversarial example class. For the evaluation, adversarial examples were generated in the same manner as in the other experiments.

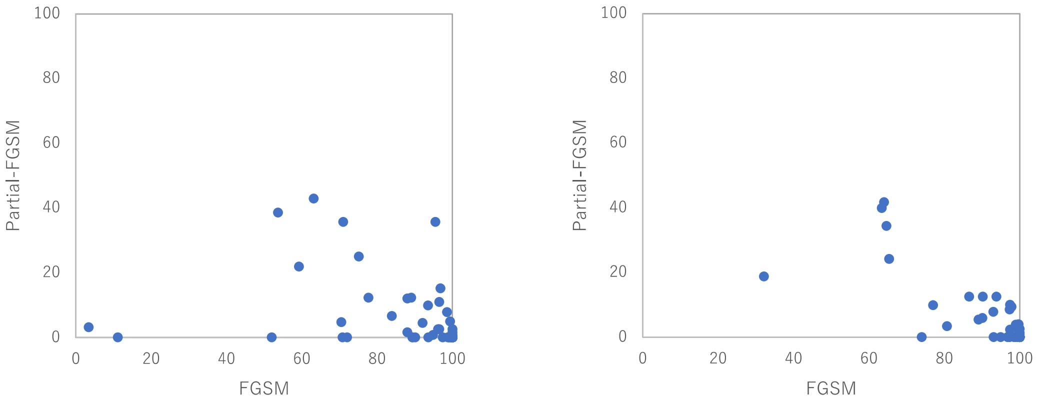
The results demonstrating the accuracy of the 2CB detection model for each dataset are listed in Tables 4 and 5, and an accuracy comparison between complete perturbation and partial perturbation is shown in Figures 8 and 9. Tables 4 and 5 show the detection rates of whether the adversarial examples generated for the 85 datasets in the UCR were identified via 2CB detection. For example, for the “Adiac” dataset, the FGSM and PGD showed 100%, thus indicating that all adversarial examples were detected via 2CB detection, whereas partial-FGSM (denoted by P-FGM in Tables 4 and 5) and partial-PGD (denoted by P-PGD in Tables 4 and 5) showed 0%, thus indicating that they were not detected. These values were the same as the detection rates when focusing on values other than the original data (denoted by ORG in Tables 4 and 5). The detection rate was the highest for the FGSM, which was used to train the detection model, followed by PGD, which was not used for training but utilized complete perturbations, as summarized in Tables 4 and 5. Conversely, the partial-perturbation methods, i.e., partial-FGSM and partial-PGD, were almost undetectable most of their data were exactly the same as those of the original samples. Thus, partial perturbation can be more difficult to defend than complete perturbation, particularly when the method is based on input data features, such as 2CB detection.

Table 4: Accuracy for 2CB detection models on each dataset (%). (ORG denotes original samples, P-FGM denotes partial-FGSM, and P-PGD denotes partial-PGD) (1/2)

Dataset	**FCN**					**ResNet**				
	ORG	**FGSM**	**PGD**	**P-FGM**	**P-PGD**	**ORG**	**FGSM**	**PGD**	**P-FGM**	**P-PGD**
Adiac	99.74	100.00	100.00	0.00	0.00	98.98	100.00	73.15	0.00	0.00
ArrowHead	100.00	100.00	88.57	0.00	0.00	100.00	100.00	90.86	0.00	0.00
Beef	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
BeetleFly	100.00	100.00	45.00	0.00	0.00	100.00	100.00	85.00	0.00	0.00
BirdChicken	100.00	100.00	80.00	0.00	0.00	100.00	100.00	95.00	0.00	0.00
CBF	56.22	63.11	57.56	42.97	42.97	56.44	63.44	61.78	39.84	39.06
Car	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
ChlorineConcentration	100.00	99.04	81.51	0.03	0.03	100.00	99.84	94.12	0.00	0.00
CinCECGTorso	99.93	89.35	55.94	0.00	0.00	99.78	100.00	99.06	0.00	0.00
Coffee	100.00	100.00	100.00	0.00	0.00	100.00	100.00	96.43	0.00	0.00
Computers	99.60	94.80	98.40	0.78	6.40	99.60	96.80	70.80	0.00	0.00
CricketX	95.64	92.05	90.26	4.43	5.21	97.69	97.44	85.39	2.31	3.08
CricketY	98.21	93.59	85.39	0.00	0.00	95.90	98.97	84.10	3.91	3.13
CricketZ	88.97	98.46	93.85	7.81	9.38	91.03	97.44	89.23	10.00	7.81
DiatomSizeReduction	99.02	100.00	100.00	0.78	0.78	99.02	100.00	99.02	0.78	0.78
DistalPhalanxOutlineAgeGroup	100.00	100.00	89.21	0.00	0.00	100.00	100.00	92.81	0.00	0.00
DistalPhalanxOutlineCorrect	100.00	100.00	99.64	0.00	0.00	99.64	100.00	76.45	0.00	0.00
DistalPhalanxTW	100.00	100.00	95.68	0.00	0.00	100.00	99.28	85.61	0.00	0.00
ECG200	90.00	88.00	84.00	12.00	12.00	92.00	93.00	82.00	7.81	7.81
ECG5000	97.96	96.42	76.47	2.58	2.44	97.31	99.51	84.53	2.82	2.76
ECGFiveDays	99.77	99.07	95.94	0.00	0.00	99.77	100.00	71.43	0.00	0.00
Earthquakes	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
ElectricDevices	94.40	70.80	84.89	0.00	0.00	94.76	97.34	78.47	0.00	0.00
FaceAll	97.40	89.35	81.01	0.00	0.00	97.52	93.08	82.54	0.00	0.00
FaceFour	94.32	70.46	67.05	4.69	6.25	96.59	80.68	68.18	3.41	3.13
FacesUCR	85.71	96.73	90.98	15.18	14.06	85.95	86.59	74.10	12.50	12.50
FiftyWords	100.00	99.78	99.78	0.00	0.00	100.00	100.00	100.00	0.00	0.00
Fish	99.43	100.00	100.00	0.00	0.00	99.43	100.00	100.00	0.00	0.00
FordA	100.00	99.92	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
FordB	100.00	99.38	100.00	4.95	12.50	100.00	99.75	100.00	0.00	0.00
GunPoint	99.33	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
Ham	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
HandOutlines	100.00	100.00	97.57	0.00	0.00	100.00	100.00	0.00	0.00	0.00
Haptics	100.00	99.68	100.00	0.00	0.00	99.68	100.00	80.84	0.00	0.00

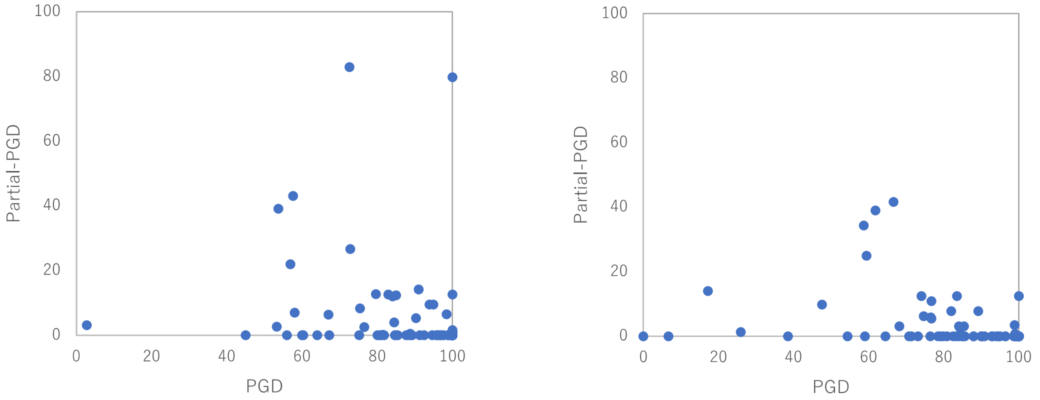
Table 5: Accuracy for 2CB detection models on each dataset (%). (ORG denotes original samples, P-FGM denotes partial-FGSM, and P-PGD denotes partial-PGD) (2/2)

Dataset	FCN					ResNet				
	ORG	FGSM	PGD	P-FGM	P-PGD	ORG	FGSM	PGD	P-FGM	P-PGD
Herring	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
InlineSkate	100.00	100.00	64.00	0.00	0.00	100.00	100.00	98.73	0.00	0.00
InsectWingbeatSound	100.00	99.55	99.95	0.00	0.00	100.00	99.85	99.95	0.00	0.00
ItalyPowerDemand	87.27	70.94	72.60	35.73	82.81	85.03	74.05	64.53	0.00	0.00
LargeKitchenAppliances	96.00	95.47	100.00	35.73	79.69	96.53	97.33	100.00	8.53	12.50
Lightning2	93.44	93.44	75.41	9.84	8.20	90.16	77.05	47.54	9.84	9.84
Lightning7	87.67	89.04	84.93	12.33	12.33	94.52	89.04	76.71	5.48	5.48
Mallat	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
Meat	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
MedicalImages	99.74	100.00	98.95	0.00	0.00	100.00	100.00	99.74	0.00	0.00
MiddlePhalanxOutlineAge-Group	100.00	100.00	99.35	0.00	0.00	100.00	100.00	85.07	0.00	0.00
MiddlePhalanxOutlineCorrect	100.00	100.00	100.00	0.00	0.00	100.00	100.00	90.03	0.00	0.00
MiddlePhalanxTW	100.00	100.00	98.70	0.00	0.00	100.00	100.00	79.22	0.00	0.00
MoteStrain	85.06	77.72	79.63	12.34	12.66	85.30	90.18	83.55	12.50	12.50
NonInvasiveFetalECGThorax1	100.00	100.00	100.00	0.00	0.00	100.00	100.00	100.00	0.00	0.00
NonInvasiveFetalECGThorax2	100.00	100.00	99.85	0.00	0.00	100.00	100.00	100.00	0.00	0.00
OSULeaf	100.00	100.00	84.71	0.00	0.00	100.00	100.00	88.02	0.00	0.00
OliveOil	100.00	100.00	100.00	0.00	0.00	100.00	100.00	90.00	0.00	0.00
PhalangesOutlinesCorrect	100.00	100.00	91.26	0.00	0.00	99.88	100.00	58.97	0.00	0.00
Phoneme	90.67	83.81	57.96	6.59	6.86	89.66	90.08	76.53	5.94	5.94
Plane	100.00	100.00	87.62	0.00	0.00	100.00	100.00	99.05	0.00	0.00
ProximalPhalanxOutlineAge-Group	100.00	100.00	75.12	0.00	0.00	100.00	100.00	94.63	0.00	0.00
ProximalPhalanxOutlineCorrect	100.00	100.00	100.00	0.00	0.00	100.00	100.00	38.49	0.00	0.00
ProximalPhalanxTW	100.00	100.00	60.00	0.00	0.00	100.00	100.00	83.42	0.00	0.00
RefrigerationDevices	98.13	97.33	96.80	0.00	0.00	98.13	94.93	90.13	0.00	0.00
ScreenType	99.73	88.00	94.93	1.56	9.38	98.13	97.87	74.67	9.38	6.25
ShapeletSim	93.33	3.33	2.78	3.13	3.13	76.11	32.22	17.22	18.75	14.06
ShapesAll	100.00	100.00	67.17	0.00	0.00	100.00	100.00	54.33	0.00	0.00
SmallKitchenAppliances	98.67	52.00	60.27	0.00	0.00	98.67	98.93	6.67	0.00	0.00
SonyAIBORobotSurface1	73.88	59.24	56.91	21.88	21.88	70.05	65.39	59.40	24.22	25.00
SonyAIBORobotSurface2	63.48	75.13	72.82	25.00	26.56	63.27	64.64	58.76	34.38	34.38
StarLightCurves	100.00	100.00	92.39	0.00	0.00	100.00	100.00	100.00	0.00	0.00
Strawberry	100.00	100.00	94.60	0.00	0.00	100.00	100.00	100.00	0.00	0.00
SwedishLeaf	100.00	98.72	81.76	0.00	0.00	99.36	99.52	79.84	0.00	0.00
Symbols	99.80	100.00	100.00	0.00	0.78	99.80	100.00	96.48	0.00	0.00
SyntheticControl	59.33	53.67	53.67	38.54	39.06	56.33	64.00	66.67	41.67	41.67
ToeSegmentation1	74.56	96.49	82.90	10.94	12.50	74.12	93.86	76.75	12.50	10.94
ToeSegmentation2	99.23	90.00	80.77	0.00	0.00	99.23	98.46	93.08	0.00	0.00
Trace	100.00	72.00	97.00	0.00	0.00	100.00	100.00	88.00	0.00	0.00
TwoLeadECG	99.21	99.74	88.67	0.52	0.39	98.68	99.65	84.02	0.00	0.00
TwoPatterns	100.00	100.00	99.95	1.56	1.56	99.95	99.65	98.93	3.96	3.44
UwaveGestureLibraryAll	100.00	99.97	99.67	0.00	0.00	99.97	99.97	100.00	0.00	0.00
UwaveGestureLibraryX	99.97	99.97	100.00	0.00	0.00	99.97	99.94	99.97	0.00	0.00
UwaveGestureLibraryY	99.97	99.97	100.00	0.00	0.00	99.97	100.00	100.00	0.00	0.00
UwaveGestureLibraryZ	100.00	99.97	100.00	0.00	0.00	100.00	99.97	100.00	0.00	0.00
Wafer	100.00	99.81	100.00	0.00	0.00	100.00	99.94	99.48	0.00	0.00
Wine	100.00	11.11	96.30	0.00	0.00	100.00	100.00	100.00	0.00	0.00
WordSynonyms	100.00	100.00	100.00	0.00	0.00	99.84	99.84	99.84	0.00	0.00
Worms	98.70	100.00	84.42	2.60	3.90	98.70	100.00	25.97	1.30	1.30
WormsTwoClass	97.40	96.10	53.25	2.60	2.60	97.40	100.00	84.42	2.60	2.60
Yoga	100.00	100.00	99.90	0.00	0.00	100.00	100.00	100.00	0.00	0.00



(a) FCN (b) ResNet

Figure 8: Comparison of accuracy for the 2CB detection between FGSM and partial-FGSM (%) for: (a) FCN and (b) ResNet



(a) FCN (b) ResNet

Figure 9: Comparison of accuracy for the 2CB detection between PGD and partial-PGD (%) for: (a) FCN and (b) ResNet

6 Conclusion

This study focuses on adversarial attacks using adversarial examples for time series data classification. The findings show that adversarial examples of time series data that cannot be identified as adversarial examples in time series data classification can be generated using partial perturbations. The perturbation of an entire dataset exhibits clear indicators of data perturbation, which is not encountered when using image data. Therefore, partial-FGSM and partial-PGD methods were proposed, and these methods use only partial perturbations, thus rendering them simpler and more realistic attack methods. Furthermore, the effectiveness of the proposed methods was evaluated experimentally. Datasets for which the attack performance did not sufficiently degrade existed and were negligible, even when the perturbation range was reduced to 1/10. Furthermore, we verified the effectiveness of the 2CB detection method, which detects adversarial examples based on the differences in data characteristics. These results indicate minimal protection. Therefore, the findings of this study show that the threat of partial perturbations was significant in terms of abnormalities and the difficulty in defending against them, particularly in time series data. In the future, defense methods against the partial-FGSM and partial-PGD methods will be studied, and a feature analysis of the datasets for which partial perturbation is effective will be conducted.

Acknowledgements

This study was supported by the JSPS Grant-in-Aid for Scientific Research 23K11340 and the Hiroshima City University Special Research Fund.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems, pp. 1097–1105, 2012, doi: 10.1145/3065386.

[2] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. “Intriguing properties of neural networks,” In ICLR, 2014.

[3] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and Harnessing Adversarial Examples,” In ICLR, 2015.

[4] Samuel Henrique Silva and Peyman Najafirad, “Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey,” arXiv preprint arXiv: 2007.00753, 2020.

[5] X. Yuan, P. He, Q. Zhu, and X. Li, “Adversarial Examples: Attacks and Defenses for Deep Learning,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 9, pp. 2805-2824, Sept. 2019, doi: 10.1109/TNNLS.2018.2886017.

[6] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, Qian Wang, “Recent Advances in Adversarial Training for Adversarial Robustness,” in Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, pp. 4312-4321, 2021, doi: 10.24963/ijcai.2021/591.

[7] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel, “On the (statistical) detection of adversarial examples,” arXiv preprint arXiv:1702.06280, 2017.

[8] N. Carlini and D. Wagner, “Audio Adversarial Examples: Targeted Attacks on Speech-to-Text,” 2018 IEEE Security and Privacy Workshops (SPW), 2018, pp. 1-7, doi: 10.1109/SPW.2018.00009.

[9] Z. Shao, Z. Wu, and M. Huang, “AdvExpander: Generating Natural Language Adversarial Examples by Expanding Text,” in IEEE/ACM Transactions on Audio, Speech, and Language Processing, doi: 10.1109/TASLP.2021.3129339.

[10] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, “Adversarial Attacks on Deep Neural Networks for Time Series Classification,” in 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1-8, doi: 10.1109/IJCNN.2019.8851936, 2019.

[11] F. Karim, S. Majumdar, and H. Darabi, “Adversarial Attacks on Time Series,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 10, pp. 3309-3320, 1, Oct. 2021, doi: 10.1109/TPAMI.2020.2986319.

[12] M. G. Abdu-Aguye, W. Gomaa, Y. Makihara, and Y. Yagi, “Detecting Adversarial Attacks In Time-Series Data,” in ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3092-3096, doi: 10.1109/ICASSP40776.2020.9053311, 2020.

[13] J. Teraoka and K. Tamura, “Detecting Adversarial Examples for Time Series Classification and Its Performance Evaluation,” in Czarnowski I., Howlett R.J., Jain L.C. (eds) Intelligent Decision Technologies. Smart Innovation, Systems and Technologies, vol 238. Springer, Singapore, 2021, doi:10.1007/978-981-16-2765-1_47.

[14] J. Teraoka and K. Tamura, "Adversarial Examples of Time Series Data based on Partial Perturbations," 2022 12th International Congress on Advanced Applied Informatics (IIAI-AAI), Kanazawa, Japan, 2022, pp. 1-6, doi: 10.1109/IIAIAAI55812.2022.00011 (Present paper is the extended journal version of this paper).

[15] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” arXiv preprint arXiv:1605.07277, 2016.

[16] Aleksander Madry, Aleksander Makelov, and Ludwig Schmidt, “Towards Deep Learning Models Resistant to Adversarial Attacks,” in International Conference on Learning Representations, 2018.

[17] Alex Krizhevsky and Geoffrey Hinton, “Learning Multiple Layers of Features from Tiny Images”, 2009.

[18] H. A. Dau et al., “The UCR time series archive,” in IEEE/CAA Journal of Automatica Sinica, vol. 6, no. 6, pp. 1293-1305, November 2019, doi: 10.1109/JAS.2019.1911747.

[19] Z. Wang, W. Yan, and T. Oates, “Time series classification from scratch with deep neural networks: A strong baseline,” 2017 International Joint Conference on Neural Networks (IJCNN), pp. 1578-1585, doi: 10.1109/IJCNN.2017.7966039, 2017.

[20] H. Ismail Fawaz, G. Forestier, J. Weber et al. “Deep learning for time series classification: a

review," Data Mining and Knowledge Discovery, 33, 917–963 (2019), doi: doi.org/10.1007/s10618-019-00619-1.

[21] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick McDaniel, "Technical Report on the CleverHans v2.1.0 Adversarial Examples Library," arXiv preprint arXiv:1610.00768, 2016.