Training Robust Deep Models for Time-Series Domain: Novel Algorithms and Theoretical Analysis

Taha Belkhouja, Yan Yan, Janardhan Rao Doppa

School of EECS, Washington State University
{taha.belkhouja, yan.yan1, jana.doppa}@wsu.edu

Abstract

Despite the success of deep neural networks (DNNs) for real-world applications over time-series data such as mobile health, little is known about how to train robust DNNs for time-series domain due to its unique characteristics compared to images and text data In this paper, we fill this gap by proposing a novel algorithmic framework referred as **RO***bust Training for* **T***ime*-**S***eries (ROTS)* to create robust deep models for time-series classification tasks. Specifically, we formulate a *min-max* optimization problem over the model parameters by explicitly reasoning about the robustness criteria in terms of additive perturbations to time-series inputs measured by the global alignment kernel (GAK) based distance. We also show the generality and advantages of our formulation using the summation structure over time-series alignments by relating both GAK and dynamic time warping (DTW). This problem is an instance of a family of compositional min-max optimization problems, which are challenging and open with unclear theoretical guarantee. We propose a principled stochastic compositional alternating gradient descent ascent (SCAGDA) algorithm for this family of optimization problems. Unlike traditional methods for time-series that require approximate computation of distance measures, SCAGDA approximates the GAK based distance *on-the-fly* using a moving average approach. We theoretically analyze the convergence rate of SCAGDA and provide strong theoretical support for the estimation of GAK based distance. Our experiments on real-world benchmarks demonstrate that ROTS creates more robust deep models when compared to adversarial training using prior methods that rely on data augmentation or new definitions of loss functions. We also demonstrate the importance of GAK for time-series data over the Euclidean distance.

1 Introduction

Predictive analytics over time-series data enables many important real-world applications including mobile health, smart grid management, smart home automation, and finance. In spite of the success of deep neural networks (DNNs) (Wang, Yan, and Oates 2017), very little is known about the robustness of DNNs for time-series data. Recent work on adversarial examples for image (Kolter and Madry 2018) and text data (Wang, Singh, and Li 2019) exposed the brittleness of DNNs and motivated methods to improve their robustness. Therefore, training robust deep models is a necessary requirement when we deploy DNNs over time-series data in critical and high-stakes applications (e.g., mobile health (Belkhouja and Doppa 2020)). Real-world deployment of deep models for time-series data pose several robustness challenges. First, labeled training set for time-series domain tend to be smaller when compared to image and text domains. Consequently, the learned DNNs may not perform well on unseen data drawn from the same distribution. Second, disturbances due to noisy sensor observations or different sampling frequencies can potentially result in poor performance of DNN models. Third, adversarial attacks for malicious purposes to break DNN models can pose security threats.

Prior work on robustness for image data considers training DNNs to be robust to adversarial attacks in a L_p ball and input perturbations. Algorithms to improve robustness of DNNs fall into two broad categories. First, adversarial training using data augmentation (e.g., adversarial examples or input perturbations). Second, optimizing an explicit loss function for robustness criterion (e.g., similar input images should produce similar DNN outputs). Almost all prior methods are designed for images and are based on the L_p norm distance. Since time-series data has unique characteristics (e.g., sparse peaks, fast oscillations) , L_p distance can rarely capture the true similarity between time-series pairs and prior methods are likely to fail on time-series data, as demonstrated by results in Fig. 1. The main question of this paper is: *what are good methodologies to train robust DNNs for time-series domain by handling its unique challenges?*

To answer this question, we propose a novel and principled framework referred as **RO***bust Training for* **T***ime*-**S***eries (ROTS)* to create robust DNNs for time-series data. We employ additive noise variables to simulate perturbations within a small neighborhood around each training example. We incorporate these additive noise variables to formulate a *min-max* optimization problem to reason about the robustness criteria in terms of disturbances to time-series inputs by minimizing the worst-case risk. To capture the special characteristics of time-series signals, we employ the global alighnment kernel (GAK) based distance (Cuturi et al. 2007) to define neighborhood regions for training examples. We show the generality and advantage of our

formulation using the summation structure over time-series alignments by relating both GAK and dynamic time warping (DTW) (Berndt and Clifford 1994).

Unfortunately, the min-max optimization problem with GAK based distance is challenging and may fall in the family of compositional min-max problems due to lack of theoretical guarantees. To efficiently solve this general family of optimization problems, we develop a principled *stochastic compositional alternating gradient descent ascent (SCAGDA)* algorithm by carefully leveraging the underlying structure of this problem. Another key computational challenge is that time-series distance measures including the GAK based distance involve going through all possible alignments between pairs of time-series inputs, which is expensive, e.g., $O(T^2)$ for GAK where T is the length of time-series signal. As a consequence, the computational cost grows significantly for iterative optimization algorithms where we need to repeatedly compute the distance between time-series signals. SCAGDA randomly samples a constant number of alignments at each iteration to approximate the GAK based distance *on-the-fly* using a moving average approach and reduces the computational-complexity to $O(T)$.

Our theoretical analysis shows that SCAGDA achieves an ϵ-primal gap in $O(1/\epsilon^2)$ iterations for a family of nonconvex-nonconcave compositional min-max optimization problems. *To the best of our knowledge, this is the first convergence rate established for nonconvex-nonconcave min-max problems with a compositional structure*. We also prove that SCAGDA approximates GAK based distance during the execution of algorithm: the approximation error converges as the primal gap converges. This result provides strong theoretical support for our algorithm design that leverages the structure of the ROTS formulation. Our experiments on real-world datasets show that DNNs learned using ROTS are more robust than prior methods; and GAK based distance has a more appropriate bias for time-series than L_2.

Contributions. The key contribution of this paper is the development and evaluation of the ROTS algorithmic framework to train robust deep models for time-series domain.

- Novel SCAGDA algorithm for a family of nonconvex-nonconcave compositional min-max problems that covers ROTS as a special case with GAK based distance. SCAGDA approximates the GAK distance by randomly sampling a constant number of time-series alignments.
- Theoretical analysis of SCAGDA shows that the iteration complexity to achieve ϵ-primal gap and the ϵ-approximation error of GAK distance is $O(1/\epsilon^2)$.
- Comprehensive experimental evaluation of ROTS on diverse real-world benchmark datasets and comparison with state-of-the-art baselines. The source code of ROTS algorithms are available at ROTS GitHub

2 Problem Setup and Formulation

We consider the problem of learning *robust* DNN classifiers over time-series data. We are given a training set of n input-output pairs $\{(x_i, y_i)\}_{i=1}^n$. Each input $x_i \in \mathcal{X}$ is a time-series signal, where $\mathcal{X} \subseteq \mathbb{R}^{C\times T}$ with C denoting the number of channels and T being the window-size of the signal; and $y_i \in \mathcal{Y}$ is the associated ground-truth label, where $\mathcal{Y} \in \{1, \cdots, \mathcal{C}\}$ is a set of $\mathcal{C}$ discrete class labels. For example, in a health monitoring application using physiological sensors for patients diagnosed with cardiac arrhythmia, we use the measurements from wearable devices to predict the likelihood of a cardiac failure. Traditional empirical risk minimization learns a DNN classifier $f : \mathcal{X} \times \Theta \to \mathcal{Y}$ with weights $w \in \Theta$ that maps time-series inputs to classification labels for a hypothesis space Θ and a loss function ℓ:

$$\min_{w\in\Theta} \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i, w), y_i).$$

Training for robustness. We would like the learned classifier $f(x, w)$ to be robust to disturbances in time-series inputs due to noisy observations or adversarial attacks. For example, a failure in the prediction task for the above health monitoring application due to such disturbances can cause injury to the patient without the system notifying the needed assistance. Therefore, we want the trained DNN classifier to be invariant to such disturbances. Mathematically, for an appropriate distance function $d(x, x')$ over time-series inputs x and x', we want the classifier f to predict the same classification label as x for all inputs x' such that $d(x, x') < \varepsilon$, where ε stands for the bound on allowed disturbance to input x. This goal can be achieved by reasoning about the worst-case empirical risk over possible perturbations $a_i \in \mathbb{R}^{C\times T}$ of x_i such that $d(x_i, x_i + a_i) \leq \varepsilon$. The resulting *min-max* optimization problem is given below.

$$\begin{aligned} \min_{w\in\Theta} \quad & \frac{1}{n}\sum_{i=1}^{n} \max_{a_i} \ell(f(x_i + a_i, w), y_i) \\ \text{s.t.} \quad & d(x_i, x_i + a_i) \leq \varepsilon \end{aligned} \tag{1}$$

In practice, instead of solving the above hard constrained problem, one can solve an equivalent soft constrained problem using regularization as follows

$$\min_{w\in\Theta} \max_{a_i} \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i + a_i, w), y_i) - \lambda d(x_i, x_i + a_i) \tag{2}$$

There is a natural interpretation of this optimization problem. The *inner maximization* problem serves the role of an attacker whose goal is to find adversarial examples that achieves the largest loss. The *outer minimization* problem serves the role of a defender whose goal is to find the parameters of the deep model w by minimizing the adversarial loss from the inner attack problem. This formulation is applicable to all types of data by selecting an appropriate distance function d. For example, L_p-norm distance is usually used in the image domain (Kolter and Madry 2018).

Typical stochastic approaches to solving the above adversarial training problem include alternating stochastic optimization (Junchi Y. 2020) and stochastic gradient descent ascent (GDA) (Lin, Jin, and Jordan 2020; Yan et al. 2020). The alternating method first fixes w and solves the inner maximization approximately to get each a_i (e.g., using

stochastic gradient descent). Next, a_i is fixed and the outer minimization is solved over w. These two steps are performed alternatively until convergence. The GDA method computes the gradient of w and a_i simultaneously at each iteration, and then use these gradients to update w and a_i. Both methods require the ability to compute the unbiased estimation of the gradients w.r.t. w and a_i. When d is decomposable, e.g., L_p-norm, then its stochastic gradient can be easily computed. However, in time series domain, commonly used distance measures may not be decomposable, so its stochastic gradients are not accessible. Consequently, one has to calculate the exact gradient of $d(x_i, x_i + a_i)$ w.r.t, a_i. We will investigate this key challenge in Section 3.2.

3 ROTS Algorithmic Framework

In this section, we describe the technical details of our proposed ROTS framework to train robust DNN classifiers for time-series domain. First, we instantiate the min-max formulation with GAK based distance as it appropriately captures the similarity between time-series signals. Second, we provide an efficient algorithm to solve the GAK-based formulation to learn parameters of DNN classifers.

3.1 Distance Measure for Time-Series

Unlike images and text, time-series data exhibits unique characteristics such as sparse peaks, fast oscillations, and frequency/time shifting which are important for pattern matching and data classification. Hence, measures such as Euclidean distance that do not account for these characteristics usually fail in recognizing the similarities between time-series signals. To address this challenge, elastic measures have been introduced for pattern-matching tasks for time-series domain (Cuturi et al. 2007), where one time-step of a signal can be associated with many time-steps of another signal to compute the similarity measure.

Time-series alignment. Given two time series x=$(x_1, \cdots, x_{T_1})$ and x'=$(x'_1, \cdots, x'_{T_2})$ for $T_1, T_2 \in \mathbb{N}_+$, the alignment $\pi = (\pi_1, \pi_2)$ is defined as a pair of increasing integral vectors of length $r \leq T_1 + T_2 - 1$ such that $1 = \pi_1(1) \leq \cdots \leq \pi_1(r) = T_1$ and $1 = \pi_2(1) \leq \cdots \leq \pi_2(r) = T_2$ with unitary increments and without simultaneous repetitions, which presents the coordinates of x and x'. This alignment defines the one-to-many alignment between x and x' to measure their similarity. Using a candidate alignment π, we can compute their similarity as follows:

$$d_\pi(x, x') = \sum_{i=1}^{|\pi|} \mathrm{dist}(x_{\pi_1(i)}, x'_{\pi_2(i)}) \tag{3}$$

where $|\pi|$=r denotes the length of alignment and $\mathrm{dist}(\cdot,\cdot)$ in the above equation is the Minkowski distance:

$$\mathrm{dist}(x_{\pi_1(i)}, x'_{\pi_2(i)}) = \|x_{\pi_1(i)} - x'_{\pi_2(i)}\|_p,\ p \in \{1, \cdots, \infty\}$$

Global alignment kernel (GAK) based distance. The concept of alignment allows us to take into consideration the intrinsic properties of time-series signals, such as frequency shifts, to compute their similarity. There are some well-known approaches to define distance metrics using time-series alignment. For example, dynamic time warping (DTW) (Berndt and Clifford 1994) selects the alignment with the minimum distance:

$$D_{\mathrm{DTW}}(x, x') = \min_{\pi \in \mathcal{A}} d_\pi(x, x'),$$

where $\mathcal{A}$ denotes the set of all possible alignments.

While DTW only takes into account one candidate alignment, global alignment kernel (GAK) (Cuturi et al. 2007) takes all possible alignments into consideration:

$$k_{\mathrm{GAK}}(x, x') = \sum_{\pi \in \mathcal{A}} \exp\Big(-\frac{d_\pi(x, x')}{\nu}\Big) \tag{4}$$

where ν is a hyper-parameter and $d_\pi(\cdot,\cdot)$ is defined in Equation (3). In practice, to handle the diagonally dominance issue (Cuturi et al. 2007; Wu et al. 2018; Cuturi 2011), $D_{\mathrm{GAK}} := -\nu \log(k_{\mathrm{GAK}})$ is typically used as a distance measure for a pair of time-series signals. GAK enjoys several advantages over DTW (Cuturi 2011): (i) differentiable, (ii) positive definite, (iii) coherent measure over all possible alignments. Therefore, k_{GAK} (or D_{GAK}) is a better fit to train robust DNNs for the time-series domain.

On the other hand, GAK can also be a more *general* measure than DTW due to its summation structure, as $\lim_{\nu \to 0} D_{\mathrm{GAK}}(x, x') = D_{\mathrm{DTW}}(x, x')$, i.e., arbitrarily close to DTW by changing ν. The following proposition shows the tight approximation of the soft minimum of GAK to the hard minimum of DTW (proof and details in Appendix C.6).

Proposition 1. *For a time-series pair (x, x'), we have:*

$$0 \leq D_{DTW}(x, x') - D_{\mathrm{GAK}}(x, x') \leq \nu \log(|\mathcal{A}|).$$

As shown, D_{GAK} converges to D_{DTW} in $\nu \log(|\mathcal{A}|)$ as ν decreases. Due to the above advantages and the approximation ability of D_{GAK} to D_{DTW}, we consider the more *general* k_{GAK} and D_{GAK} in our ROTS method.

3.2 SCAGDA Optimization Algorithm

By plugging k_{GAK} from Equation (4) to replace d into the min-max formulation in Equation (2), we reach the following objective function of our ROTS framework:

$$\min_{w \in \Theta} \max_{a_i} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i + a_i, w), y_i) + \lambda \overbrace{\log\big(k_{\mathrm{GAK}}(x_i, x_i + a_i)\big)}^{= d(x_i, x_i + a_i)} \tag{5}$$

where ν outside $\log$ in D_{GAK} can be merged into λ. The above problem is decomposable over individual training examples (i.e., index i), so we can compute stochastic gradients by randomly sampling a batch of data and employ stochastic gradient descent ascent (SGDA) (Lin, Jin, and Jordan 2020; Yan et al. 2020), a family of stochastic algorithms for solving min-max problems.

Key challenge. The second term $\log(k_{\mathrm{GAK}}(x_i, x_i + a_i))$ has a *compositional* structure due to the outer $\log$ function. By chain rule, its gradient w.r.t. the dual variable a_i is

$$\nabla_{a_i} \log(k_{\mathrm{GAK}}(x_i, x_i + a_i)) = \frac{\nabla_{a_i} k_{\mathrm{GAK}}(x_i, x_i + a_i)}{k_{\mathrm{GAK}}(x_i, x_i + a_i)}$$

Algorithm 1: SCAGDA (Stochastic Compositional Alternating Gradient Descent Ascent)

1: Initialize w_0, a_i^0 for $i = 1, \cdots, n$ and $\omega_i^0 = 0$ for $i = 1, \cdots, n$, step sizes $\{\eta_k\}_{k=1}^K$ and $\{\gamma_k\}_{k=1}^K$.
2: **for** $k = 0, \cdots, K-1$ **do**
3: Randomly sample an index i_1 to compute stochastic gradient $\nabla_w f_{i_1}(w_k, a_{i_1}^k)$
4: Set: $w_{k+1} = w_k - \eta_k \nabla_w f_i(w_k, a_{i_1}^k)$
5: Randomly sample an index i_2 to compute stochastic gradient $\nabla_a f_{i_2}(w_{k+1}, a_{i_2}^k)$
6: Randomly sample two independent indices j_1, j_2 of h_{i_2} to compute $h_{i_2,j_1}(a_{i_2}^k)$ and $\nabla h_{i_2,j_2}(a_{i_2}^k)$
7: Set:

$$\omega_i^{k+1} = \begin{cases} \omega_i^k & \text{for } i \neq i_2 \\ (1-\beta)\omega_{i_2}^k + \beta h_{i_2,j_1}(a_{i_2}^k) & \text{for } i = i_2 \end{cases}$$

8: Set: $a_{i_2}^{k+1} = a_{i_2}^k + \gamma_k(\nabla_a f_{i_2}(w_{k+1}, a_{i_2}^k) - \nabla h_{j2}(a_{i_2}^k)^\top \nabla g(\omega_{i_2}^{k+1}))$
9: **end for**
10: **return** final solution w_K

where one has to go through all possible alignments to compute k_{GAK} and $\nabla_{a_i} k_{\mathrm{GAK}}$ (see Equation (4)) and there is no unbiased estimation (i.e., stochastic gradients) for it.

Consequently, at each iteration, SGDA has to compute the exact value of $k_{\mathrm{GAK}}(x_i, x_i + a_i)$ and $\nabla_{a_i} k_{\mathrm{GAK}}(x_i, x_i + a_i)$ according to the chain rule, which leads to an additional time-complexity of $O(CT^2)$ per SGDA iteration, where C and T denote the number of channels and window-size respectively. This computational bottleneck will lead to extremely slow training algorithm when C and/or T is large, which is the case in many real-world applications.

One candidate approach to alleviate the computational challenge due to $\log(k_{\mathrm{GAK}})$ part of the objective is to make use of the inner summation structure of k_{GAK}. Since k_{GAK} involves a summation over all alignments, as shown in (4), we can use only a *subset* of alignments for estimating the full summation. This procedure will give an unbiased estimation of k_{GAK}, but the outer logarithmic function makes it a *biased* estimation for $\nabla_{a_i} \log(k_{\mathrm{GAK}}(x_i, x_i + a_i))$. However, such biased estimation violates the assumption in SGDA studies, so their theoretical analysis cannot hold.

There is another line of research investigating stochastic compositional gradient methods for minimization problems with compositional structure (Wang, Fang, and Liu 2017; Chen, Sun, and Yin 2020). However, *min-max* optimization with *compositional* structure, including our case shown in Equation (5), is not studied yet. It is unclear whether these techniques and analysis hold for min-max problems.

SCAGDA algorithm. We propose a novel *stochastic compositional alternating gradient descent ascent (SCAGDA)* algorithm to solve a family of nonconvex-nonconcave min-max compositional problems, which include ROTS (Equation (5)) as a special case. We summarize SCAGDA in Algorithm 1. Specifically, we consider solving the following family of problems:

$$\min_w \max_{a_i} \frac{1}{n} \sum_{i=1}^n \phi_i(w, a_i) \tag{6}$$

where $\phi_i(w, a_i) := f_i(w, a_i) - g(\frac{1}{m}\sum_{j=1}^m h_{i,j}(a_i))$.

Mapping Problem (6) to ROTS (5). As mentioned above, ROTS for time-series in Equation (5) is a special case of Problem (6) as shown below. The variables $\phi_i(w, a_i), f_i, g, h_{i,j}$ in Problem (6) can be instantiated by the following mappings:

- f_i in ϕ_i of (6) $\Rightarrow$ the loss ℓ on the i-th data in (5)
- $-g(\cdot)$ in ϕ_i of (6) $\Rightarrow \lambda \log(\cdot)$ in (5)
- $\frac{1}{m}\sum_{j=1}^m h_{i,j}(a_i)$ in ϕ_i of (6) $\Rightarrow k_{\mathrm{GAK}}(x_i, x_i + a_i) = \sum_{\pi \in \mathcal{A}} \exp(-d_\pi(x_i, x_i + a_i)/\nu)$ in (5), where m and j corresponds to the total number of alignment paths $|\mathcal{A}|$ and the index of alignment path, respectively. Note that the summation form of k_{GAK} can be easily converted to an average form due to $\log(x) = \log(x/m) + \log(m)$.

Algorithmic analysis of SCAGDA. To introduce and analyze Algorithm 1 for solving Problem (6), we first introduce some notations. Denote $P(w) := \max_{a_i} \frac{1}{n}\sum_{i=1}^n \phi_i(w, a_i)$ as the *primal function* of the above min-max optimization problem, where we are interested in analyzing the convergence of the *primal gap* after the K-th iteration:

$$P(w_K) - \min_w P(w).$$

Let $a := (a_1, a_2, ..., a_n) \in \mathbb{R}^{C \times T \times n}$ be the concatenation of a_i for $i = 1, \cdots, n$. We also use the following notations to improve the technical exposition and ease of readability.

$$\begin{aligned} \phi(w, a) :=& \frac{1}{n}\sum_{i=1}^n \phi_i(w, a_i), \\ h_i(a_i) :=& \frac{1}{m}\sum_{j=1}^m h_{i,j}(a_i), \\ h(a) :=& \frac{1}{n}(h_1(a_1), \cdots, h_n(a_n)), \end{aligned}$$

where the last term $h(a)$ is the concatenation of all h_i for $i = 1, \cdots, n$. In Appendix C.7, we provide the details of how (6) is specifically viewed as a stochastic problem.

As mentioned above while discussing the key challenge of the compositional structure in ROTS (5), conventional SGDA methods for Problem (6) require us to compute the full gradient of the compositional part $g(h_i(a_i))$, i.e., $\nabla h_i(a_i)^\top \nabla g(h_i(a_i))$, which involves *all* alignments in the case of ROTS.

In contrast, SCAGDA only samples a constant number of $h_{i,j}(a_i)$ over j (i.e., over a subset of alignments for ROTS) and $\nabla h_{i,j}(a_i)$ (**Line 6**). Subsequently, SCAGDA employs a simple iterative *moving average* (MA) approach to accumulate $h_{i,j}(a_i)$ into ω_i^{k+1} at iteration k for estimating $h_i(a_i)$ (**Line 7**). The key idea behind moving average method is to control the variance of the estimation for $h_i(a_i^{k+1})$ using a weighted average from the previous estimate $h_i(a_i^k)$. Even though $\nabla g(\omega_i^{k+1})$ is a biased estimation

Algorithm 2: ROTS Instantiation of SCAGDA

Input: A training set $\{(x, y) \in \mathcal{X} \times \mathcal{Y}\}^{n_{\text{train}}}$; mini-batch size s, deep neural network $f(w, x, y)$; learning rates η_k and γ_k, loss function $l(\cdot)$, distance function $D(\cdot, \cdot)$.
Output: Classifier weights $w \in \Theta$

1: Randomly initialize weights of the DNN classifier: $w_0 \in \Theta$
// vector of worst-case perturbations, one for each time-series
2: Initialize $a_0 = 0$ and $\omega_0 = 0$
// Multiple iterations of SCAGDA
3: **for** $k = 0, \cdots, K-1$ **do**
4: Randomly sample a mini-batch of data samples indexed by $\mathcal{I}_k$ s.t. $|\mathcal{I}_k| = s$
5: Compute the stochastic gradient w.r.t. w_k
$\mathcal{G}_{w,k} = \frac{1}{s}\sum_{i\in\mathcal{I}_k} \nabla_w l(f(w_k, x_i + a_i^k, y_i)$
6: Perform stochastic gradient descent on w_k
$w_{k+1} = w_k - \eta_k \mathcal{G}_{w,k}$
7: Randomly sample a mini-batch of alignments indexed by $\widehat{\mathcal{A}}_i^k$ for each data index $i \in \mathcal{I}_k$
8: Moving average for $i \in \mathcal{I}_k$
$\omega_i^{k+1} = (1-\beta)\omega_i^k + \beta \sum_{\pi\in\widehat{\mathcal{A}}_i^k} \exp(-d_\pi(x_i, x_i + a_i^k)/\nu)$
9: $\omega_i^{k+1} = \omega_i^k$ for $i \notin \mathcal{I}_k$.
10: Compute $\mathcal{G}_{a,k,i} = \nabla_a \ell(f(w_{k+1}, x_i + a_i^k, y_i) - \sum_{\pi\in\widehat{\mathcal{A}}_i^k} \exp(-d_\pi(x_i, x_i+a_i^k)/\nu)\cdot\nabla_a d_\pi(x_i, x_i+a_i^k)\cdot \frac{\lambda}{\omega_i^{k+1}\nu}$ for $i \in \mathcal{I}_k$
11: Perform stochastic gradient ascent over perturbations
$a_i^{k+1} = a_i^k + \gamma_k \mathcal{G}_{a,k,i}$
12: **end for**
13: **return** weights of the learned DNN classifier, w_K

of $\nabla g(h_i(a_i^k))$, we can still use smoothness condition (introduced in Section 4 later) and bound the approximation error $\mathbb{E}[\|\omega_i^k - h_i(a_i^{k-1})\|^2]$ where $\omega_k := (\omega_1^k, ..., \omega_n^k)$ is the concatenation of all $\{\omega_i^k\}_{i=1}^n$ at iteration k, as shown in Theorem 2.

Therefore, instantiation of SCAGDA for ROTS does not require us to perform computation over *all* alignments contained in $h_i(a_i)$ for each time-series training sample, which leads to a more efficient algorithm with high scalability on large datasets. As shown in **Line 3** and **5**, SCAGDA updates the primal variable w and dual variable a in an *alternating* scheme, which means that w_{k+1} is updated based on a_k, while a_{k+1} is updated based on w_{k+1}. This is different from SGDA, which updates a_k based on w_k instead. We instantiate SCAGDA for the proposed ROTS framework as shown in Algorithm 2. The primal variable update is provided in Line 6, and the dual variable update is provided in Line 11. In particular, Line 8 and 8 correspond to the moving average step for estimating k_{GAK} using randomly sampled alignment subset $\widehat{\mathcal{A}}_i^k$ for the i-th time-series training example.

In the next section, we show that our algorithm can converge to primal gap $P(w_K) - \min_w P(w) \leq \epsilon$ with iteration complexity $O(1/\epsilon^2)$, where ϵ is a pre-defined threshold. *To the best of our knowledge, this is the first optimization algorithm and convergence analysis for the famaily of compositional min-max optimization problems shown in (6).*

4 Theoretical Analysis

In this section, we present novel theoretical convergence analysis for SCAGDA algorithm. As mentioned in the previous section, for the problem (6), existing theoretical analysis of SGDA (Lin, Jin, and Jordan 2020; Yan et al. 2020), stochastic alternating gradient descent ascent (SAGDA) (Junchi Y. 2020) require us to compute exact gradient of $g(h_i(a_i))$ at each iteration. On the other hand, it is unclear if stochastic compositional alternating gradient algorithms for minimization problems (Wang, Fang, and Liu 2017; Chen, Sun, and Yin 2020) can handle the complex min-max case.

Summary of results. We answer the following question: *can we establish convergence guarantee of our SCAGDA algorithm for nonconvex-nonconcave compositional min-max optimization problems?*

Theorem 1 proves that SCAGDA shown in Algorithm 1 converges to an ϵ-primal gap in $O(\frac{1}{\epsilon^2})$ iterations. Theorem 2 demonstrates the efficacy of the moving average strategy to approximate GAK based distance: the approximation error $\|\omega_i^K - h_i(a_i^{K-1}))\|^2$ is also bounded by ϵ in expectation when the ϵ-primal gap is achieved.

4.1 Main Results

The following commonly used assumptions are used in our analysis. Due to the space limit, definitions, proofs, and detailed constant dependencies can be found in Appendix C.

Assumption 1. *Suppose $\mu, L, C_g, C_h, L_g, L_h \geq 0$.*
(i) $\phi(w, a)$ satisfies two side μ-PL condition:

$$\|\nabla_w \phi(w,a)\|^2 \geq 2\mu(\phi(w,a) - \min_{w'} f(w,a)),$$
$$\|\nabla_a \phi(w,a)\|^2 \geq 2\mu(\max_{a'} f(w,a) - \phi(w,a)).$$

(ii) $\phi(w, a)$ is L-smooth in w for fixed a.
(iii) $\phi(w, a)$ is L-smooth in a_i for fixed w.
(iv) g (resp. h) is C_g (resp. C_h)-Lipschitz continuous.
(v) g (resp. h) is L_g (resp. L_h)-smooth.
(vi) $\exists\ \sigma > 0$ s.t. $\mathbb{E}[\|\nabla_w \phi_i(w, a_i) - \nabla_w \phi(w, a)\|^2] \leq \sigma^2$, $\mathbb{E}[\|\nabla_a f_i(w, a_i) - \nabla_a f_i(w, a_i)\|^2] \leq \sigma^2$, $\mathbb{E}[\|h_{i,j}(a_i) - h(a)\|^2] \leq \sigma^2$, and $\mathbb{E}[\|\nabla h_{i,j}(a_i) - \nabla h(a)\|^2] \leq \sigma^2$

We present our main results for SCAGDA below.

Theorem 1. *Suppose Assumption 1 holds. In Algorithm 1, set $\eta_k = \eta = O(1/\epsilon^2)$, $\gamma_k = \gamma = O(1/L^2)$ and $\beta = \sqrt{18\mu\eta}$. After running Algorithm 1 for K iterations where $K = \tilde{O}(1/\epsilon^2)$ ($\tilde{O}$ hides logarithmic factor), we have*

$$\mathbb{E}[P(w_K) - P^*] + \frac{1}{8}\mathbb{E}[P(w_K) - \phi(w_K, a_K)]$$
$$+ \Big(\frac{4C_h^4 L_g^4 \eta_K}{\mu^5}\Big)^{1/2} \mathbb{E}[\|\omega_K - h(a_{K-1})\|^2] \leq \epsilon$$

Remark 1. The above theorem gives us two critical observations of the behavior of SCAGDA. **(1)** After running K

iterations of SCAGDA, the primal gap $P(w_{K+1}) - P^*$ converges to ϵ in expectation, since all terms in the left hand side of the inequality are non-negative. This result shows that SCAGDA is able to effectively solve the compositional min-max optimization problem shown in Equation (6). **(2)** The required iteration complexity of SCAGDA is $O(1/\epsilon^2)$. To put this result in perspective, we compare it with related theoretical results. The rate for nonconvex-nonconcave min-max problem without compositional structure is shown to be $O(1/\epsilon)$ (Junchi Y. 2020). However, this improvement requires unbiased estimation (or exact value) of the gradient and computing the exact $g(h_i(a_i))$ at each iteration. Our iteration complexity is in the same order of that for (Chen, Sun, and Yin 2020), whose convergence result is $O(1/\epsilon^2)$ for nonconvex compositional *minimization* problems instead of *min-max* ones. The difference is that their convergence metric is the average squared norm of gradients, while ours is for the primal gap. Importantly, *this is the first result on convergence rate for stochastic compositional min-max problems*.

Theorem 2. *After $K = \tilde{O}(1/\epsilon^2)$ iterations of Algorithm 1, we have: $\mathbb{E}[\|\omega_i^K - h_i(a_i^{K-1})\|^2] \leq O(\epsilon)$.*

Remark 2. The above result shows that as SCAGDA algorithm is executed, the approximation error of $\|\omega_i^K - h_i(a_i^{K-1})\|^2$ converges to ϵ in the expectation as it is achieving the ϵ-primal gap. For the condition numbers, we always have $L \geq \mu$. In practice, we usually set the accuracy level ϵ to a very small value, so the condition $\epsilon \leq O(L^3/\mu^2)$ will generally hold. This result provides strong theoretical support that if we apply SCAGDA to optimize our ROTS problem in (5), it is able to approximate k_{GAK} on-the-fly, where we only need a constant number of alignments, rather than *all* possible alignments for computing k_{GAK} in each iteration of SCAGDA. When we have ϵ-primal gap, we also achieve ϵ-accurate estimation of k_{GAK} at the same time.

5 Related Work

Prior work on robustness of DNNs is mostly focused on image/text domains; and can be classified into two categories.

Adversarial training employs augmented data such as adversarial examples (Kolter and Madry 2018; Wang, Singh, and Li 2019) and input perturbations. Methods to create adversarial examples include general attacks such as Carlini & Wagner attack (Carlini and Wagner 2017), boundary attack (Brendel, Rauber, and Bethge 2018), and universal attacks (Moosavi-Dezfooli et al. 2017). Recent work regularizes adversarial example generation methods to obey intrinsic properties of images (Laidlaw and Feizi 2019; Xiao et al. 2018; Hosseini et al. 2017). There are also specific adversarial methods for NLP domain (Samanta and Mehta 2017; Gao et al. 2018). There is little to no prior work on adversarial techniques for time-series domain. Fawaz et al. (Fawaz et al. 2019) employed the standard Fast Gradient Sign method (Kurakin, Goodfellow, and Bengio 2017) to create adversarial noise for time-series. Network distillation was also employed to train a student model for creating adversarial attacks (Fazle, Somshubra, and Houshang 2020). This method is severely limited: it can generate adversarial examples for only a small number of target labels and cannot guarantee generation of adversarial example for every input.

Training via explicit loss function employ an explicit loss function to capture the robustness criteria and optimize it. Stability training (Zheng et al. 2016a; Li et al. 2019) for images is based on the criteria that similar inputs should produce similar DNN outputs. Adversarial training can be interpreted as min-max optimization, where a hand-designed optimizer such as projected gradient descent is employed to (approximately) solve inner maximization. (Xiong and Hsieh 2020) train a neural network to guide the optimizer. Since characteristics of time-series (e.g., fast-pace oscillations, sharp peaks) are different from images/text, L_p distance based methods are not suitable for time-series domain.

In summary, there is no prior work to train robust DNNs for time-series domain in a principled manner. This paper precisely fills this important gap in our scientific knowledge.

6 Experiments and Results

We present experimental evaluation of ROTS on real-world time-series benchmarks and compare with prior methods.

6.1 Experimental Setup

Datasets. We employed diverse univariate and multi-variate time-series benchmark datasets from the UCR repository (Bagnall et al. 2020). Table 1 describes the details of representative datasets for which we show the results (due to space limits) noting that our overall findings were similar on other datasets from the UCR repo. We employ the standard training/validation/testing splits for these datasets.

Algorithmic setup and baselines. We employ a 1D-CNN architecture (Bai, Kolter, and Koltun 2018) as the deep model for our evaluation. The details of the neural architecture are provided in the Appendix. We ran ROTS algorithm shown in Appendix for a maximum of 500 iterations to train robust models. To estimate GAK distance within ROTS, we employed 15 percent of the total alignments noting that larger sample sizes didn't improve the optimization accuracy and increased the training time. We also employ adversarial training to create models using baseline attacks that are not specific to image domain for comparison: Fast Gradient Sign method (FGS) (Kurakin, Goodfellow, and Bengio 2017) that was used by Fawaz et al. (2019) and Projected Gradient De-

Name	Classes	Input Size ($C \times T$)
ECG200	2	1×97
BME	3	1×129
ECG5000	5	1×141
MoteStrain	2	1×85
SyntheticControl	6	1×61
RacketSports	4	6×30
Epilepsy	4	3×206
ERing	6	4×65
FingerMovements	2	28×50

Table 1: Description of different datasets.

scent (PGD)(Madry et al. 2017). We also compare ROTS against stability training (STN) (Zheng et al. 2016b).

Evaluation metrics. We evaluate the robustness of created models using different attack strategies on the testing data. The prediction accuracy of each model (via ground-truth labels of time-series) is used as the metric. To ensure robustness, DNN models should be least sensitive to different types of perturbations over original time-series signals. We measure the accuracy of each DNN model against: *1) Adversarial noise* is introduced by FGS and PGD baseline attacks; and *2) Gaussian noise* $\sim \mathcal{N}(0, \Sigma)$ that may naturally occur to perturb time-series. The covariance matrix Σ diagonal elements (i.e., variances) are all equal to σ. DNNs are considered robust if they are successful in maintaining their accuracy performance against such noises.

6.2 Results and Discussion

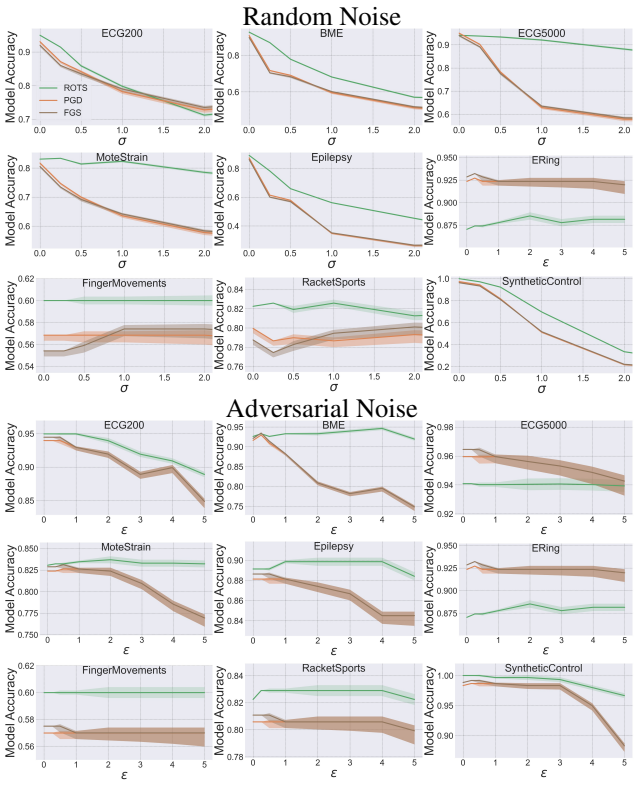

Figure 1: Comparison of ROTS algorithm vs. adversarial training algorithm using baseline FGS and PGD attacks.

ROTS vs. adversarial training. One of our key hypothesis is that Euclidean distance-based perturbations do not capture the appropriate notion of invariance for time-series domain to improve the robustness of the learned model. We show that using baseline attacks to create augmented data for adversarial training does not create robust models. From Figure 1, we can observe that models from our ROTS algorithm achieve significantly higher accuracy than the baselines. For example, on MoteStrain dataset, ROTS has a steady performance against both types of noises, unlike the baselines. On

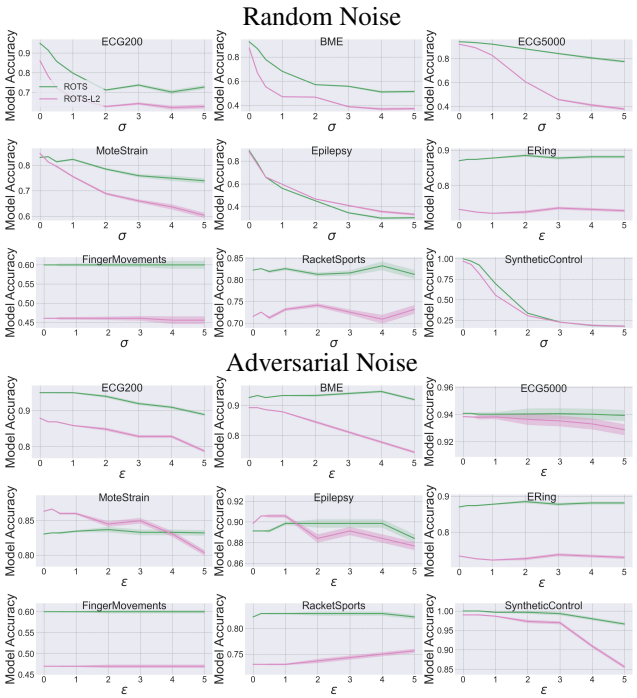

Figure 2: Comparison of ROTS algorithm using GAK distance (k_{GAK}) vs. ROTS using Euclidean distance (L_2).

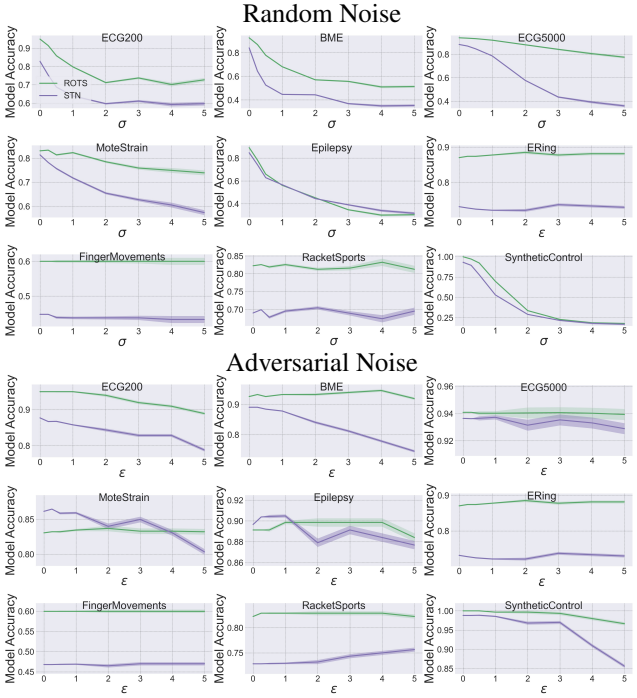

Figure 3: Comparison of ROTS vs. stability training (STN).

the other datasets, we can clearly observe that in most cases, ROTS outperforms the baselines. We conclude that adversarial training using prior methods and attack strategies is not as effective as our ROTS method, where we perform explicit primal-dual optimization to create robust models.

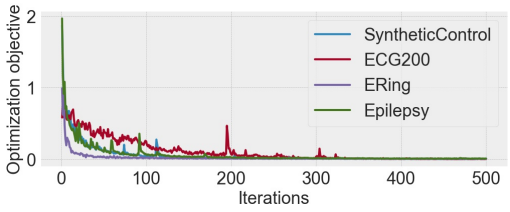

Figure 4: Empirical convergence of ROTS algorithm.

ROTS vs. ROTS with L2 distance. We want to demonstrate that choosing the right distance metric to compute similarity between time-series signals is critical to create robust models. Therefore, we compare models created by ROTS by using two different distance metrics: 1) The standard Euclidean distance $\|\cdot\|_2$ used in image domains and prior work; and 2) Using the GAK distance D_{GAK}. From Figure 2, we can clearly see that the GAK distance is able to explore the time-seris input space better to improve robustness. The Euclidean distance either performs significantly worse than GAK (e.g., on ECG5000, ERing, and RacketSports datasets) or performs comparably to GAK (e.g., on SyntheticControl or Epilepsy datasets). This experiment concludes that GAK is a suitable distance metric for time-series domain.

ROTS vs. stability training. Unlike adversarial training, stability training (STN) employs the below loss function (Zheng et al. 2016b) to introduce stability to the deep model.

$$Loss_{STN} = L_0(x, \theta) + 0.01 \times L_{stability}(x, x', \theta)$$

where x is the original input, x' is a perturbed version of x using additive Gaussian noise $\sim \mathcal{N}(0, 0.04^2)$, L_0 is the cross-entropy loss, and $L_{stability}$ relies on KL-divergence. We experimentally demonstrate that ROTS formulation is more suitable than STN for creating robust DNNs for timeserirs domain. Figure 3 shows a comparison between DNNs trained using STN and ROTS. We observe that for most datasets, ROTS creates significantly more robust DNNs when compared to STN for both types of perturbations. ROTS algorithm is specifically designed for time-series domain by making appropriate design choices, whereas STN is designed for image domain. Hence, ROTS allows us to create more robust DNNs for time-series domain.

Empirical convergence. We demonstrate the efficiency of ROTS algorithm by observing the empirical rate of convergence. Figure 4 shows the optimization objective over iterations on some representative datasets noting that we observe similar patterns on other datasets. We can observe that ROTS converges roughly before 150 iterations for most datasets. Figure 5 shows the accuracy gap results and the computational runtime when comparing ROTS with sampled alignments and original GAK (i.e., all alignment paths). The results clearly match with our theoretical analysis that accuracy gap decreases over training iterations leading to convergence. We conclude from these results that ROTS converges quickly in practice and supports our theoretical analysis.

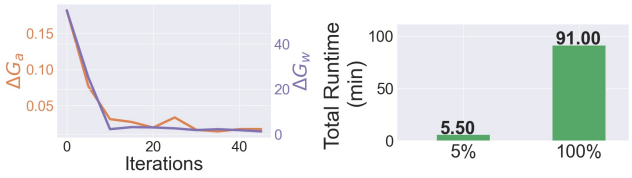

Figure 5: The accuracy gap in the gradients over weights ΔG_W and over perturbations ΔG_a using 5% of alignments and GAK using all alignments for ROTS training on ERing (Left) and the comparison of the computational runtime between both settings of ROTS (Right).

7 Conclusions

We introduced the ROTS algorithm to train robust deep neural networks (DNNs) for time-series domain. The training problem was formulated as a min-max optimization problem to reason about the worst-case risk in a small neighborhood defined by the global alignment kernel (GAK) based distance. Our proposed stochastic compositional alternating gradient descent and ascent (SCAGDA) algorithm carefully leverages the structure of the optimization problem to solve it efficiently. Our theoretical and empirical analysis showed that ROTS and SCAGDA are effective in creating more robust DNNs over prior methods and GAK based distance is better suited for time-series over the Euclidean distance.

Acknowledgements. This research is supported in part by the AgAID AI Institute for Agriculture Decision Support, supported by the National Science Foundation and United States Department of Agriculture - National Institute of Food and Agriculture award #2021-67021-35344.

References

Bagnall, A.; Lines, J.; Vickers, W.; and Keogh, E. 2020. The UEA & UCR Time Series Classification Rep. www.timeseriesclassification.com.

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

Belkhouja, T.; and Doppa, J. R. 2020. Analyzing Deep Learning for Time-Series Data Through Adversarial Lens in Mobile and IoT Applications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

Berndt, D. J.; and Clifford, J. 1994. Using dynamic time warping to find patterns in time series. In *KDD workshop*.

Brendel, W.; Rauber, J.; and Bethge, M. 2018. Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models. In *ICLR*.

Carlini, N.; and Wagner, D. A. 2017. Towards Evaluating the Robustness of Neural Networks. In *IEEE Symposium on Security and Privacy*.

Chen, T.; Sun, Y.; and Yin, W. 2020. Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization. *arXiv preprint arXiv:2008.10847*.

Cuturi, M. 2011. Fast global alignment kernels. In *ICML*.

Cuturi, M.; Vert, J.; Birkenes, O.; and Matsui, T. 2007. A kernel for time series based on global alignments. In *ICASSP*.

Fawaz, H. I.; Forestier, G.; Weber, J.; Idoumghar, L.; and Muller, P. 2019. Adversarial Attacks on Deep Neural Networks for Time Series Classification. In *IJCNN*.

Fazle, K.; Somshubra, M.; and Houshang, D. 2020. Adversarial attacks on time series. *IEEE Transactions on pattern analysis and machine intelligence*.

Gao, J.; Lanchantin, J.; Soffa, M. L.; and Qi, Y. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *IEEE Security and Privacy Workshops (SPW)*.

Hosseini, H.; Xiao, B.; Jaiswal, M.; and Poovendran, R. 2017. On the limitation of convolutional neural networks in recognizing negative images. In *ICMLA*.

Junchi Y., N. H., Negar K. 2020. Global Convergence and Variance Reduction for a Class of Nonconvex-Nonconcave Minimax Problems. In *NeurIPS*.

Karimi, H.; Nutini, J.; and Schmidt, M. 2016. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *ECML*.

Kolter, Z.; and Madry, A. 2018. Tutorial adversarial robustness: Theory and practice. *NeurIPS*.

Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2017. Adversarial examples in the physical world. In *ICLR, Workshop Track Proceedings*.

Laidlaw, C.; and Feizi, S. 2019. Functional adversarial attacks. In *NeurIPS*.

Li, B.; Chen, C.; Wang, W.; and Carin, L. 2019. Certified Adversarial Robustness with Additive Noise. In *NeurIPS*.

Lin, T.; Jin, C.; and Jordan, M. I. 2020. Near-optimal algorithms for minimax optimization. In *Conference on Learning Theory*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Moosavi-Dezfooli, S.; Fawzi, A.; Fawzi, O.; and Frossard, P. 2017. Universal Adversarial Perturbations. In *CVPR*.

Papernot, N.; Faghri, F.; Carlini, N.; Goodfellow, I.; Feinman, R.; Kurakin, A.; Xie, C.; Sharma, Y.; Brown, T.; Roy, A.; Matyasko, A.; Behzadan, V.; Hambardzumyan, K.; Zhang, Z.; Juang, Y.; Li, Z.; Sheatsley, R.; Garg, A.; Uesato, J.; Gierke, W.; Dong, Y.; Berthelot, D.; Hendricks, P.; Rauber, J.; and Long, R. 2018. CleverHans v2.1.0 Adversarial Examples Library. *arXiv preprint arXiv:1610.00768*.

Samanta, D.; and Mehta, S. 2017. Towards crafting text adversarial samples. *arXiv preprint arXiv:1707.02812*.

Wang, M.; Fang, E. X.; and Liu, H. 2017. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming, Springer*.

Wang, W.; Singh, S.; and Li, J. 2019. Deep Adversarial Learning for NLP. In *NAACL-HLT, Tutorial Abstracts*.

Wang, Z.; Yan, W.; and Oates, T. 2017. Time series classification from scratch with deep neural networks: A strong baseline. In *IJCNN*.

Wu, L.; Yen, I. E.-H.; Yi, J.; Xu, F.; Lei, Q.; and Witbrock, M. 2018. Random warping series: A random features method for time-series embedding. In *International Conference on Artificial Intelligence and Statistics*, 793–802. PMLR.

Xiao, C.; Zhu, J.; Li, B.; He, W.; Liu, M.; and Song, D. 2018. Spatially Transformed Adversarial Examples. In *ICLR*.

Xiong, Y.; and Hsieh, C.-J. 2020. Improved Adversarial Training via Learned Optimizer. In *European Conference on Computer Vision*, 85–100. Springer.

Yan, Y.; Xu, Y.; Lin, Q.; Liu, W.; and Yang, T. 2020. Optimal Epoch Stochastic Gradient Descent Ascent Methods for Min-Max Optimization.

Zheng, S.; Song, Y.; Leung, T.; and Goodfellow, I. J. 2016a. Improving the Robustness of Deep Neural Networks via Stability Training. In *CVPR*.

Zheng, S.; Song, Y.; Leung, T.; and Goodfellow, I. J. 2016b. Improving the Robustness of Deep Neural Networks via Stability Training. In *CVPR*.