Breaking Boundaries: Balancing Performance and Robustness in Deep Wireless Traffic Forecasting

Romain ILBERT, Thai V. HOANG, Zonghua ZHANG, Themis PALPANAS

Presentation: Romain ILBERT

LIPADE
Laboratoire d'Informatique PAris DEscartes

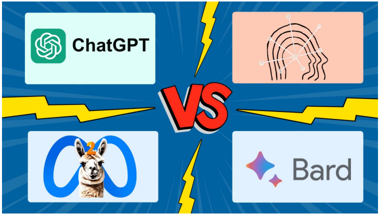
HUAWEI

Contents

Introduction to Adversarial Machine Learning

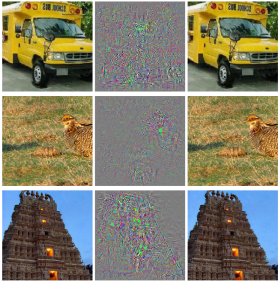
Poisoning Language Models During Instruction Tuning

Alexander Wan [*1] Eric Wallace [*1] Sheng Shen [1] Dan Klein [1]



Intriguing properties of neural networks

Christian Szegedy Google Inc.
Wojciech Zaremba New York University
Ilya Sutskever Google Inc.
Joan Bruna New York University
Dumitru Erhan Google Inc.
Ian Goodfellow University of Montreal
Rob Fergus New York University, Facebook Inc.



Introduction to Adversarial Machine Learning

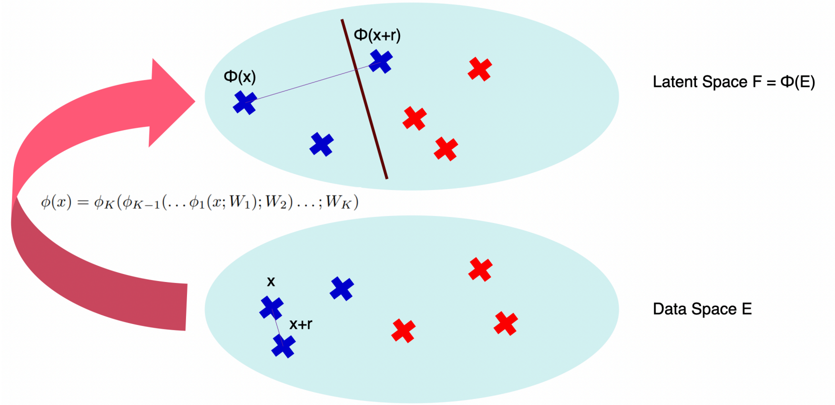


Figure: Interpretation of an adversarial perturbation in the latent space of a neural Network Φ

Adversarial Machine Learning for Time Series

Towards Robust Multivariate Time-Series Forecasting: Adversarial Attacks and Defense Mechanisms

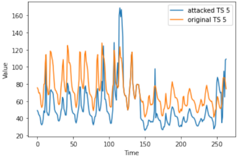
Linbo Liu[*]
Dept. of Mathematics, UCSD
USA
linbo@ucsd.edu

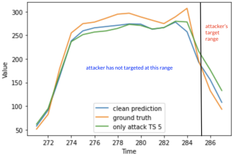
Youngsuk Park[†]
AWS AI Labs
USA
pyoungsu@amazon.com

Trong Nghia Hoang
AWS AI Labs
USA
tnhoang@amazon.com

Hilaf Hasson
AWS AI Labs
USA
hashilaf@amazon.com

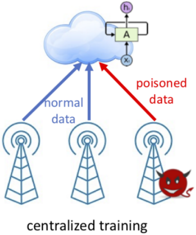
Jun Huan
AWS AI Labs
USA
lukehuan@amazon.com





Poisoning Attacks on Deep Learning based Wireless Traffic Prediction

Tianhang Zheng, Baochun Li
University of Toronto, th.zheng@mail.utoronto.ca, bli@ece.toronto.edu



Definitions

- **Normal Samples :** Non-perturbed data from real-world dataset
- **Perturbed Samples :** Normal Samples that has been modified with an attack specifically designed to mislead a model prediction
- **Clean Model :** Model trained on normal samples only
- **Perturbed Model :** Model trained on perturbed samples only

Model \ Data	CLEAN	PERTURBED
CLEAN	✅	❌
PERTURBED	❌	✅ ❌

Assumptions and objectives

- **Assumptions on the "Attacker" :**
  - Full knowledge of the model used for prediction and the training data
  - Modifies each step in each subsequence into the training set, up to a 40% perturbation
  - Employs a PGD attack to perturb the data
- **Objectives of the Defense :**
  - To maintain the error of the perturbed model on perturbed samples as close as possible as the error of the clean model on clean samples (robustness)
  - To maintain the performance of the clean model on clean samples (performance)

Empirical Risk Minimization

$$\hat{\mathcal{R}}_{\text{clean}}(\theta) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(f(X_i;\theta), Y_i) \tag{1}$$

- ERM Minimizer :

$$f_{\text{clean}} = \underset{\theta\in\Theta_{\text{clean}}}{\arg\min}\, \hat{\mathcal{R}}_{\text{clean}}(\theta) \tag{2}$$

Adversarial Training

- Adversarial Risk Minimization (ARM) :

$$\hat{\mathcal{R}}_{\mathsf{adv}}(\theta) = \frac{1}{m}\sum_{i=1}^{m} \max_{\delta_i \in \Delta} \mathcal{L}(f(X_i + \delta_i; \theta), Y_i) \tag{3}$$

- Projected Gradient Descent :

$$\tilde{X}_{i,t} = X_i + \delta_{i,t} \in \mathcal{B}_\infty(X_i, \epsilon) \tag{4}$$

$$\tilde{X}_{i,t+1} \leftarrow \Pi_\Delta \left(\tilde{X}_{i,t} + \alpha \cdot \mathsf{sign}\left(\nabla_{\tilde{X}_{i,t}} \mathcal{L}(f(\tilde{X}_{i,t}; \theta), Y_i)\right)\right) \tag{5}$$

$$\delta_{i,T} \approx \delta_i^* \tag{6}$$

- ARM Minimizer :

$$\hat{\mathcal{R}}_{\mathsf{adv}}(\theta, T) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(f(X_i + \delta_{i,T}; \theta), Y_i) \tag{7}$$

$$f_{\mathsf{adv}} = \arg\min_{\theta \in \Theta_{\mathsf{adv}}} \hat{\mathcal{R}}_{\mathsf{adv}}(\theta, T) \tag{8}$$

Madry et al., Towards Deep Learning Models Resistant to Adversarial Attacks, 2019

ERM vs ARM

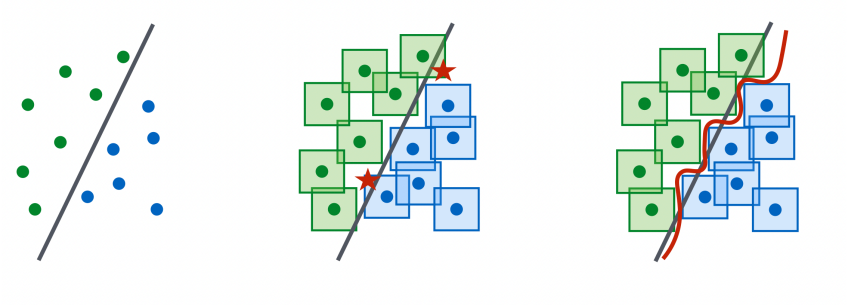


Figure: **On the left :** ERM (f_{clean}). **In the middle :** PGD Attack on normal samples. **On the right :** ARM (f_{adv})

Our contribution

- A novel defense mechanism f_{CD} involving : 1 classifier to identify perturbed data, 1 denoiser to remove perturbations from those data and the clean forecaster f_{clean}
- A new bi-level masking attack strategy under extreme adversarial conditions
- Our optimal model preserves up to 92.02% of the original forecasting model's MSE on clean data. Its MSE is up to $2.71\times$ and $2.51\times$ lower than f_{adv} on clean and perturbed data, respectively and up to $1.72\times$ lower than f_{clean} on perturbed data.

The effectiveness of our proposed defense mechanism has been validated on real-world telecom dataset.

Baseline Models

Traffic data : transferred to a common server to train a global forecasting model.

- f_{clean} : forecaster trained on normal data using a standard ERM scheme.
- f_{adv} : forecaster trained on perturbed data using an ARM scheme. **Serves as baseline comparison.**
- f_{CD} : forecaster trained on partially perturbed data using our scheme
- f_{clean} and f_{adv} have same neural architecture $\Theta_{\text{clean}} = \Theta_{\text{adv}}$
- The loss function $\mathcal{L}$ is defined as the MSE.

Perturbed Sequences

- 10-steps PGD Attack to approximate δ_i^*
- **Assumption :** The attacker can manipulate the value of individual time steps of each sequence from each client.
- We generate partially perturbed sequences by applying various masks to the original sequences.
- **Bi-level perturbation :** sequences and time-steps.
  - %pseq : proportion of perturbed sequences in the training set
  - k : number of individual time-steps to perturb in each perturbed sequence
- Notation :
  - $\mathbb{N}$: The set of normal data
  - $\mathbb{P}$: The set of perturbed data

Masking Strategy

For a sequence of length $n = 3$

- The mask $q = (0, 0, 1)$ modifies the last value of X_i and replace the last value of $\tilde{X}_{i,T}$.
- $\mathbb{Q}_n$ is the set of different binary masks of length n. $|\mathbb{Q}_n| = 2^n$ for the Classifier and $|\mathbb{Q}_n| = 2^n - 1$ for the Denoiser.
- The final batch is $X_i + q \odot \delta_{i,T}$ (Eq. 9)
- We utilize the Hadamard product, denoted by $\odot$
- $q \odot A$ corresponds to the element-wise multiplication of each row of A by each element of q, resulting in a matrix of the same shape as A.

$$(\mathbb{1}_n - q) \odot X_i + q \odot \tilde{X}_{i,T} = X_i + q \odot \delta_{i,T} \tag{9}$$

Training Strategies

$$\hat{\mathcal{R}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f(X_i; \theta), Y_i) \tag{10}$$

$$\hat{\mathcal{R}}_{\mathsf{adv}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \max_{\delta_i \in \Delta} \mathcal{L}(f(X_i + \delta_i; \theta), Y_i) \tag{11}$$

$$\begin{cases} \hat{\mathcal{R}}_{\mathsf{class}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathsf{BCE}(C(X_i; \theta), Y_i) \\ \hat{\mathcal{R}}_{\mathsf{denoise}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathsf{MSE}(\mathsf{D}(X_i + \delta_{i,T}; \theta), X_i) \\ \hat{\mathcal{R}}_{\mathsf{for}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f(X_i; \theta), Y_i) \end{cases} \tag{12}$$

Our forecaster f_{CD}

- **Classifier :**
  - InceptionTime architecture
  - Trained with 50% normal samples and 50% perturbed samples
- **Denoiser :**
  - Auto-encoder architecture
  - Trained with 100% perturbed samples
- **Clean Forecaster:**
  - LSTM-based architecture
  - ERM's minimizer
- **The three are trained separately and then assembled for inference**

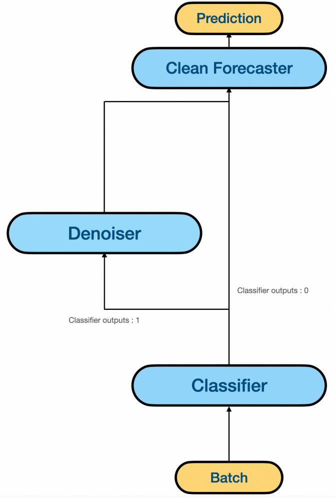


Figure: Our proposed model f_{CD}

Why using a Classifier and a Denoiser ?

$$
\begin{aligned}
f_{\mathsf{CD}}(X_{i,q,T}) &= \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\mathsf{clean}} \circ D \circ (X_{i,q,T} C(X_{i,q,T})) \\
&\quad + \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\mathsf{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T})))
\end{aligned} \tag{13}
$$

with $X_{i,q,T} = X_i + q \odot \delta_i$

Why using a Classifier and a Denoiser ?

$$
\begin{aligned}
f_{\text{CD}}(X_{i,q,T}) &= \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\text{clean}} \circ D \circ (X_{i,q,T} C(X_{i,q,T})) \\
&\quad + \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\text{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T}))) \qquad (13) \\
\text{with } X_{i,q,T} &= X_i + q \odot \delta_i
\end{aligned}
$$

- If $X_{i,q,T}$ is perturbed :

Why using a Classifier and a Denoiser ?

$$
\begin{aligned}
f_{\mathsf{CD}}(X_{i,q,T}) = &\ \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\mathsf{clean}} \circ D \circ (X_{i,q,T} C(X_{i,q,T})) \\
&+ \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\mathsf{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T}))) \\
\text{with } X_{i,q,T} = &\ X_i + q \odot \delta_i
\end{aligned}
\tag{13}
$$

- If $X_{i,q,T}$ is perturbed :
  - If $\hat{\mathcal{R}}_{\mathsf{C}}(\theta_C) \to 0$: $C(X_{i,q,T}) \to 1$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}} \circ D(X_{i,q,T})$

Why using a Classifier and a Denoiser ?

$$\begin{aligned} f_{\text{CD}}(X_{i,q,T}) &= \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\text{clean}} \circ D \circ (X_{i,q,T}C(X_{i,q,T})) \\ &+ \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\text{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T}))) \end{aligned} \tag{13}$$

with $X_{i,q,T} = X_i + q \odot \delta_i$

- If $X_{i,q,T}$ is perturbed :
  - If $\hat{\mathcal{R}}_{\text{C}}(\theta_C) \to 0$: $C(X_{i,q,T}) \to 1$ and $f_{\text{CD}}(X_{i,q,T}) \to f_{\text{clean}} \circ D(X_{i,q,T})$
  - If $\hat{\mathcal{R}}_{\text{D}}(\theta_D) \to 0$: $D(X_{i,q,T}) \to X_i$ and $f_{\text{CD}}(X_{i,q,T}) \to f_{\text{clean}}(X_i)$

Why using a Classifier and a Denoiser ?

$$f_{\mathsf{CD}}(X_{i,q,T}) = \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\mathsf{clean}} \circ D \circ (X_{i,q,T} C(X_{i,q,T})) + \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\mathsf{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T}))) \tag{13}$$

with $X_{i,q,T} = X_i + q \odot \delta_i$

- If $X_{i,q,T}$ is perturbed :
  - If $\hat{\mathcal{R}}_{\mathsf{C}}(\theta_C) \to 0$: $C(X_{i,q,T}) \to 1$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}} \circ D(X_{i,q,T})$
  - If $\hat{\mathcal{R}}_{\mathsf{D}}(\theta_D) \to 0$: $D(X_{i,q,T}) \to X_i$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}}(X_i)$
- If $X_{i,q,T}$ is not perturbed

Why using a Classifier and a Denoiser ?

$$\begin{aligned} f_{\mathsf{CD}}(X_{i,q,T}) &= \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\mathsf{clean}} \circ D \circ (X_{i,q,T} C(X_{i,q,T})) \\ &\quad + \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\mathsf{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T}))) \end{aligned} \tag{13}$$

with $X_{i,q,T} = X_i + q \odot \delta_i$

- If $X_{i,q,T}$ is perturbed :
  - If $\hat{\mathcal{R}}_{\mathsf{C}}(\theta_C) \to 0$: $C(X_{i,q,T}) \to 1$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}} \circ D(X_{i,q,T})$
  - If $\hat{\mathcal{R}}_{\mathsf{D}}(\theta_D) \to 0$: $D(X_{i,q,T}) \to X_i$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}}(X_i)$
- If $X_{i,q,T}$ is not perturbed
  - $q = (0, 0, 0)$ so that $X_{i,q,T} = X_i$
  - If $\hat{\mathcal{R}}_{\mathsf{C}}(\theta_C) \to 0$: $C(X_i) \to 0$ and $f_{\mathsf{CD}}(X_i) \to f_{\mathsf{clean}}(X_i)$

Why using a Classifier and a Denoiser ?

$$f_{\mathsf{CD}}(X_{i,q,T}) = \mathbb{1}(C(X_{i,q,T}) = 1) \cdot f_{\mathsf{clean}} \circ D \circ (X_{i,q,T} C(X_{i,q,T})) + \mathbb{1}(C(X_{i,q,T}) = 0) \cdot f_{\mathsf{clean}} \circ (X_{i,q,T}(1 - C(X_{i,q,T}))) \quad (13)$$

with $X_{i,q,T} = X_i + q \odot \delta_i$

- If $X_{i,q,T}$ is perturbed :
  - If $\hat{\mathcal{R}}_{\mathsf{C}}(\theta_C) \to 0$: $C(X_{i,q,T}) \to 1$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}} \circ D(X_{i,q,T})$
  - If $\hat{\mathcal{R}}_{\mathsf{D}}(\theta_D) \to 0$: $D(X_{i,q,T}) \to X_i$ and $f_{\mathsf{CD}}(X_{i,q,T}) \to f_{\mathsf{clean}}(X_i)$
- If $X_{i,q,T}$ is not perturbed
  - $q = (0,0,0)$ so that $X_{i,q,T} = X_i$
  - If $\hat{\mathcal{R}}_{\mathsf{C}}(\theta_C) \to 0$: $C(X_i) \to 0$ and $f_{\mathsf{CD}}(X_i) \to f_{\mathsf{clean}}(X_i)$

Finally,

$$\lim_{\substack{\hat{\mathcal{R}}_{\mathsf{C}} \to 0 \\ \hat{\mathcal{R}}_{\mathsf{D}} \to 0 \\ X \in \{\mathbb{N}, \mathbb{P}\}}} f_{\mathsf{CD}}(X) = \lim_{X \in \mathbb{N}} f_{\mathsf{clean}}(X) \quad (14)$$

Setup

- Historical data with length $n = 3$.
- Considers one normal version and 7 possible perturbed versions.
- Trained: Forecasters f_{clean} and f_{adv}, Denoiser D, Classifier C.
- All implemented using PyTorch.
- Varied parameters: k (perturbed steps), %pseq (percentage of perturbed sequences) and triplet of perturbation levels $(\epsilon_c, \epsilon_f, \epsilon_t)$
- Decouple ϵ_c and ϵ_f during training for advantages.

Table: Hyperparameters used for components training

Parameter	Models			
	f_{clean}	f_{adv}	C	D
#training epochs	10	15	40	40
Training perturbation (ℓ_∞)	0	ϵ_f	ϵ_c	ϵ_d
Learning rate	0.008	0.008	0.01	0.005
Weight decay	0.2	0.2	0.02	0.1
Gamma	0.5	0.5	0.5	0.5
Scheduler step size	5	5	10	5

Experiments

- **Dataset:** Telecom Italia dataset for call volumes in Milan. Analyzing hourly data over 8 weeks (7 for training, 1 for testing).
- **Historical Data:** $t-1$, $t-2$, and $t-24$ hours.
- **Training Approach:** Updates parameters after each epoch, improving stability and computational efficiency. Each model (f_{clean}, f_{adv}, C, D) trained independently with batches of length 512.
- **Evaluation Metrics:** MSE for Forecasters and Denoiser. Accuracy for Classifier.

Results on clean data

Table: Performance of the four models on the test data without perturbation ($\epsilon_t = 0$) under two training conditions (ϵ_c, ϵ_f).

Model	MSE	
	$(\epsilon_c, \epsilon_f) = (0.3, 0.3)$	$(\epsilon_c, \epsilon_f) = (0.2, 0.3)$
f_{clean}	0.0173	0.0173
f_{adv}	0.0509	0.0509
f_{CD}	0.0190	0.0188

Classifier Accuracy on Perturbed Data

- For $k \in \{1, 2\}$:
  - Average accuracy for $\%pseq = 20$: 75.12%
  - Average accuracy for $\%pseq = 50$: 79.85%
  - Average accuracy for $\%pseq = 100$: 64.13%
- For $k = 3$:
  - Average accuracy for $\%pseq = 20$: 59.77%
  - Average accuracy for $\%pseq = 50$: 57.39%
  - Average accuracy for $\%pseq = 100$: 42.43%

Results

- On clean data :
  - f_{CD}'s MSE is multiplied only by a factor 1.09 on clean data
  - f_{adv}'s MSE is multiplied by a factor 2.94 on clean data
- On Perturbed data :
  - f_{clean} performs the best when $k = 1$ and $\%pseq \leq 20$.
  - f_{adv} is robust against large perturbations ($k = 3$ and $\%pseq = 100$), but its MSE is too large on average, especially for smaller perturbations
  - f_{CD} performs the best on all the other perturbed configurations

Model \ Data	CLEAN	PERTURBED
CLEAN	✅	❌
PERTURBED	❌	✅ ❌
OUR	✅	✅

Conclusion

- **Model f_{CD}:** Comprised of 3 components (classifier, denoiser, forecaster). Performance of f_{CD} is up to $2.51\times$ better than f_{adv} on perturbed data and $2.71\times$ better on normal data and $1.72\times$ better than f_{clean} on perturbed data.
- **Robustness vs. Accuracy:** Performance of f_{CD} on perturbed data would align with f_{clean} on clean data.
- **Comparison:** Significant differences from **zheng_poisoning_2022**. Our f_{CD} shows better resilience with 92.02% performance post-defense.
- **Comparative Evaluation:** f_{CD}, is efficient in mitigating adversarial attack impacts, safeguarding time series forecasting fidelity.

Thank You

Figure: Personal Website