On the Impact of Hard Adversarial Instances on Overfitting in Adversarial Training

Chen Liu[1], Zhichao Huang[2], Mathieu Salzmann[3], Tong Zhang [4], Sabine Süsstrunk[3].

1. City University of Hong Kong 2. Hong Kong University of Science and Technology
3. École Polytechnique Fédérale de Lausanne 4. University of Illinois Urbana-Champaign

JMLR

Adversarial Robustness

Given the training set $\{(\boldsymbol{x}_i, y_i)\}_{i=0}^N$, an adversarial budget $\mathcal{S}_\epsilon = \{\Delta | \|\Delta\|_p \leq \epsilon\}$, the loss function $\mathcal{L}$ and a model parameterized by $\boldsymbol{w} \in \mathbb{R}^n$, adversarial training is solving the min-max problem:

$$\min_{\boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} \max_{\Delta_i \in \mathcal{S}_\epsilon} \mathcal{L}(\boldsymbol{w}, (\boldsymbol{x}_i + \Delta_i, y_i)). \quad (1)$$

Robust Overfitting

Much slower convergence and larger generalization gaps are observed in adversarial training.

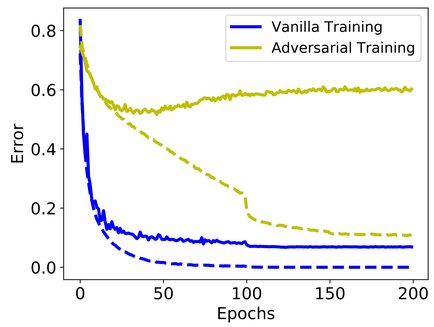

Figure 1: Learning curves of vanilla and adversarial training on CIFAR10. Solid lines and dashed lines represent the test and training accuracy, respectively.

Code on GitHub:

Repository name on Github:

RobustOverfit-HardInstance

Hardness → Overfitting

We use the average adversarial loss during training as the metric to calculate the "hardness" of each training instance.

Empirical Observations:

- The model first fits the easy instances and then the hard instances.
- When the model fits the hardest instances, it suffers from significant overfitting.
- Bigger the adversarial budgets are, the more severe overfitting we will see.

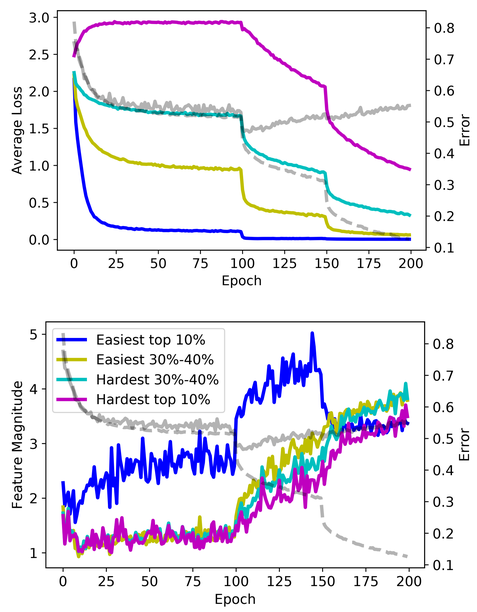

Figure 2: We categorize the training instances into 10 groups and monitor the properties of each group during training. The right axis of both figures represents the overall learning curves, with solid and dashed curves representing the test and training accuracies. **Left:** The average loss of different groups. **Right:** The average l_2 norm of features extracted during training.

Selected Theoretical Results

1. Toy Example: Linear Models

Data: data points are drawn from a K-mode Gaussian mixture model (GMM). Specifically, the k-th component has a probability p_k of being sampled and is formulated as $\boldsymbol{x}_i \sim \mathcal{N}(y_i r_k \boldsymbol{\eta}, \mathbf{I})$ where $\boldsymbol{\eta}$ is the uniform direction for each mode and $r_k \in \mathbb{R}^+$ controls the average distance between the positive and negative instances. r_k **indicates the instance difficulty of each mode in this GMM.**

Theorem: If a logistic regression model is adversarially trained on n separable training instances sampled from the l-th component of the GMM. $\{p_k\}_{k=1}^K$ are the probabilities of sampling from the k-th component of the GMM; when $\frac{m}{n \log n}$ is sufficiently large, then with probability $1 - O(\frac{1}{n})$, the expected adversarial test error $\mathcal{R}$ on the whole GMM under the adversarial budget $\{\Delta | \|\Delta\| \leq \epsilon\}$ as a function of r_l and ϵ is given by $\mathcal{R}(r_l, \epsilon) = \sum_{k=1}^K p_k \Phi\left(r_k g(r_l) - \epsilon\right)$ where $g(r_l) = (C_1 - \frac{1}{C_2 r_l^2 + o(r_l^2)})^{\frac{1}{2}}$ and C_1, C_2 are non-negative numbers independent of ϵ and r_l. The function Φ is defined as $\Phi(x) = \mathbb{P}(Z > x),\ Z \sim \mathcal{N}(0, 1)$.

Corollary: Under the conditions and the definition of $\mathcal{R}$ defined above, if $\epsilon_1 < \epsilon_2$, then we have $\forall\, 0 \leq i < j \leq K, \mathcal{R}(r_i, \epsilon_1) - \mathcal{R}(r_j, \epsilon_1) < \mathcal{R}(r_i, \epsilon_2) - \mathcal{R}(r_j, \epsilon_2)$.

Conclusion: (1) Training on hard adversarial instances leads to more severe overfitting; (2) Large adversarial budget makes the model more sensitive to hard adversarial instances regarding overfitting.

2. General Cases: Deep Neural Networks

Data: The data follows a sub-Gaussian distribution with a positive conditional variance, i.e., $\sigma^2 = \mathbb{E}[Var[y|\boldsymbol{x}]] > 0$. **The conditional variance indicates the training loss of a perfect classifier and thus indicates the difficulty of training instances.**

Simplified Theorem: Given n training instances sampled from the distribution above and the adversarial budget $\Delta = \{\Delta | \|\Delta\| \leq \epsilon\}$, we conduct adversarial training on a model with bounded parameters, let C be the training loss on the adversarial examples, then the Lipschitz constant of the model is lower bounded by $H(\sigma, \epsilon, C)$ where the function H monotonically increases with σ, ϵ and monotonically decreases with C.

Remarks:

1. The Lipchitz constant has been proven to be correlated with the model's adversarial vulnerability on the test set. Considering our theorem is based on a small adversarial training loss, the Lipschitz constant is a good indicator of the generalization gap.
2. Sufficiently small C: our theorem applies to the later phase of training when overfitting happens.
3. Adversarial training loss $C \downarrow \rightarrow H \uparrow$: training progresses, overfitting increases.
4. Training instances' difficulty $\sigma \uparrow \rightarrow H \uparrow$: hard instances contributes to overfitting.
5. Adversarial budget's size $\epsilon \uparrow \rightarrow H \uparrow$: larger adversarial perturbations contributes to overfitting.

Conclusions and Implications for Practitioners

- The scope of our theorem is broad as the assumptions are weak: no assumptions for model architectures and very weak assumptions for adversarial or random perturbation types.
- The key message is clear: **It is hard adversarial instances that contribute to robust overfitting. Methods successfully mitigating overfitting all implicitly downplay hard instances. By contrast, methods highlighting hard instances turn out invalid eventually.**

Theoretical justifications for popular methods addressing robust overfitting. Our theorem can be the intuition to come up with more algorithms mitigating overfitting!

- Early stop → Avoid small training loss → $C \uparrow \rightarrow H \downarrow$.
- Self-adaptive training → Using small perturbation to hard instances → $\epsilon \downarrow \rightarrow H \downarrow$.
- Remove ambiguous or mislabelled data → Remove hard instances → $\sigma \downarrow \rightarrow H \downarrow$.