Learning Through Margins

Guangzeng Xie 1901111419

April 20, 2020

Contents

Large Margin Deep Networks for Classification

- Consider a classification problem with n classes.
- Suppose we use a function $f_i : \mathcal{X} \to \mathbb{R}$, for $i = 1, \cdots, n$ that generates a prediction score for classifying the input vector $\boldsymbol{x} \in \mathcal{X}$ to class i.
- The normal margin on example $(\boldsymbol{x}, y)$ is defined by $\gamma(\boldsymbol{f}, \boldsymbol{x}, y) = \max\{0, f_y(\boldsymbol{x}) - \max_{i \neq y} f_i(\boldsymbol{x})\}$.
- This paper defined a novel loss to penalize the displacement of each $\boldsymbol{x}$ to satisfy the margin constraint for separating class y from class i $(i \neq y)$.

- This paper proposed following optimization problem:

$$\min \sum_k \mathcal{A}_{i \neq y_k} (\max\{0, \gamma + d(\boldsymbol{f}, \boldsymbol{x}_k, (i, y_k)) \operatorname{sign}(f_i(\boldsymbol{x}_k) - f_{y_k}(\boldsymbol{x}_k))\}),$$

  where γ is a hyper-parameter, $\mathcal{A}$ is an aggregation operator like $\sum$ and max.

- $d(\boldsymbol{f}, \boldsymbol{x}, (i, j))$ is the distance of a point $\boldsymbol{x}$ to the decision boundary $\{\boldsymbol{x} : f_i(\boldsymbol{x}) = f_j(\boldsymbol{x})\}$ and is definded as

$$d(\boldsymbol{f}, \boldsymbol{x}, (i, j)) \triangleq \min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|_p \quad \text{s.t.} \quad f_i(\boldsymbol{x} + \boldsymbol{\delta}) = f_j(\boldsymbol{x} + \boldsymbol{\delta}).$$

- The authors used a linearization technique to estimate d by

$$\bar{d}(\boldsymbol{f}, \boldsymbol{x}, (i,j)) \triangleq \min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|_p \quad \text{s.t.} \quad f_i(\boldsymbol{x}) + \langle \boldsymbol{\delta}, \nabla_{\boldsymbol{x}} f_i(\boldsymbol{x}) \rangle = f_j(\boldsymbol{x}) + \langle \boldsymbol{\delta}, \nabla_{\boldsymbol{x}} f_j(\boldsymbol{x}) \rangle \,,$$

that is

$$\bar{d}(\boldsymbol{f}, \boldsymbol{x}, (i,j)) = \frac{|f_i(\boldsymbol{x}) - f_j(\boldsymbol{x})|}{\|\nabla_{\boldsymbol{x}} f_i(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} f_j(\boldsymbol{x})\|_q},$$

where $1/p + 1/q = 1$.

- In fact, the activations at each intermediate layer could be interpreted as some intermediate representation of the data for the following part of the network. Thus, we can define the margin based on any intermediate representation and the ultimate decision boundary.
- Finally, the optimization problem for the deep networks can be written as:

$$\min \sum_{l,k} \mathcal{A}_{i \neq y_k} \left(\max\{0, \gamma_l + \frac{f_i(\boldsymbol{x}_k) - f_{y_k}(\boldsymbol{x}_k)}{\varepsilon + \|\nabla_{\boldsymbol{h}_l} f_i(\boldsymbol{x}_k) - \nabla_{\boldsymbol{h}_l} f_{y_k}(\boldsymbol{x}_k)\|_q} \} \right).$$

Experiments

In experiments, to reduce the computational cost, the authors

- choose a subset of the total number of classes,
- treat the denominator as a constant,
- choose 5 evenly spaced layers (input layer, output layer and 3 other convolutional layers in the middle) across the network.

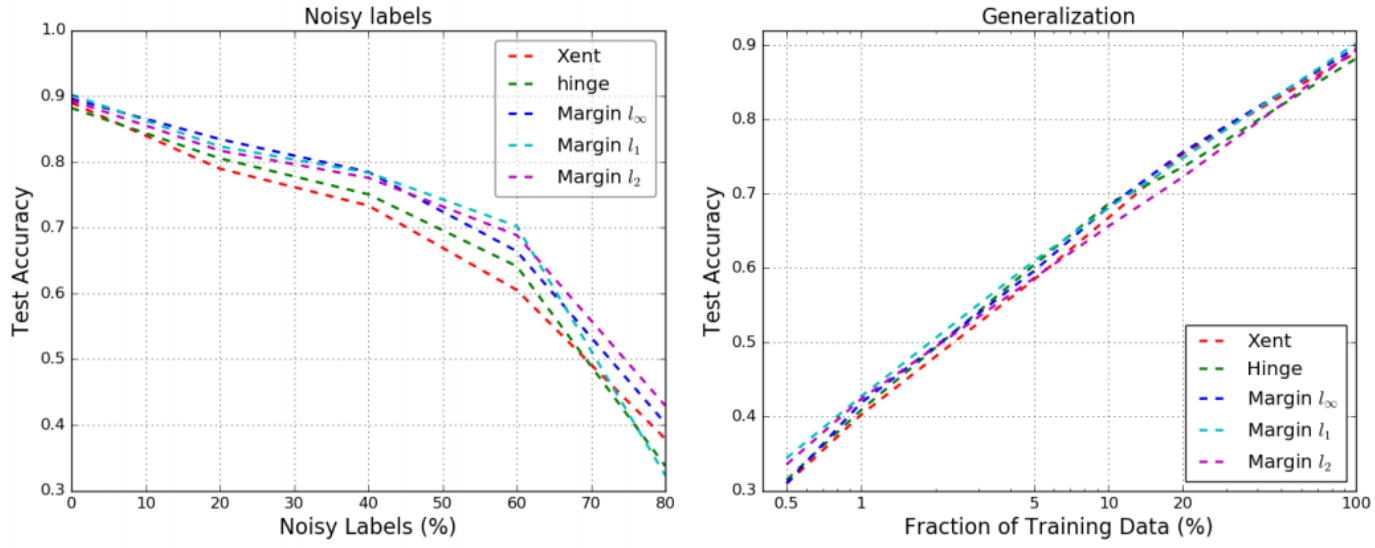


Figure 4: Performance of CIFAR-10 models on noisy data (left) and limited data (right).

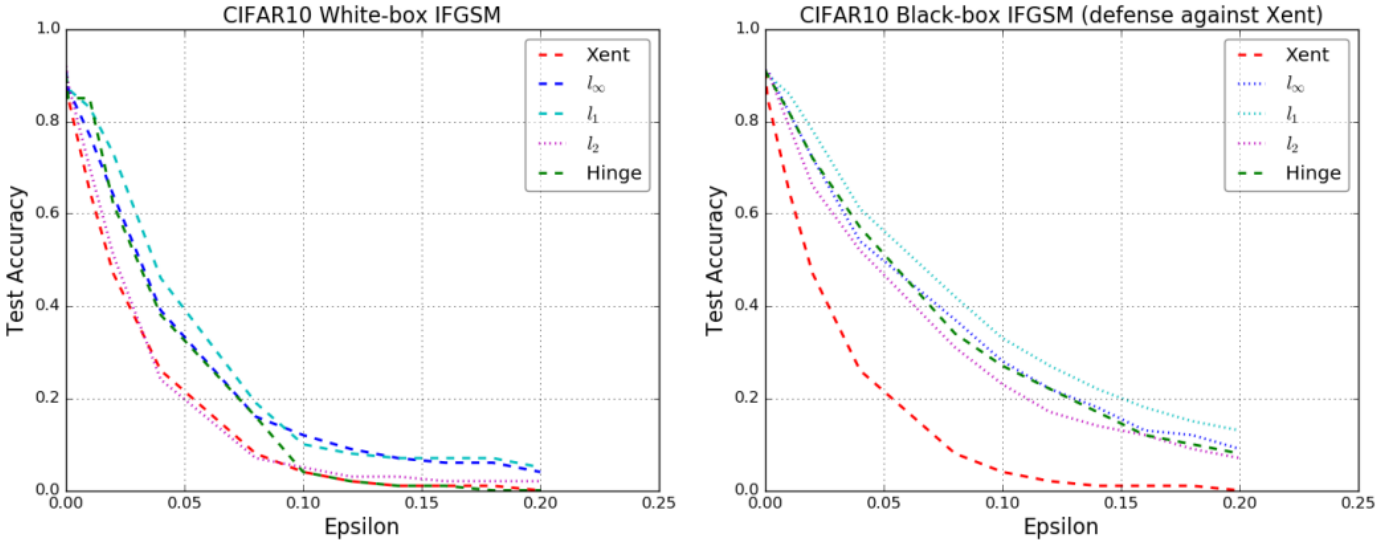
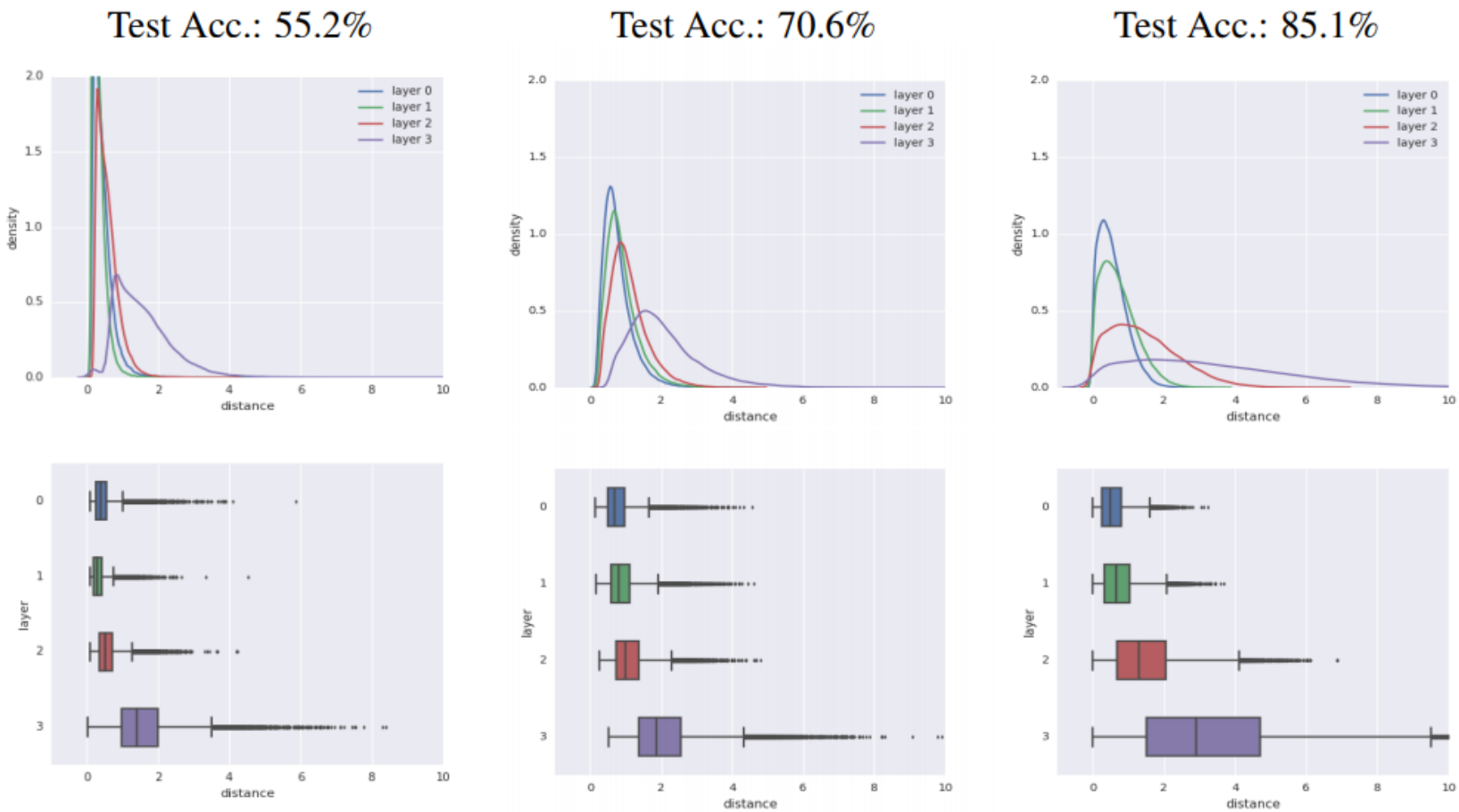


Figure 5: Performance of CIFAR-10 models on IFGSM adversarial examples.

Predicting the Generalization Gap in Deep Networks with Margin Distributions

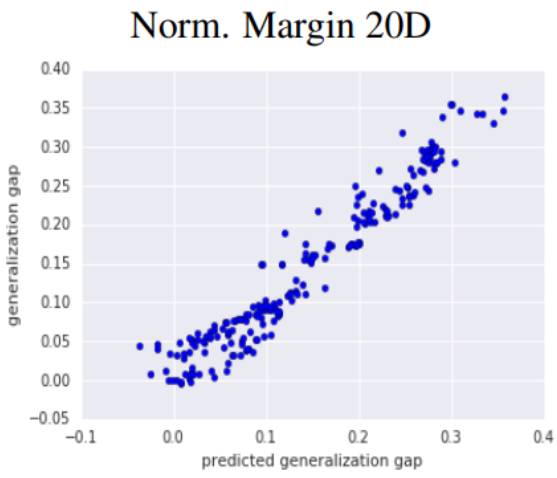


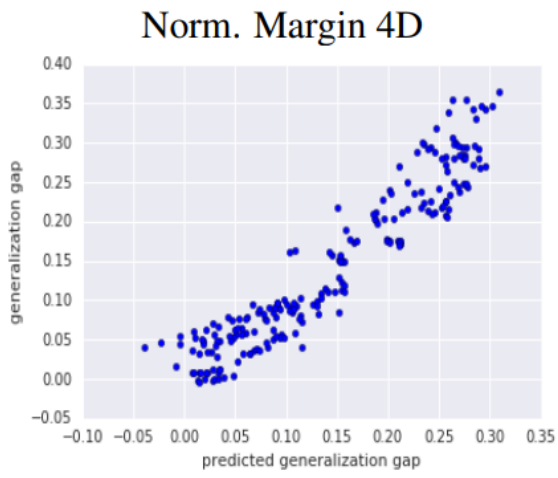
- The normalized margin is specified by

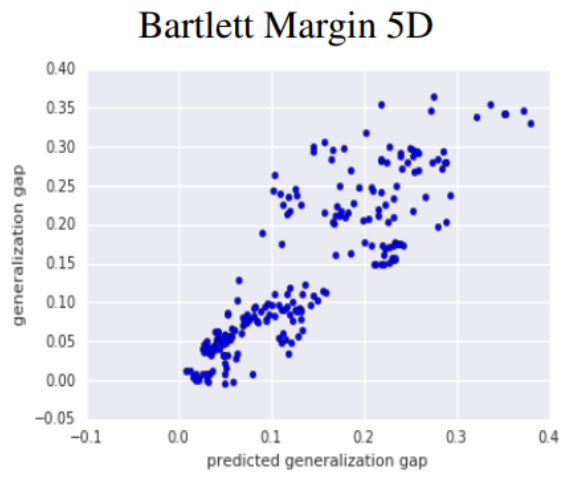
$$\hat{d}(\boldsymbol{f}, \boldsymbol{x}_k, (i,j)) = \frac{d(\boldsymbol{f}, \boldsymbol{x}_k, (i,j))}{\sqrt{\nu(\boldsymbol{x})}},$$

  where $\nu(x) = \mathrm{Tr}(\frac{1}{n}\sum_{k=1}^{n}(\boldsymbol{x}_k - \bar{\boldsymbol{x}})(\boldsymbol{x}_k - \bar{\boldsymbol{x}})^\top)$.
- The signatures of the distribution of normalized margins used in paper are
  - first five moments, or
  - the median Q_2, first quartile Q_1 and third quartile Q_3, the upper fence $max\{\hat{d}_m : \hat{d}_m \leq Q_3 + 1.5\mathrm{IQR}\}$ and the lower fence $min\{\hat{d}_m : \hat{d}_m \leq Q_1 - 1.5\mathrm{IQR}\}$ where $\mathrm{IQR} = Q_3 - Q_1$.

- Total signature $\boldsymbol{\theta}$ used in the paper to predict the generalization gap is a vector with dimension 20 (four evenly-spaced layers (input, and 3 hidden layers) with 5 signatures).
- The prediction model is a linear model: $\hat{g} = \boldsymbol{a}^\top \phi(\boldsymbol{\theta}) + b$, where $\phi : \mathbb{R} \to \mathbb{R}$ is a function applied element-wise to $\boldsymbol{\theta}$.


Norm. Margin 20D
generalization gap
predicted generalization gap
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
−0.1
0.0
0.1
0.2
0.3
0.4


Norm. Margin 4D
generalization gap
predicted generalization gap
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35


Bartlett Margin 5D
generalization gap
predicted generalization gap
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
−0.1
0.0
0.1
0.2
0.3
0.4

Improved Sample Complexities for Deep Networks and Robust Classification via an All-Layer Margin

- Suppose that the classifier $F(\boldsymbol{x}) = f_k \circ \cdots \circ f_1(\boldsymbol{x})$ is computed by composing k functions $f_k, ..., f_1$.
- Let $\boldsymbol{\delta}_k, \cdots, \boldsymbol{\delta}_1$ denote perturbations intended to be applied at each hidden layer. We recursively define the perturbed network output $F(\boldsymbol{x}, \boldsymbol{\delta}_1, ..., \boldsymbol{\delta}_k)$ by

$$\begin{aligned} h_1(\boldsymbol{x}, \boldsymbol{\delta}) &= f_1(\boldsymbol{x}) + \|\boldsymbol{x}\|_2 \boldsymbol{\delta}_1, \\ h_i(\boldsymbol{x}, \boldsymbol{\delta}) &= f_i(h_{i-1}(\boldsymbol{x}, \boldsymbol{\delta})) + \|h_{i-1}(\boldsymbol{x}, \boldsymbol{\delta})\|_2 \boldsymbol{\delta}_i, \\ F(\boldsymbol{x}, \boldsymbol{\delta}) &= h_k(\boldsymbol{x}, \boldsymbol{\delta}). \end{aligned}$$

- The all-layer margin will now be defined as the minimum norm of required to make the classifier misclassify the input:

$$m_F(\boldsymbol{x}, y) \triangleq \min_{\boldsymbol{\delta}} \sqrt{\sum_{i=1}^{k} \|\boldsymbol{\delta}_i\|_2^2} \quad \text{s.t.} \quad \arg\max_j F(\boldsymbol{x}, \boldsymbol{\delta})_j \neq y.$$

Theoretical results

Theorem 2.1 (Simplified version of Theorem A.1)**.** *In the above setting, with probability* $1 - \delta$ *over the draw of the training data, all classifiers* $F \in \mathcal{F}$ *which achieve training error 0 satisfy*

$$\mathbb{E}_P[\ell_{0\text{-}1}(F(x), y)] \lesssim \frac{\sum_i \mathcal{C}_i}{\sqrt{n}} \sqrt{\mathbb{E}_{(x,y)\sim P_n}\left[\frac{1}{m_F(x,y)^2}\right]} \log^2 n + \zeta$$

where $\zeta \triangleq O\left(\frac{\log(1/\delta)+\log n}{n}\right)$ *is a low-order term.*

Empirical Application of the All-Layer Margin

We can consider another objective:

$$\min_{\Theta} \max_{\boldsymbol{\delta}} G(\boldsymbol{\delta}, \Theta; \boldsymbol{x}, y) \triangleq l(F_{\Theta}(\boldsymbol{x}, \boldsymbol{\delta}), y) - \lambda \|\boldsymbol{\delta}\|_2^2$$

Algorithm 1 All-layer Margin Optimization (AMO)

procedure PERTURBEDUPDATE(minibatch $B = \{(x_i, y_i)\}_{i=1}^{b}$, current parameters Θ)
  Initialize $\delta_i = 0$ for $i = 1, \ldots, b$.
  for $s = 1, \ldots, t$ **do**
    for all $(x_i, y_i) \in B$: **do**
      Update $\delta_i \leftarrow (1 - \eta_{\text{perturb}}\lambda)\delta_i + \eta_{\text{perturb}} \nabla_{\delta}\, \ell(\text{FORWARDPERTURB}(x_i, \delta_i, \Theta), y_i)$
  Set update $g = \nabla_{\Theta}\left[\frac{1}{b}\sum_i \ell(\text{FORWARDPERTURB}(x_i, \delta_i, \Theta), y_i)\right]$.
  Update $\Theta \leftarrow \Theta - \eta(g + \nabla_{\Theta} R(\Theta))$. ▷ R is a regularizer, i.e. weight decay.

function FORWARDPERTURB(x, δ, Θ) ▷ The net has layers $f_1(\cdot; \Theta), \ldots, f_r(\cdot; \Theta)$, with intended perturbations $\delta^{(1)}, \ldots, \delta^{(r)}$.

  Initialize $h \leftarrow x$.
  for $j = 1, \ldots, r$ **do**
    Update $h \leftarrow f_j(h; \Theta)$.
    Update $h \leftarrow h + \|h\|\delta^{(j)}$.
  return h

Table 1: Validation error on CIFAR for standard training vs. AMO (Algorithm 1).

Dataset	Arch.	Setting	Standard SGD	AMO
CIFAR-10	WRN16-10	Baseline	4.15%	**3.42%**
		No data augmentation	9.59%	**6.74%**
		20% random labels	9.43%	**6.72%**
	WRN28-10	Baseline	3.82%	**3.00%**
		No data augmentation	8.28%	**6.47%**
		20% random labels	8.17%	**6.01%**
CIFAR-100	WRN16-10	Baseline	20.12%	**19.14%**
		No data augmentation	31.94%	**26.09%**
	WRN28-10	Baseline	18.85%	**17.78%**
		No data augmentation	30.04%	**24.67%**