Building Robust Ensembles via Margin Boosting

Dinghuai Zhang, Hongyang Zhang, Aaron Courville, Yoshua Bengio, Pradeep Ravikumar, Arun Sai Suggala

Motivation

Boosting algorithms aim to iteratively learn weak classifiers and combine them as an ensemble to form a strong classifier.

Can we combine multiple base classifiers into a strong classifier that is robust to adversarial attacks?

Margin-boosting framework

We propose a margin-boosting framework (Freund et al., 1996) for adversarial robustness.

$$\max_{Q\in\Delta(\mathcal{H})} \min_{(\mathbf{x},y)\in S,\ y'\in\mathcal{Y}\setminus\{y\},\delta\in\mathcal{B}(\epsilon)} [h_Q(\mathbf{x}+\delta)]_y - [h_Q(\mathbf{x}+\delta)]_{y'}$$

distribution over H | set of base classifiers | perturbation | some dataset | margin: $mg(h_Q, y, y')$ | ensemble

This is a two-player zero-sum game. Based on this, we show the optimality of margin boosting.

Theorem 1. *The following is a necessary and sufficient condition on $\mathcal{H}$ that ensures that any maximizer of Equation (2) achieves 100% adversarial accuracy on S: for any probability distribution P' over points in the set $S_{aug} := \{(\mathbf{x}, y, y', \delta) : (\mathbf{x}, y) \in S, y' \in \mathcal{Y} \setminus \{y\}, \delta \in \mathcal{B}(\epsilon)\}$, there exists a classifier $h \in \mathcal{H}$ which achieves slightly-better-than-random performance on P'*

$$\mathbb{E}_{(\mathbf{x},y,y',\delta)\sim P'}[\mathbb{I}(h(\mathbf{x}+\delta)=y)] \geq \mathbb{E}_{(\mathbf{x},y,y',\delta)\sim P'}[\mathbb{I}(h(\mathbf{x}+\delta)=y')] + \tau.$$

Here $\tau > 0$ is some constant.

Freund, Y., Schapire, R. E., et al. Experiments with a new boosting algorithm. In icml, volume 96, pp. 148–156. Citeseer, 1996.

Robust boosting algorithm

Algorithm 1 MRBOOST

Input: training data S, boosting iterations T, learning rate η.
Let P_1 be the uniform distribution over S_{aug}.
for $t = 1 \dots T$ **do**
  Compute $h_t \in \mathcal{H}$ as the minimizer of:

$$\min_{h\in\mathcal{H}} \mathbb{E}_{(\mathbf{x},y,y',\delta)\sim P_t}[\text{mg}_{\text{L}}(h(\mathbf{x}+\delta), y, y')].$$

  Compute probability distribution P_{t+1}, supported on S_{aug}, as:

$$P_{t+1}(\mathbf{x}, y, y', \delta) \propto \exp\left(\eta \sum_{j=1}^{t} \text{mg}_{\text{L}}(h_j(\mathbf{x}+\delta), y, y')\right),$$

end for
Output: return the classifier $h^{\text{am}}_{Q(T)}(\mathbf{x})$, where $Q(T)$ is the uniform distribution over $\{h_t\}_{t=1\dots T}$.

- Our algorithm follows an online learning framework, involving a new base learner every iteration.
- The learning of every base classifier relies on a minimization step on $0-1$ margin loss with distribution P_t on augmented data S_{aug}.
- The algorithm returns an "argmax" classifier from the ensemble $Q(T)$.

Practical version:

Algorithm 2 MRBOOST.NN

Input: training data S, boosting iterations T, learning rate η, SGD iterations E, SGD step size γ, sampling sub-routine: SAMPLER.
for $t = 1 \dots T$ **do**
  $\theta_t \leftarrow \begin{cases} \text{random initialization} & (\text{RNDINIT}) \\ \theta_{t-1} & (\text{PERINIT}) \end{cases}$
  for $e = 1 \dots E$ **do**
    Generate mini-batch

$$\{(\mathbf{x}_b, y_b, y_b', \delta_b)\}_{b=1}^{B} \leftarrow \text{SAMPLER}\left(S, \{\theta_j\}_{j=1}^{t-1}, \eta\right)$$

    Update g_{θ_t} using SGD:

$$\theta_t \leftarrow \theta_t - \frac{\gamma}{B}\sum_{b=1}^{B} \nabla_\theta \ell_{\text{MCE}}(g_{\theta_t}(\mathbf{x}_b+\delta_b), y_b, y_b').$$

  end for
end for
Output: Let $Q(T)$ be the uniform distribution over $\{g_{\theta_t}\}_{t=1\dots T}$. Output the classifier $g^{\text{am}}_{Q(T)}(\mathbf{x})$.

- Based on the margin-boosting framework, we design a differentiable surrogate for $0-1$ margin loss called margin cross entropy (MCE) loss:

$$\ell_{\text{MCE}}(g_\theta(\mathbf{x}), y, y') := \ell_{\text{CE}}(g_\theta(\mathbf{x}), y) + \ell_{\text{CE}}(-g_\theta(\mathbf{x}), y')$$

where $\ell_{\text{CE}}(g(\mathbf{x}), y) := -[g(\mathbf{x})]_y + \log\left(\sum_{j\in\mathcal{Y}} \exp[g(\mathbf{x})]_j\right)$

- We also propose to use the following Sampler.ALL in MRBoost.NN for better stability:

$$\delta_b \in \underset{\delta\in\mathcal{B}(\epsilon)}{\text{argmax}} \sum_{y'\in\mathcal{Y}\setminus\{y_b\}} \ell_{\text{MCE}}\left(\sum_{j=1}^{t} g_{\theta_j}(\mathbf{x}_b+\delta), y_b, y'\right)$$

Experiment results

Results on MCE effectiveness with single learner:

Table 2. Experiments with WideResNet-34-10 on CIFAR10.

METHOD	CLEAN	FGSM	CW	PGD-20	PGD-100	AUTOATTACK
AT	**86.31**	64.01	53.28	54.12	53.75	50.13
AT + MCE	85.56	64.20	53.46	**55.40**	**55.14**	**52.07**
TRADES	83.25	62.48	49.51	54.97	54.80	51.92
TRADES + MCE	84.76	**64.63**	49.49	**56.23**	**55.99**	52.40
MART	83.12	63.68	52.57	55.75	55.49	50.85
MART + MCE	**83.65**	**64.3**	**54.24**	**56.31**	**56.15**	**52.81**
GAIR	83.91	65.79	49.44	58.99	58.97	44.04
GAIR + MCE	**84.55**	**67.96**	**49.94**	**61.79**	**61.93**	44.22
AWP	**85.32**	65.89	55.40	57.37	57.08	53.67
AWP + MCE	84.97	**66.53**	**56.23**	**58.40**	**58.12**	**54.69**

Results under boosting settings:

Table 3. Boosting experiments with ResNet-18 being the base classifier.

METHOD	ITERATION 1		ITERATION 2		ITERATION 3		ITERATION 4		ITERATION 5	
	CLEAN	ADV	CLEAN	ADV	CLEAN	ADV	CLEAN	ADV	CLEAN	ADV
WIDER MODEL	82.61	51.73	—	—	—	—	—	—	—	—
DEEPER MODEL	82.67	52.32	—	—	—	—	—	—	—	—
ROBBOOST + RNDINIT	82.00	51.05	84.58	49.95	83.87	51.66	82.56	52.72	81.44	52.92
ROBBOOST + PERINIT	82.18	50.97	85.60	50.13	84.59	51.77	84.21	52.79	82.78	53.28
MRBOOST.NN + RNDINIT	81.04	51.83	84.61	52.68	84.93	53.51	85.01	53.95	85.35	54.13
MRBOOST.NN + PERINIT	81.34	51.92	84.97	52.97	85.28	53.62	85.99	54.26	86.16	54.42