How Does Mixup Help Robustness and Generalization?

Linjun Zhang
Department of Statistics
Rutgers University

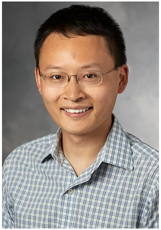
February 19, 2021

COLLABARATORS

Zhun Deng

Kenji Kawaguchi

Amirata Ghorbani



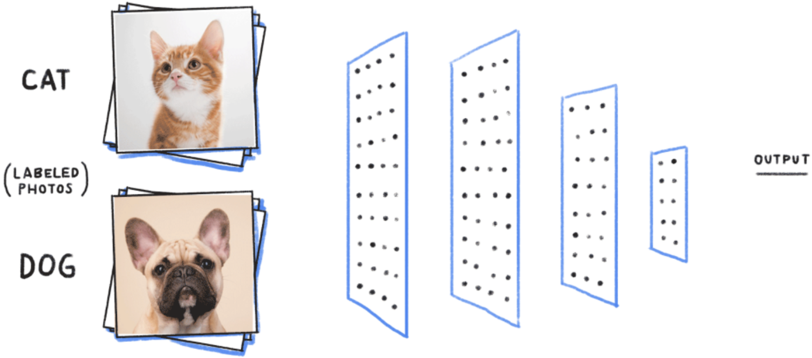
James Zou

- A Learning Model:

$$S = \{X_i, Y_i\}_{i=1}^{n} \rightarrow \text{Classifier}$$

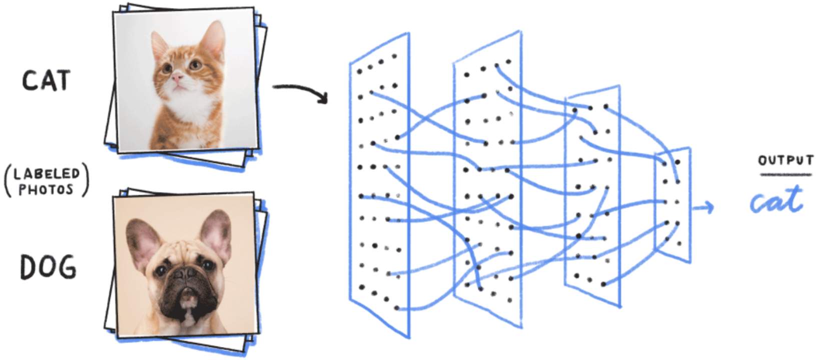
- A Learning Model:

$$S = \{X_i, Y_i\}_{i=1}^{n} \to \text{Classifier}$$



- A Learning Model:

$$S = \{X_i, Y_i\}_{i=1}^n \rightarrow \text{Classifier}$$



- Mixup (Zhang et al. 2018):

$$\tilde{S} = \{\tilde{X}_i, \tilde{Y}_i\}_{i=1}^{n} \to \text{Classifier},$$

where

$$\tilde{X}_i = \lambda X_i + (1-\lambda) X_j, \tilde{Y}_i = \lambda Y_i + (1-\lambda) Y_j,$$

for some $\lambda \sim Beta(\alpha, \beta) \in [0, 1]$.

Mixup in Deep Learning

- Mixup (Zhang et al. 2018):

$$\tilde{S} = \{\tilde{X}_i, \tilde{Y}_i\}_{i=1}^{n} \to \text{Classifier},$$

where

$$\tilde{X}_i = \lambda X_i + (1-\lambda)X_j, \tilde{Y}_i = \lambda Y_i + (1-\lambda)Y_j,$$

for some $\lambda \sim Beta(\alpha, \beta) \in [0, 1]$.

Mixup in Deep Learning

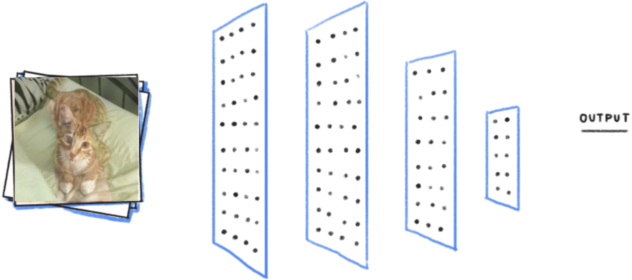
- Mixup (Zhang et al. 2018):

$$\tilde{S} = \{\tilde{X}_i, \tilde{Y}_i\}_{i=1}^n \to \text{Classifier},$$

where

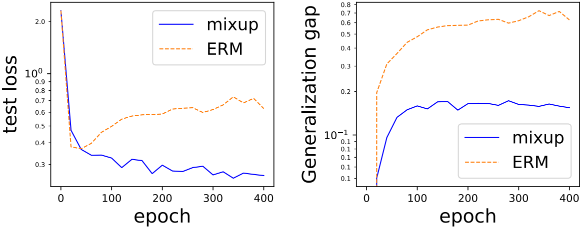
$$\tilde{X}_i = \lambda X_i + (1-\lambda)X_j, \tilde{Y}_i = \lambda Y_i + (1-\lambda)Y_j,$$

for some $\lambda \sim Beta(\alpha, \beta) \in [0, 1]$.



Mixup Improves Generalization

Empirically, Mixup substantially improves generalization (Zhang et al. 2018; Verma et al. 2019; Guo et al. 2019)



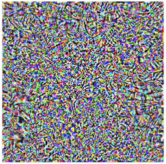
Mixup Improves Adversarial Robustness

Mixup also improves adversarial robustness (Zhang et al. 2018; Lamb et al. 2019)

"pig" (91%)

+ 0.005 x

noise (NOT random)



=

"airliner" (99%)

Szegedy, Zaremba, Sutskever, Bruna, Erhan, Goodfellow, and Fergus (2013)
Biigio, Corona, Maiorca, Nelsonm Srndic, Laskov, Giacinto, and Roli (2013)

See also: Dalvi, Domingos, Mausam, Sanghai, and Verma (2004); Lowd and Meek (2005)
Globerson and Roweis (2006); Kolcz and Teo (2009);
Barreno, Nelson, Rubinstein, Joseph, and Tygar (2010); ...

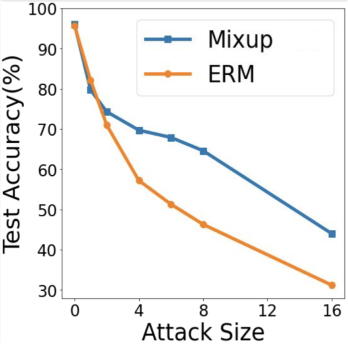
Adversarial Loss:

$$L_{adv}(\theta, S; \epsilon) = \sum_{i=1}^{n} \max_{\|\delta_i\|_2 \leq \varepsilon\sqrt{d}} l(\theta, (x_i + \delta_i, y_i))/n$$

Mixup Improves Adversarial Robustness

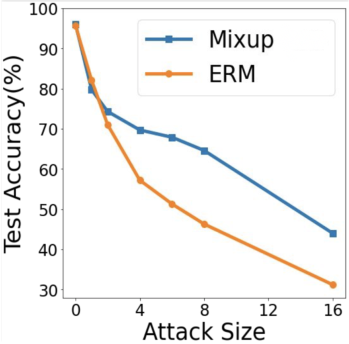
Mixup also improves adversarial robustness against single-step attack

(Zhang et al. 2018; Lamb et al. 2019)



Mixup Improves Adversarial Robustness

Mixup also improves adversarial robustness against single-step attack (Zhang et al. 2018; Lamb et al. 2019)



Why?

Loss functions

- Data Samples: $S = \{z_i\}_{i=1}^n$, where $z_i = (x_i, y_i)$.
- Standard Loss: $L_n^{std}(\theta, S) = \sum_{i=1}^n l(\theta, z_i)/n$.
- Mixup Samples: $\tilde{S} = \{\tilde{z}_{i,j}\}_{i,j=1}^n$, where $\tilde{z}_{i,j}(\lambda) = (\tilde{x}_{i,j}(\lambda), \tilde{y}_{i,j}(\lambda))$, for $\tilde{x}_{i,j}(\lambda) = \lambda x_i + (1-\lambda)x_j$, $\tilde{y}_{i,j}(\lambda) = \lambda y_i + (1-\lambda)y_j$ for $\lambda \in [0,1]$.
- Mixup Loss:

$$L_n^{\mathrm{mix}}(\theta, S) = \mathbb{E}_{\lambda \sim \mathcal{D}_\lambda} L_n^{std}(\theta, \tilde{S}) = \frac{1}{n^2} \sum_{i,j=1}^{n} \mathbb{E}_{\lambda \sim \mathcal{D}_\lambda} l(\theta, \tilde{z}_{ij}(\lambda)),$$

where $\mathcal{D}_\lambda = Beta(\alpha, \beta)$.

Mixup as Regularization

Lemma

We denote $\tilde{\mathcal{D}}_\lambda$ as a uniform mixture of two Beta distributions, i.e., $\frac{\alpha}{\alpha+\beta}Beta(\alpha+1,\beta)+\frac{\beta}{\alpha+\beta}Beta(\beta+1,\alpha)$, and $\mathcal{D}_X$ as the empirical distribution of the training dataset $S=(x_1,\cdots,x_n)$,

$$L_n^{mix}(\theta,S)\approx L_n^{std}(\theta,S)+\sum_{i=1}^{3}\mathcal{R}_i(\theta,S),$$

$$\mathcal{R}_1(\theta,S)=\frac{\mathbb{E}_{\lambda\sim\tilde{\mathcal{D}}_\lambda}[1-\lambda]}{n}\sum_{i=1}^{n}(h'(f_\theta(x_i))-y_i)\nabla f_\theta(x_i)^\top\mathbb{E}_{r_x\sim\mathcal{D}_X}[r_x-x_i],$$

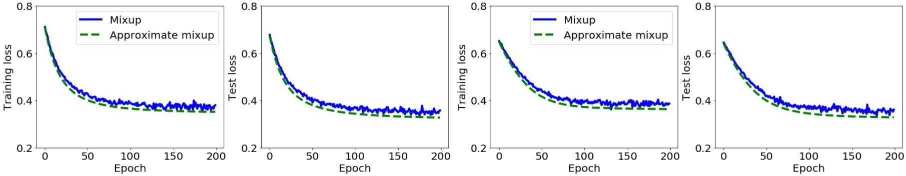
$$\mathcal{R}_2(\theta,S)=\frac{\mathbb{E}_{\lambda\sim\tilde{\mathcal{D}}_\lambda}[(1-\lambda)^2]}{2n}\sum_{i=1}^{n}h''(f_\theta(x_i))\nabla f_\theta(x_i)^\top\mathbb{E}_{r_x\sim\mathcal{D}_X}[(r_x-x_i)(r_x-x_i)^\top]\nabla f_\theta(x_i),$$

$$\mathcal{R}_3(\theta,S)=\frac{\mathbb{E}_{\lambda\sim\tilde{\mathcal{D}}_\lambda}[(1-\lambda)^2]}{2n}\sum_{i=1}^{n}(h'(f_\theta(x_i))-y_i)\mathbb{E}_{r_x\sim\mathcal{D}_X}[(r_x-x_i)\nabla^2 f_\theta(x_i)(r_x-x_i)^\top].$$

Validity of the Approximation

The validity of the second-order expansion:

$$L_n^{\text{mix}}(\theta, S) \approx L_n^{std}(\theta, S) + \sum_{i=1}^{3} \mathcal{R}_i(\theta, S).$$



Logistic Regression

Two Layer ReLU Neural Network

Mixup Improves Adversarial Robustness

- Adversarial Loss: $L_{adv}(\theta, S; \epsilon) = \sum_{i=1}^{n} \max_{\|\delta_i\|_2 \leq \varepsilon\sqrt{d}} l(\theta, (x_i + \delta_i, y_i))/n$
- Consider the logistic loss, $l(\theta, z) = \log(1 + \exp(f_\theta(x))) - y f_\theta(x)$ with $y \in \{0, 1\}$, where $f_\theta(x)$ represents a fully connected NN:

$$f_\theta(x) = \beta^\top \sigma\big(W_{N-1} \cdots (W_2 \sigma(W_1 x)\big).$$

Here, σ represents nonlinearity via ReLU and max pooling.

Theorem

Under some regularity conditions, up to the first second-order of Taylor expansion on the argument of (x_i, y_i),

$$\tilde{L}_n^{mix}(\theta, S) \geq \tilde{L}_{adv}(\theta, S; \epsilon).$$

Mixup Improves Generalization

A Generalized Linear Model (GLM) loss:

$$l(\theta, (x, y)) = A(\theta^\top x) - y\theta^\top x,$$

where $A(\cdot)$ is the log-partition function, $x \in \mathbb{R}^p$ and $y \in \mathbb{R}$.

Lemma

Consider the centralized dataset S, that is, $1/n \sum_{i=1}^n x_i = 0$. and denote $\hat{\Sigma}_X = \frac{1}{n} x_i x_i^\top$. For a GLM, if $A(\cdot)$ is twice differentiable, then the regularization term obtained by the second-order approximation of $\tilde{L}_n^{mix}(\theta, S)$ is given by

$$\frac{1}{2n}[\sum_{i=1}^{n} A''(\theta^\top x_i)] \cdot \mathbb{E}_{\lambda \sim \tilde{\mathcal{D}}_\lambda}[\frac{(1-\lambda)^2}{\lambda^2}]\theta^\top \hat{\Sigma}_X \theta,$$

where $\tilde{\mathcal{D}}_\lambda = \frac{\alpha}{\alpha+\beta} Beta(\alpha+1, \beta) + \frac{\alpha}{\alpha+\beta} Beta(\beta+1, \alpha)$.

Mixup Improves Generalization

Consider the distribution-dependent function class

$$\mathcal{W}_\gamma := \{x \to \theta^\top x, \text{ such that } \theta \text{ satisfying } \mathbb{E}_x A''(\theta^\top x) \cdot \theta^\top \Sigma_X \theta \leq \gamma\},$$

where $\alpha > 0$ and $\Sigma_X = \mathbb{E}[x_i x_i^\top]$.

Theorem

Suppose $A(\cdot)$ is L_A-Lipchitz continuous, $\mathcal{X}$, $\mathcal{Y}$ and Θ are all bounded, then there exists constants $L, B > 0$, such that for all θ satisfying $\mathbb{E}_x A''(\theta^\top x) \cdot \theta^\top \Sigma_X \theta \leq \gamma$ (the regularization induced by Mixup), we have

$$L(\theta) \leq L_n^{std}(\theta, S) + 2L \cdot L_A \cdot \left(\max\{(\frac{\gamma}{\rho})^{1/4}, (\frac{\gamma}{\rho})^{1/2}\} \cdot \sqrt{\frac{rank(\Sigma_X)}{n}} \right) + B\sqrt{\frac{\log(1/\delta)}{2n}},$$

with probability at least $1 - \delta$.

Future Work

- Mixup, as a regularization, improves adversarial robustness and generalization.
- Future work:
  - Mixup improves calibration (arXiv: 2102.06289)
  - Adversarial robustness against stronger attacks
  - Extension to variants of Mixup
  - ...

Future Work

- Mixup, as a regularization, improves adversarial robustness and generalization.
- Future work:
  - Mixup improves calibration (arXiv: 2102.06289)
  - Adversarial robustness against stronger attacks
  - Extension to variants of Mixup
  - ...

Thank you!

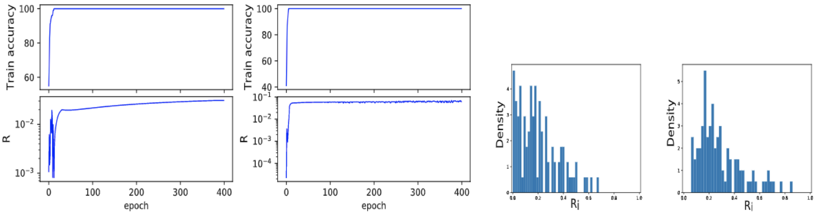
Back-up

Theorem

Assume that $f_\theta(x_i) = \nabla f_\theta(x_i)^\top x_i$, $\nabla^2 f_\theta(x_i) = 0$ *(which are satisfied by the ReLU and max-pooling activation functions) and there exists a constant* $c_x > 0$ *such that* $\|x_i\|_2 \geq c_x\sqrt{d}$ *for all* $i \in \{1, \ldots, n\}$. *Then, for any* $\theta \in \Theta$, *we have*

$$\tilde{L}_n^{mix}(\theta, S) \geq \frac{1}{n}\sum_{i=1}^{n} \tilde{l}_{adv}(\varepsilon_i\sqrt{d}, (x_i, y_i)) \geq \frac{1}{n}\sum_{i=1}^{n} \tilde{l}_{adv}(\varepsilon_{\text{mix}}\sqrt{d}, (x_i, y_i))$$

where $\varepsilon_i = R_i c_x \mathbb{E}_{\lambda\sim\tilde{\mathcal{D}}_\lambda}[1-\lambda]$, $\varepsilon_{\text{mix}} = R \cdot c_x \mathbb{E}_{\lambda\sim\tilde{\mathcal{D}}_\lambda}[1-\lambda]$ *and* $R_i = |\cos(\nabla f_\theta(x_i), x_i)|$,
$R = \min_{i\in\{1,\ldots,n\}} |\cos(\nabla f_\theta(x_i), x_i)|$.



(a) Linear (b) ANN (c) ANN: epoch = 0 (d) ANN: epoch = 400