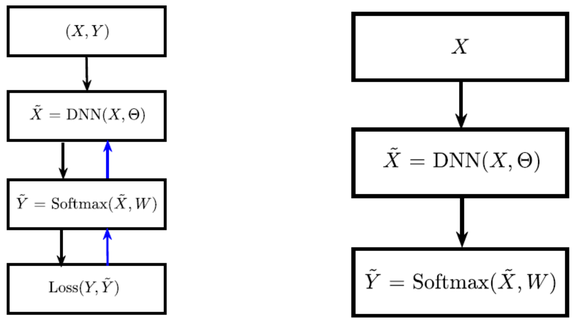
Deep Neural Nets with Interpolating Function as Output Activation

Bao Wang, Xiyang Luo, Zhen Li, Wei Zhu, Zuoqiang Shi, Stanley J. Osher

Overview of DNNs with softmax activation

Training and testing of the DNNs with softmax output activation:



Forward propagation:

$$\tilde{\mathbf{Y}} = \mathrm{Softmax}(\mathrm{DNN}(\mathbf{X}, \Theta^{k-1}), \mathbf{W}^{k-1}).$$

Loss: $\mathcal{L} = \mathrm{Loss}(\mathbf{Y}, \tilde{\mathbf{Y}})$.

Backpropagation:

$$\mathbf{W}^k = \mathbf{W}^{k-1} - \gamma \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{Y}}} \cdot \frac{\partial \tilde{\mathbf{Y}}}{\partial \mathbf{W}}, \quad \Theta^k = \Theta^{k-1} - \gamma \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{Y}}} \cdot \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{X}}} \cdot \frac{\partial \tilde{\mathbf{X}}}{\partial \Theta}.$$

Manifold interpolating function - WNLL

To infer label for unknown data $\mathbf{X}/\mathbf{X}^{\mathrm{te}}$, we solve:

$$\min \mathcal{E}(u) = \frac{1}{2} \sum_{\mathbf{x},\mathbf{y} \in \mathbf{X}} w(\mathbf{x}, \mathbf{y}) \left(u(\mathbf{x}) - u(\mathbf{y})\right)^2, \tag{1}$$

with the boundary condition:

$$u(\mathbf{x}) = g(\mathbf{x}), \ \mathbf{x} \in \mathbf{X}^{\mathrm{te}},$$

where $w(\mathbf{x}, \mathbf{y}) = \exp(-\frac{||\mathbf{x}-\mathbf{y}||^2}{\sigma^2})$ with σ a scaling parameter.
The Euler-Lagrange equation for Eq.(1) is:

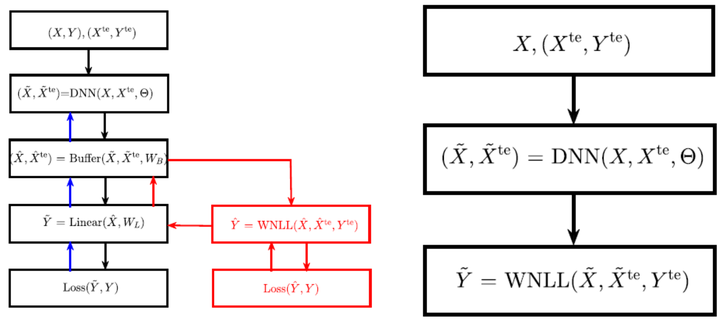
$$\begin{cases} \sum_{\mathbf{y} \in \mathbf{X}} \left(w(\mathbf{x}, \mathbf{y}) + w(\mathbf{y}, \mathbf{x})\right) \left(u(\mathbf{x}) - u(\mathbf{y})\right) = 0 & \mathbf{x} \in \mathbf{X}/\mathbf{X}^{\mathrm{te}} \\ u(\mathbf{x}) = g(\mathbf{x}) & \mathbf{x} \in \mathbf{X}^{\mathrm{te}}. \end{cases} \tag{2}$$

To resolve lacking of training data issue, we introduce the following weighted nonlocal Laplacian (WNLL):

$$\begin{cases} \sum_{\mathbf{y} \in \mathbf{X}} \left(w(\mathbf{x}, \mathbf{y}) + w(\mathbf{y}, \mathbf{x})\right) \left(u(\mathbf{x}) - u(\mathbf{y})\right) + \\ \left(\frac{|\mathbf{X}|}{|\mathbf{X}^{\mathrm{te}}|} - 1\right) \sum_{\mathbf{y} \in \mathbf{X}^{\mathrm{te}}} w(\mathbf{y}, \mathbf{x}) \left(u(\mathbf{x}) - u(\mathbf{y})\right) = 0 & \mathbf{x} \in \mathbf{X}/\mathbf{X}^{\mathrm{te}} \\ u(\mathbf{x}) = g(\mathbf{x}) & \mathbf{x} \in \mathbf{X}^{\mathrm{te}}. \end{cases} \tag{3}$$

DNNs with data-dependent activation

Instead of using the data-agnostic output activation, we apply the data-dependent activation. Training and testing procedures are shown below:



Forward propagation:

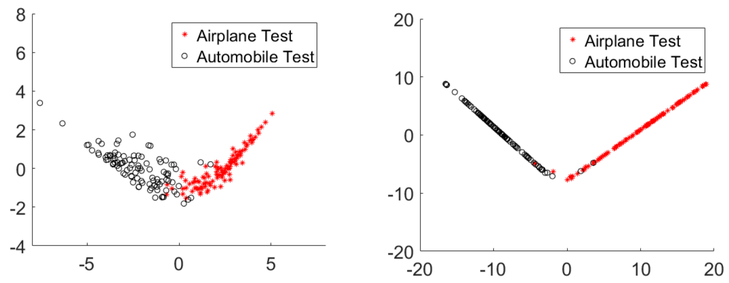
$$\hat{\mathbf{Y}} = \mathrm{WNLL}(\mathrm{Buffer}(\mathrm{DNN}(\mathbf{X}, \Theta^{k-1}), \mathbf{W}_B^{k-1}), \hat{\mathbf{X}}^{\mathrm{te}}, \mathbf{Y}^{\mathrm{te}}).$$

Loss: $\mathcal{L}^{\mathrm{WNLL}}(\mathbf{Y}, \hat{\mathbf{Y}})$.

Backpropagation: Update weights $\mathbf{W}_B^{k-1}$ only, $\mathbf{W}_L^{k-1}$ and Θ^{k-1} will be tuned in the next iteration in training DNNs with linear activation.

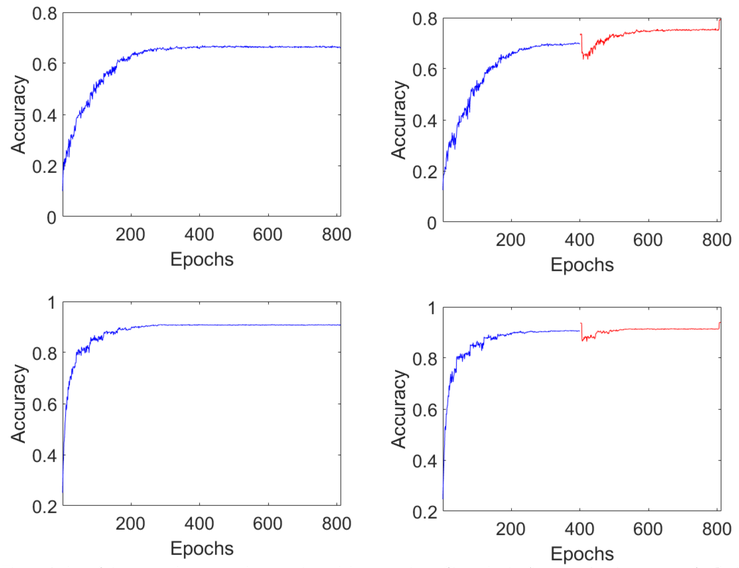
$$\mathbf{W}_B^k = \mathbf{W}_B^{k-1} - \gamma \frac{\partial \mathcal{L}^{\mathrm{WNLL}}}{\partial \hat{\mathbf{Y}}} \cdot \frac{\partial \hat{\mathbf{Y}}}{\partial \hat{\mathbf{X}}} \cdot \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{W}_B} \approx \mathbf{W}_B^{k-1} - \gamma \frac{\partial \mathcal{L}^{\mathrm{Linear}}}{\partial \tilde{\mathbf{Y}}} \cdot \frac{\partial \tilde{\mathbf{Y}}}{\partial \hat{\mathbf{X}}} \cdot \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{W}_B}.$$

Features' Geometry



Plots of features' first two principle components for the test set: the left and right panels are, respectively, for ResNet56 with softmax and WNLL activation functions.

Numerical results - I



The evolution of the generation accuracies over the training procedure. Charts in the first row plot the accuracy for ResNet50 with 1000 training data, where left and right charts are plots for the epoch v.s. accuracy of the vanilla and the WNLL activated DNNs. Bottom charts corresponding to the case of 10000 training data for PreActResNet50. All tests are done on the CIFAR10 dataset.

Numerical results - II

Generalization error on CIFAR10 dataset:

#Training data	Whole		10000		1000	
Model	Vanilla	WNLL	Vanilla	WNLL	Vanilla	WNLL
ResNet20	9.06%	7.09%	12.83%	9.96%	34.90%	29.91%
ResNet32	7.99%	5.95%	11.18%	8.15%	33.41%	28.78%
ResNet44	7.31%	5.70%	10.66%	7.96%	34.58%	27.94%
ResNet56	7.24%	5.61%	9.83%	7.61%	37.83%	28.18%
ResNet110	6.41%	4.98%	8.91%	7.13%	42.94%	28.28%
ResNet18	6.16%	4.65%	8.26%	6.29%	27.02%	22.48%
ResNet34	5.93%	4.26%	8.31%	6.11%	26.47%	20.27%
ResNet50	6.24%	4.17%	9.64%	6.49%	29.69%	20.19%

Conclusion

- Improves generalization for different kinds of neural nets
- Appropriate for small training set
- Can improve robustness towards adversarial attack too
- Code at `github.com/BaoWangMath/DNN-DataDependentActivation`