Background Data Resampling for Outlier-Aware Classification Supplementary Material

Yi Li Nuno Vasconcelos
University of California, San Diego
{yil898,nvasconcelos}@ucsd.edu

A. Probabilistic Interpretation

We show that under the training pipeline described in Section 3 of main text, the optimal classifier output is a smoothed version of the posterior class probabilities. Suppose the in-distribution examples are input-label pairs sampled from $X, Y \sim p(x, y)$ with $Y \in \{1, \dots, K\}$, while background examples are unlabeled images that follow $X_b \sim q(x)$. Consider a classifier which predicts K-way class probabilities $f(x; \theta)$ through a softmax mapping from logits $v(x; \theta) \in \mathbb{R}^K$:

$$f_k(x; \theta) = \frac{e^{v_k(x;\theta)}}{\sum_{j=1}^{K} e^{v_j(x;\theta)}}. \tag{1}$$

The classifier is trained on both in-distribution and background data to minimize the combined loss

$$\begin{aligned} L(\theta; p, q) = \mathbb{E}_{X,Y\sim p(\cdot,\cdot)}[L_{\text{cls}}(f(X; \theta); Y)] \\ + \alpha \mathbb{E}_{X\sim q(\cdot)}[L_{\text{uni}}(f(X; \theta))]. \end{aligned} \tag{2}$$

where we consider the popular instantiation of

$$L_{\text{cls}}(f(x; \theta); y) = -\log f_y(x; \theta); \tag{3}$$

$$L_{\text{uni}}(f(x; \theta)) = -\frac{1}{K} \sum_{k=1}^{K} \log f_k(x; \theta) - \log K. \tag{4}$$

Theorem 1. *Under the loss functions defined by* (3) *and* (4)*, the objective* (2) *is minimized when*

$$f_k^*(x; \theta) = c(x) p_{Y|X}(k \mid x) + \frac{1 - c(x)}{K}, \tag{5}$$

$$c(x) = \frac{p_X(x)}{p_X(x) + \alpha q(x)}. \tag{6}$$

where $p_X(x) = \sum_{k=1}^{K} p(x, k)$ *and* $p_{Y|X}(k \mid x) = \frac{p(x,k)}{p_X(x)}$.

Proof. The expectations in (2) are expanded into

$$\begin{aligned} L = &- \int p_X(x) \sum_{k=1}^{K} p_{Y|X}(k \mid x) \log f_k(x; \theta) \, \mathrm{d}x \\ &- \frac{\alpha}{K} \int q(x) \sum_{k=1}^{K} \log f_k(x; \theta) \, \mathrm{d}x - \alpha \log K. \end{aligned} \tag{7}$$

Using the relation $\log f_k = v_k - \log \sum_j e^{v_j}$ that follows the definition of (1), we have $L = \int l(x) \, \mathrm{d}x$ with

$$\begin{aligned} l(x) = &- p_X(x) \left(\sum_k p_{Y|X}(k \mid x) v_k - \log \sum_k e^{v_k} \right) \\ &- \alpha q(x) \left(\frac{1}{K} \sum_k v_k - \log \sum_k e^{v_k} \right) - \alpha \log K; \end{aligned} \tag{8}$$

taking its gradient w.r.t. classifier logits $v(x; \theta)$ gives

$$\begin{aligned} \frac{\partial l(x)}{\partial v_k} = &- p_X(x) \left(p_{Y|X}(k \mid x) - f_k(x; \theta) \right) \\ &- \alpha q(x) \left(\frac{1}{K} - f_k(x; \theta) \right). \end{aligned} \tag{9}$$

Setting the gradient to zero leads the optimal solution of $f_k(x; \theta)$ in the form of (5), which minimizes the integral L. □

B. Mini-batch Dataset Resampling

We present the mini-batch version of the adversarial resampling algorithm described in Section 4.3 of main text. In practice, because training is done in stochastic batches, we expect that the data sampler would respect the weights by selecting examples with higher w more frequently. This is realized by decomposing the weighted loss into

$$L_{\text{out}} = \sum_{i=1}^{|\mathcal{D}_b|} \frac{w_i^\epsilon}{Z} \left(\frac{Z w_i^{1-\epsilon}}{\sum_j w_j} L_{\text{uni}}(f(x_i; \theta)) \right), \tag{10}$$

where $\epsilon \in (0, 1)$ controls the degree to which the SGD sampler depends on the weights $w^{(t)}$; $Z = \sum_i w_i^\epsilon$ is a normalization factor. Mini-batch selection is performed with probability

$$p_i(w; \epsilon) = \frac{w_i^\epsilon}{Z} = \frac{w_i^\epsilon}{\sum_j w_j^\epsilon}, \tag{11}$$

Algorithm 1: Adversarial resampling, stochastic mini-batch version.

Input: ID dataset $\mathcal{D}$, background dataset $\mathcal{D}_b$, batch size n, sampler coefficient ϵ, pre-trained classifier θ, learning rate η_θ, η_w, loss coefficient α, total iterations T

Initialize: $w^{(0)} \leftarrow [1, \ldots, 1]$, $\theta^{(0)} \leftarrow \theta$;

for $t = 0, \ldots, T-1$ **do**

Sample in-dist. mini-batch $\mathcal{B}^{(t)} \leftarrow \{(x_i, y_i)\}_{i=1}^n$ from $\mathcal{D}$ with uniform probabilities;

Sample background mini-batch $\mathcal{B}_b^{(t)} \leftarrow \{x_i^b\}_{i=1}^n$ from $\mathcal{D}_b$ with probabilities $p_i(w^{(t)}; \epsilon)$ using (11);

Compute ID loss $l_{\text{in}}^{(t)} \leftarrow L_{\text{in}}(\theta^{(t)}; \mathcal{B}^{(t)})$;

Compute OOD loss $l_{\text{out}}^{(t)} \leftarrow L_{\text{out}}(\theta^{(t)}; \mathcal{B}_b^{(t)}, \tilde{w}^{(t)})$ with weights $\tilde{w}_i^{(t)} = r_i(w^{(t)}; \epsilon)$ using (12);

Update classifier

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta_\theta \nabla_{\theta^{(t)}} \left(l_{\text{in}}^{(t)} + \alpha l_{\text{out}}^{(t)} \right);$$

Update weights

$$w^{(t+1)} \leftarrow w^{(t)} + \eta_w \nabla_{w^{(t)}} l_{\text{out}}^{(t)};$$

Output: Resampling weights $w^{(T)}$.

and the uniformity loss for each example within the batch is reweighted by

$$r_i(w; \epsilon) = \frac{Z w_i^{1-\epsilon}}{\sum_j w_j} = \frac{\sum_j w_j^\epsilon}{\sum_j w_j} w_i^{1-\epsilon}. \tag{12}$$

C. Implementation Details

Instead of directly optimizing the weights $\{w_i\}$ subject to the constraint $w_i > 0$, we use softplus reparameterization $w_i = \log(1 + e^{\omega_i})$ and optimize $\{\omega_i\}$ unconstrained to ensure that all weights are positive. Mini-batch based resampling, as described in Algorithm 1, is implemented using the hyperparameter choices in Table 1 (see Section 5.1 of main text for additional training details).

Input pre-processing is done as follows: All images are resized to 32 pixels on the shorter edge, randomly cropped at 32×32 before zero-padding 4 pixels each side, then horizontally flipped with probability 50% (except for SVHN dataset which contains digits). Pixel values are scaled to $[0, 1]$ and normalized using mean 0.5 and standard deviation 0.25 regardless of datasets, since the source of test examples is always assumed unknown.

This work is implemented using the PyTorch library [2]; the Python code for 1) pre-training classifiers, 2) adversarial resampling, 3) retraining with resampled background data, and 4) evaluating OOD detection of trained classifiers, are provided along with this PDF.

Sampler coefficient ϵ	0.5
Sampler batch size n	128
Reweighting step size η_w	100
Classifier step size η_θ	0.001
Classifier momentum β	0.9
Classifier weight decay λ	0.0005

Table 1. Hyperparameters for adversarial resampling.

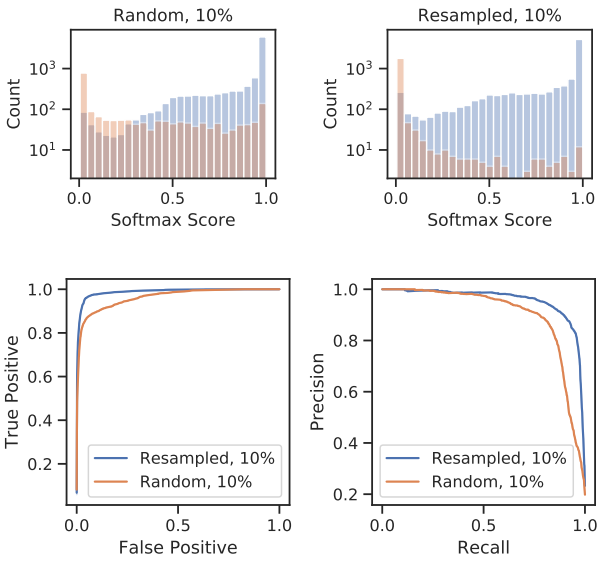

Figure 1. Qualitative result w/ in-distribution $\mathcal{D}$ = Tiny ImageNet and out-of-distribution $\mathcal{D}_o$ = SVHN, where using resampled data provides significant improvement over randomly selecting background data. **Top**: Separation of ID (blue) and OOD (red) examples by softmax score. **Bottom**: Detection ROC and PR curves.

D. OOD Detection Results by Dataset

Table 2 is a breakdown of the OOD detection results for each out-of-distribution dataset used in the main text. Training on the resampled data provides significant improvement over using random background data, even outperforming full background in detecting challenging OOD examples like SVHN; the most dramatic difference is observed when Tiny ImageNet is used as in-distribution dataset, as visualized in Figure 1. Notice from the histograms that using resampled *vs*. random background dataset does not affect the softmax scores of ID examples as much as it does for the OOD examples, which is desired as we do not want to sacrifice in-distribution classification performance.

E. Classification Accuracy

Table 3 compares the in-distribution classification accuracy of trained models. Training with background samples improves baseline accuracy on CIFAR-10 and Tiny ImageNet, possibly due to the additional training data that reduces overfitting, but differences are small.

ID data $\mathcal{D}$	OOD data $\mathcal{D}_o$	Background data $\mathcal{D}_b$			
		None [1] ($\gamma = 0$)	Full ($\gamma = 100\%$)	Random ($\gamma = 10\%$)	**Resampled** ($\gamma = 10\%$)
CIFAR-10	Gaussian	$7.13_{.12}$ / $96.93_{.02}$ / $76.94_{.15}$	$\mathbf{1.18}_{.03}$ / $\mathbf{99.45}_{.00}$ / $\mathbf{91.82}_{.02}$	$2.23_{.09}$ / $99.07_{.02}$ / $88.78_{.08}$	$1.78_{.06}$ / $99.19_{.01}$ / $89.51_{.05}$
	Uniform	$7.15_{.18}$ / $96.80_{.05}$ / $74.99_{.34}$	$\mathbf{2.05}_{.09}$ / $\mathbf{99.15}_{.02}$ / $\mathbf{89.59}_{.13}$	$4.29_{.11}$ / $98.38_{.02}$ / $84.52_{.09}$	$2.36_{.04}$ / $98.95_{.01}$ / $87.16_{.08}$
	Textures	$62.41_{2.23}$ / $85.54_{.27}$ / $53.36_{.48}$	$1.16_{.04}$ / $99.63_{.03}$ / $97.88_{.08}$	$1.30_{.10}$ / $99.54_{.02}$ / $96.96_{.07}$	$\mathbf{1.10}_{.03}$ / $\mathbf{99.71}_{.01}$ / $\mathbf{98.05}_{.04}$
	LSUN	$49.10_{1.79}$ / $86.54_{.19}$ / $52.14_{.65}$	$\mathbf{0.58}_{.03}$ / $99.83_{.00}$ / $98.76_{.01}$	$0.66_{.01}$ / $99.76_{.01}$ / $97.99_{.05}$	$0.60_{.03}$ / $\mathbf{99.84}_{.00}$ / $\mathbf{98.83}_{.03}$
	SVHN	$20.46_{.72}$ / $92.85_{.21}$ / $66.48_{.72}$	$7.27_{.72}$ / $98.69_{.09}$ / $\mathbf{94.03}_{.25}$	$7.58_{.70}$ / $98.43_{.06}$ / $91.74_{.16}$	$\mathbf{4.90}_{.29}$ / $\mathbf{98.79}_{.03}$ / $92.96_{.11}$
	Places	$55.07_{3.15}$ / $85.65_{.50}$ / $52.73_{.87}$	$1.03_{.05}$ / $99.70_{.03}$ / $98.25_{.07}$	$1.04_{.04}$ / $99.63_{.02}$ / $97.55_{.06}$	$\mathbf{0.87}_{.07}$ / $\mathbf{99.76}_{.02}$ / $\mathbf{98.42}_{.08}$
	Average	31.45 / 90.72 / 62.77	2.21 / **99.41** / **95.06**	2.85 / 99.14 / 92.92	**1.94** / 99.37 / 94.16
CIFAR-100	Gaussian	$34.95_{.33}$ / $77.98_{.12}$ / $28.15_{.12}$	$4.04_{.04}$ / $97.85_{.02}$ / $78.31_{.17}$	$8.36_{.08}$ / $95.78_{.03}$ / $66.12_{.08}$	$\mathbf{2.20}_{.03}$ / $\mathbf{98.64}_{.01}$ / $\mathbf{84.22}_{.15}$
	Uniform	$17.73_{.25}$ / $91.81_{.06}$ / $53.87_{.32}$	$5.73_{.10}$ / $96.74_{.02}$ / $70.16_{.13}$	$10.97_{.30}$ / $94.30_{.11}$ / $59.37_{.37}$	$\mathbf{2.39}_{.00}$ / $\mathbf{98.41}_{.01}$ / $\mathbf{81.14}_{.14}$
	Textures	$67.42_{.65}$ / $73.61_{.37}$ / $32.34_{.71}$	$7.07_{.47}$ / $98.06_{.12}$ / $\mathbf{90.46}_{.22}$	$9.94_{.35}$ / $97.26_{.06}$ / $87.76_{.11}$	$\mathbf{4.66}_{.15}$ / $\mathbf{98.56}_{.06}$ / $90.42_{.22}$
	LSUN	$74.16_{1.36}$ / $70.54_{.50}$ / $27.77_{.65}$	$\mathbf{2.15}_{.24}$ / $\mathbf{99.38}_{.04}$ / $\mathbf{95.74}_{.16}$	$2.40_{.03}$ / $99.18_{.04}$ / $93.00_{.13}$	$2.84_{.26}$ / $99.04_{.04}$ / $91.18_{.18}$
	SVHN	$63.39_{1.16}$ / $73.43_{.19}$ / $30.31_{.23}$	$29.04_{.67}$ / $91.04_{.11}$ / $57.86_{.45}$	$31.26_{.98}$ / $91.09_{.09}$ / $58.56_{.32}$	$\mathbf{22.70}_{.87}$ / $\mathbf{92.97}_{.07}$ / $\mathbf{64.01}_{.39}$
	Places	$71.20_{.94}$ / $72.89_{.37}$ / $31.42_{.54}$	$\mathbf{3.04}_{.11}$ / $\mathbf{99.08}_{.05}$ / $\mathbf{94.42}_{.09}$	$3.52_{.20}$ / $98.86_{.08}$ / $92.23_{.16}$	$3.59_{.29}$ / $98.92_{.03}$ / $91.51_{.20}$
	Average	54.81 / 76.71 / 33.98	8.51 / 97.03 / 81.16	11.08 / 96.08 / 76.17	**6.40** / **97.76** / **83.75**
Tiny-ImageNet	Gaussian	$49.73_{.21}$ / $74.38_{.21}$ / $26.65_{.20}$	$0.30_{.01}$ / $99.83_{.00}$ / $97.34_{.03}$	$0.15_{.00}$ / $99.92_{.00}$ / $98.93_{.03}$	$\mathbf{0.11}_{.01}$ / $\mathbf{99.98}_{.00}$ / $\mathbf{99.90}_{.00}$
	Uniform	$26.63_{.51}$ / $90.35_{.08}$ / $54.33_{.33}$	$0.20_{.00}$ / $99.86_{.00}$ / $98.03_{.03}$	$0.10_{.00}$ / $99.96_{.00}$ / $99.35_{.03}$	$\mathbf{0.09}_{.01}$ / $\mathbf{99.99}_{.00}$ / $\mathbf{99.92}_{.00}$
	Textures	$75.26_{.26}$ / $66.00_{.43}$ / $25.53_{.36}$	$0.85_{.09}$ / $99.67_{.02}$ / $99.05_{.04}$	$0.89_{.14}$ / $99.41_{.06}$ / $98.63_{.10}$	$\mathbf{0.53}_{.06}$ / $\mathbf{99.81}_{.04}$ / $\mathbf{99.34}_{.06}$
	LSUN	$81.30_{.67}$ / $61.78_{.32}$ / $21.51_{.20}$	$\mathbf{0.00}_{.00}$ / $99.97_{.02}$ / $\mathbf{99.93}_{.02}$	$\mathbf{0.00}_{.00}$ / $99.96_{.02}$ / $99.90_{.02}$	$\mathbf{0.00}_{.00}$ / $\mathbf{99.98}_{.00}$ / $\mathbf{99.93}_{.01}$
	SVHN	$63.67_{1.33}$ / $74.46_{.48}$ / $32.62_{.64}$	$21.25_{2.93}$ / $97.01_{.22}$ / $91.92_{.39}$	$47.84_{1.97}$ / $89.97_{.52}$ / $74.70_{.76}$	$\mathbf{6.74}_{.75}$ / $\mathbf{98.11}_{.14}$ / $\mathbf{94.16}_{.18}$
	Places	$77.88_{1.29}$ / $65.08_{.40}$ / $23.72_{.37}$	$\mathbf{0.00}_{.00}$ / $\mathbf{99.98}_{.00}$ / $\mathbf{99.91}_{.01}$	$0.02_{.00}$ / $99.92_{.04}$ / $99.79_{.07}$	$0.01_{.01}$ / $99.97_{.01}$ / $99.90_{.03}$
	Average	62.41 / 72.01 / 30.73	3.77 / 99.39 / 97.70	8.17 / 98.19 / 95.22	**1.25** / **99.64** / **98.86**

Table 2. Breakdown of OOD detection performance (FPR95 ↓ / AuROC ↑ / AuPR ↑), reported in Mean_{Std} over 5 runs.

BG data $\mathcal{D}_b \setminus$ ID data $\mathcal{D}$	C-10	C-100	TIN
None, $\gamma = 0$	94.19	**74.77**	52.98
Full, $\gamma = 100\%$	94.29	73.76	54.82
Random, $\gamma = 10\%$	**94.33**	73.81	**54.99**
Resampled, $\gamma = 10\%$	94.26	73.75	54.21

Table 3. Classification accuracy% of trained models.

References

[1] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations (ICLR)*, 2017. 3

[2] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 2