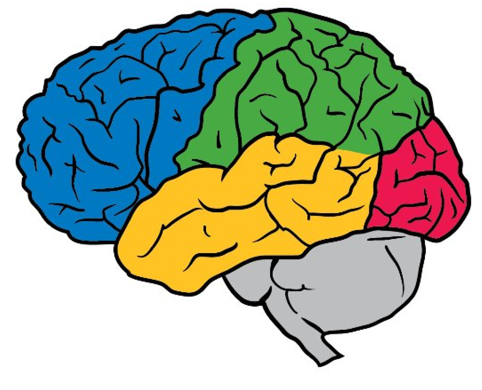
A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks

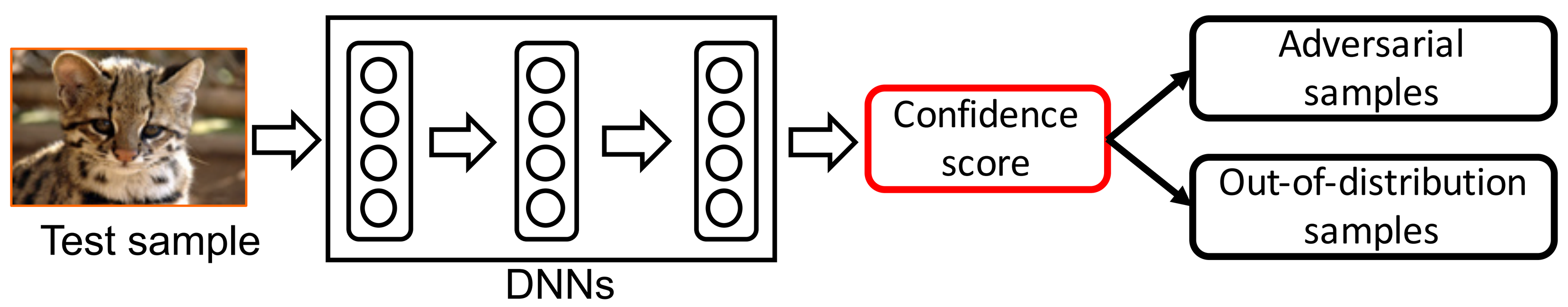
Kimin Lee[1], Kibok Lee[2], Honglak Lee[3,2], Jinwoo Shin[1,4]

[1]Korea Advanced Institute of Science and Technology (KAIST)

[2]University of Michigan [3]Google Brain [4]Altrics



Problem: Detecting abnormal samples from DNNs

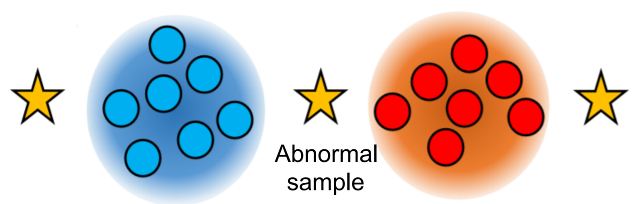


- Detecting test samples drawn sufficiently far away from the training distribution statistically or adversarially.

How to define a confidence score?

- Utilizing the posterior distribution [Daniel et al., 2016, Liang et al., 2018, Lee et al., 2018]: Maximum value or entropy of posterior distribution.
- Utilizing the features from DNNs [Feinman et al., 2017, Ma et al., 2018]: kernel density and local intrinsic dimensionality.

High-level idea: Measure the probability density of test sample on feature spaces of DNNs utilizing the concept of "generative" classifier.



$$P(f(\mathbf{x})|y) = \mathcal{N}(f(\mathbf{x})|\mu_y, \Sigma)$$
$$(\text{Confidence score}) = \log \mathcal{N}(f(\mathbf{x})|\mu_y, \Sigma)$$

- Applications: Detecting abnormal samples and class incremental learning.

Main idea: Generative classifiers from Softmax

Softmax classifier and generative classifier with GDA assumption.

- Suppose that a pre-trained Softmax neural classifier is given:

$$P(y = c|\mathbf{x}) = \frac{\exp\left(\mathbf{w}_c^\top f(\mathbf{x}) + b_c\right)}{\sum_{c'} \exp\left(\mathbf{w}_{c'}^\top f(\mathbf{x}) + b_{c'}\right)},$$ where f is penultimate layer.

- Gaussian discriminant analysis (GDA) with a tied covariance matrix:

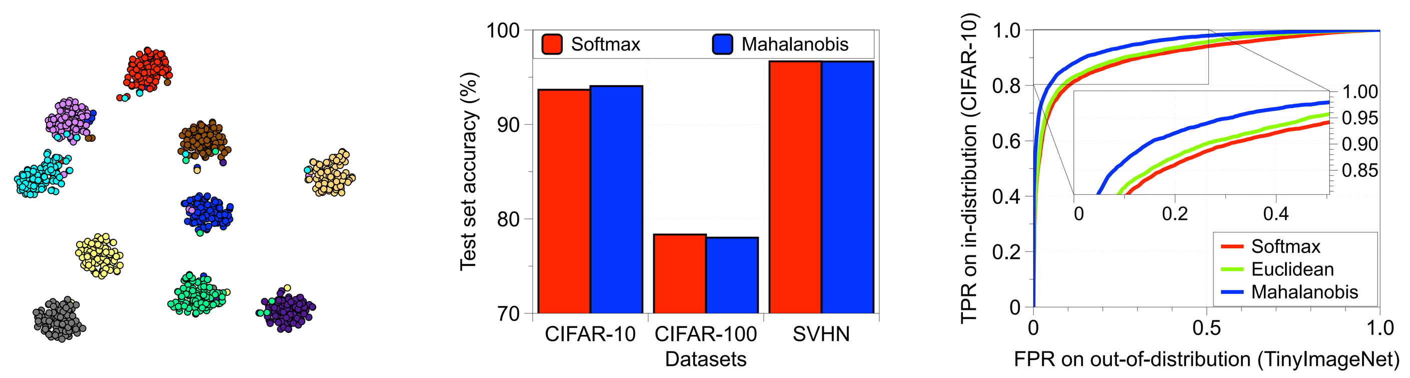
$$P(\mathbf{x}|y = c) = \mathcal{N}(\mathbf{x}|\mu_c, \Sigma), P(y = c) = \beta_c / \sum_{c'} \beta_{c'},$$

- It is well-known that posterior distribution of GDA corresponds to Softmax.

Idea: Obtaining generative classifier using pre-trained Softmax neural classifier.

- Estimating the parameters of it via empirical means and covariance:

$$\widehat{\mu}_c = \frac{1}{N_c} \sum_{i:y_i=c} f(\mathbf{x}_i), \quad \widehat{\mathbf{\Sigma}} = \frac{1}{N} \sum_c \sum_{i:y_i=c} \left(f(\mathbf{x}_i) - \widehat{\mu}_c\right)\left(f(\mathbf{x}_i) - \widehat{\mu}_c\right)^\top,$$



(a) Visualization by t-SNE (b) Classification accuracy (c) ROC curve

Figure 1: For all experiments in (a), (b) and (c), we commonly use the ResNet with 34 layers.

- Inference: $\widehat{y}(\mathbf{x}) = \arg\min_c \left(f(\mathbf{x}) - \widehat{\mu}_c\right)^\top \widehat{\mathbf{\Sigma}}^{-1} \left(f(\mathbf{x}) - \widehat{\mu}_c\right)$.

Contribution 1: New confidence score for detection

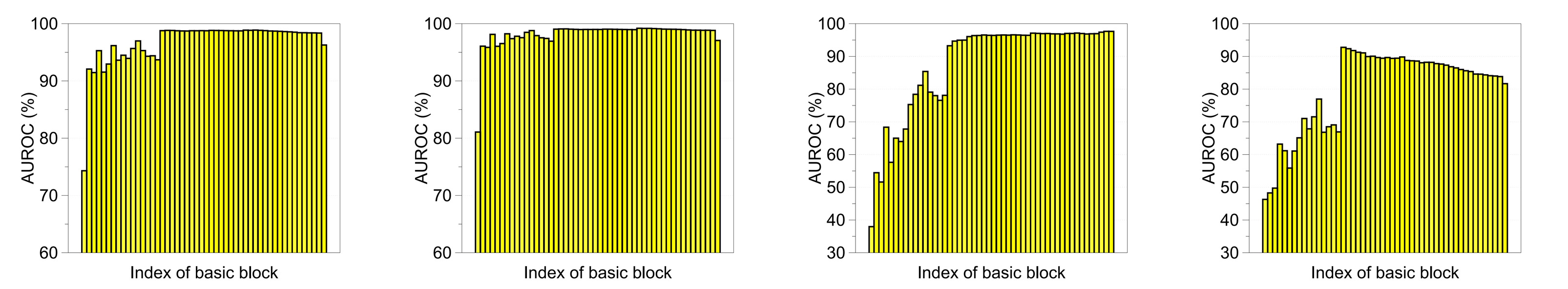
New confidence score: Mahalanobis distance between test sample and the closest class-conditional Gaussian distribution.

$$M(\mathbf{x}) = \max_c \; -\left(f(\mathbf{x}) - \widehat{\mu}_c\right)^\top \widehat{\mathbf{\Sigma}}^{-1} \left(f(\mathbf{x}) - \widehat{\mu}_c\right).$$

- Measuring the log of the probability densities of the test sample.

Calibration techniques: Input pre-processing and feature ensemble.

- Input pre-processing: adding a small controlled noise to a test sample.
- Feature ensemble: utilizing the low-level features in DNNs.



(a) TinyImageNet (b) LSUN (c) SVHN (d) DeepFool

Figure 2: AUROC (%) of threshold-based detector using the confidence score in (2) computed at different basic blocks of DenseNet trained on CIFAR-10 dataset. We measure the detection performance using (a) TinyImageNet, (b) LSUN, (c) SVHN and (d) adversarial (DeepFool) samples.

Algorithm description:

Algorithm 1 Computing the Mahalanobis distance-based confidence score.

Input: Test sample $\mathbf{x}$, weights of logistic regression detector α_ℓ, noise ε and parameters of Gaussian distributions $\{\widehat{\mu}_{\ell,c}, \widehat{\mathbf{\Sigma}}_\ell : \forall \ell, c\}$

Initialize score vectors: $\mathbf{M}(\mathbf{x}) = [M_\ell : \forall \ell]$
for each layer $\ell \in 1, \ldots, L$ **do**
  Find the closest class: $\widehat{c} = \arg\min_c \; (f_\ell(\mathbf{x}) - \widehat{\mu}_{\ell,c})^\top \widehat{\mathbf{\Sigma}}_\ell^{-1} (f_\ell(\mathbf{x}) - \widehat{\mu}_{\ell,c})$
  Add small noise to test sample: $\widehat{\mathbf{x}} = \mathbf{x} - \varepsilon \text{sign}\left(\bigtriangledown_{\mathbf{x}} \left(f_\ell(\mathbf{x}) - \widehat{\mu}_{\ell,\widehat{c}}\right)^\top \widehat{\mathbf{\Sigma}}_\ell^{-1} \left(f_\ell(\mathbf{x}) - \widehat{\mu}_{\ell,\widehat{c}}\right)\right)$
  Computing confidence score: $M_\ell = \max_c -\left(f_\ell(\widehat{\mathbf{x}}) - \widehat{\mu}_{\ell,c}\right)^\top \widehat{\mathbf{\Sigma}}_\ell^{-1} \left(f_\ell(\widehat{\mathbf{x}}) - \widehat{\mu}_{\ell,c}\right)$
end for
return Confidence score for test sample $\sum_\ell \alpha_\ell M_\ell$

- We set the weights by training logistic regression detector on validation samples

Contribution 2: Extension to incremental learning

Class incremental learning tasks: samples from new classes are added progressively to the pre-trained classifier.

Idea: Utilizing Mahalanobis distance-based score in class incremental learning.

Algorithm description:

Algorithm 2 Updating Mahalanobis distance-based classifier for class-incremental learning.

Input: set of samples from a new class $\{\mathbf{x}_i : \forall i = 1 \ldots N_{C+1}\}$, mean and covariance of observed classes $\{\widehat{\mu}_c : \forall c = 1 \ldots C\}, \widehat{\mathbf{\Sigma}}$

Compute the new class mean: $\widehat{\mu}_{C+1} \leftarrow \frac{1}{N_{C+1}} \sum_i f(\mathbf{x}_i)$
Compute the covariance of the new class: $\widehat{\mathbf{\Sigma}}_{C+1} \leftarrow \frac{1}{N_{C+1}} \sum_i (f(\mathbf{x}_i) - \widehat{\mu}_{C+1})(f(\mathbf{x}_i) - \widehat{\mu}_{C+1})^\top$
Update the shared covariance: $\widehat{\mathbf{\Sigma}} \leftarrow \frac{C}{C+1} \widehat{\mathbf{\Sigma}} + \frac{1}{C+1} \widehat{\mathbf{\Sigma}}_{C+1}$
return Mean and covariance of all classes $\{\widehat{\mu}_c : \forall c = 1 \ldots C+1\}, \widehat{\mathbf{\Sigma}}$

- Handling new class by simply computing the class mean of new class and updating the covariance.

Experimental result: detection and incremental learning

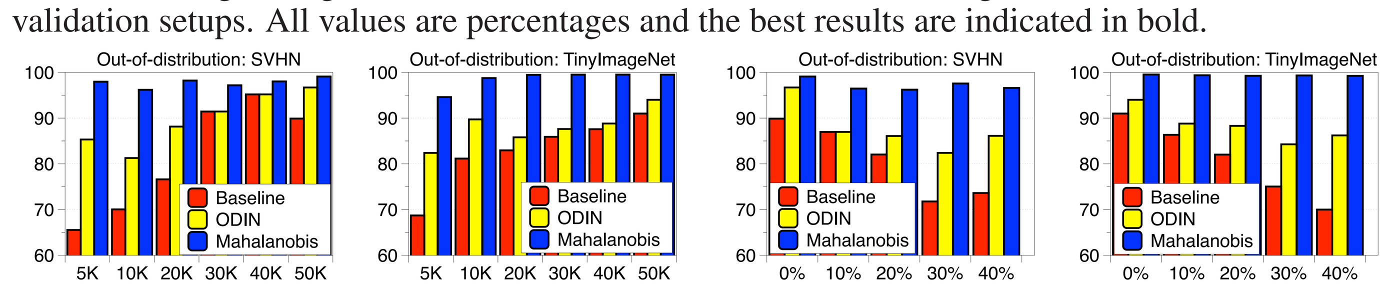
Experiments on detecting out-of-distribution (OOD) samples:

Method	Feature ensemble	Input pre-processing	TNR at TPR 95%	AUROC	Detection accuracy	AUPR in	AUPR out
Baseline [13]	-	-	32.47	89.88	85.06	85.40	93.96
ODIN [21]	-	-	86.55	96.65	91.08	92.54	98.52
Mahalanobis (ours)	-	-	54.51	93.92	89.13	91.56	95.95
	-	✓	92.26	98.30	93.72	96.01	99.28
	✓	-	91.45	98.37	93.55	96.43	99.35
	✓	✓	**96.42**	**99.14**	**95.75**	**98.26**	**99.60**

Table 1: Contribution of each proposed method on distinguishing in- and out-of-distribution test set data. We measure the detection performance using ResNet trained on CIFAR-10, when SVHN dataset is used as OOD. All values are percentages and the best results are indicated in bold.

In-dist (model)	OOD	Validation on OOD samples: TNR at TPR 95%	AUROC	Detection acc.	Validation on adversarial samples: TNR at TPR 95%	AUROC	Detection acc.
		Baseline [13] / ODIN [21] / Mahalanobis (ours)			Baseline [13] / ODIN [21] / Mahalanobis (ours)		
CIFAR-10 (DenseNet)	SVHN	40.2 / 86.2 / **90.8**	89.9 / 95.5 / **98.1**	83.2 / 91.4 / **93.9**	40.2 / 70.5 / **89.6**	89.9 / 92.8 / **97.6**	83.2 / 86.5 / **92.6**
	TinyImageNet	58.9 / 92.4 / **95.0**	94.1 / 98.5 / **98.8**	88.5 / 93.9 / **95.0**	58.9 / 87.1 / **94.9**	94.1 / 97.2 / **98.8**	88.5 / 92.1 / **95.0**
	LSUN	66.6 / 96.2 / **97.2**	95.4 / 99.2 / **99.3**	90.3 / 95.7 / **96.3**	66.6 / 92.9 / **97.2**	95.4 / 98.5 / **99.2**	90.3 / 94.3 / **96.2**
CIFAR-100 (DenseNet)	SVHN	26.7 / 70.6 / **82.5**	82.7 / 93.8 / **97.2**	75.6 / 86.6 / **91.5**	26.7 / 39.8 / **62.2**	82.7 / 88.2 / **91.8**	75.6 / 80.7 / **84.6**
	TinyImageNet	17.6 / 42.6 / **86.6**	71.7 / 85.2 / **97.4**	65.7 / 77.0 / **92.2**	17.6 / 43.2 / **87.2**	71.7 / 85.3 / **97.0**	65.7 / 77.2 / **91.8**
	LSUN	16.7 / 41.2 / **91.4**	70.8 / 85.5 / **98.0**	64.9 / 77.1 / **93.9**	16.7 / 42.1 / **91.4**	70.8 / 85.7 / **97.9**	64.9 / 77.3 / **93.8**
SVHN (DenseNet)	CIFAR-10	69.3 / 71.7 / **96.8**	91.9 / 91.4 / **98.9**	86.6 / 85.8 / **95.9**	69.3 / 69.3 / **97.5**	91.9 / 91.9 / **98.8**	86.6 / 86.6 / **96.3**
	TinyImageNet	79.8 / 84.1 / **99.9**	94.8 / 95.1 / **99.9**	90.2 / 90.4 / **98.9**	79.8 / 79.8 / **99.9**	94.8 / 94.8 / **99.8**	90.2 / 90.2 / **98.9**
	LSUN	77.1 / 81.1 / **100**	94.1 / 94.5 / **99.9**	89.1 / 89.2 / **99.3**	77.1 / 77.1 / **100**	94.1 / 94.1 / **99.9**	89.1 / 89.1 / **99.2**
CIFAR-10 (ResNet)	SVHN	32.5 / 86.6 / **96.4**	89.9 / 96.7 / **99.1**	85.1 / 91.1 / **95.8**	32.5 / 40.3 / **75.8**	89.9 / 86.5 / **95.5**	85.1 / 77.8 / **89.1**
	TinyImageNet	44.7 / 72.5 / **97.1**	91.0 / 94.0 / **99.5**	85.1 / 86.5 / **96.3**	44.7 / 69.6 / **95.5**	91.0 / 93.9 / **99.0**	85.1 / 86.0 / **95.4**
	LSUN	45.4 / 73.8 / **98.9**	91.0 / 94.1 / **99.7**	85.3 / 86.7 / **97.7**	45.4 / 70.0 / **98.1**	91.0 / 93.7 / **99.5**	85.3 / 85.8 / **97.2**
CIFAR-100 (ResNet)	SVHN	20.3 / 62.7 / **91.9**	79.5 / 93.9 / **98.4**	73.2 / 88.0 / **93.7**	20.3 / 12.2 / **41.9**	79.5 / 72.0 / **84.4**	73.2 / 67.7 / **76.5**
	TinyImageNet	20.4 / 49.2 / **90.9**	77.2 / 87.6 / **98.2**	70.8 / 80.1 / **93.3**	20.4 / 33.5 / **70.3**	77.2 / 83.6 / **87.9**	70.8 / 75.9 / **84.6**
	LSUN	18.8 / 45.6 / **90.9**	75.8 / 85.6 / **98.2**	69.9 / 78.3 / **93.5**	18.8 / 31.6 / **56.6**	75.8 / 81.9 / **82.3**	69.9 / 74.6 / **79.7**
SVHN (ResNet)	CIFAR-10	78.3 / 79.8 / **98.4**	92.9 / 92.1 / **99.3**	90.0 / 89.4 / **96.9**	78.3 / 79.8 / **94.1**	92.9 / 92.1 / **97.6**	90.0 / 89.4 / **94.6**
	TinyImageNet	79.0 / 82.1 / **99.9**	93.5 / 92.0 / **99.9**	90.4 / 89.4 / **99.1**	79.0 / 80.5 / **99.2**	93.5 / 92.9 / **99.3**	90.4 / 90.1 / **98.8**
	LSUN	74.3 / 77.3 / **99.9**	91.6 / 89.4 / **99.9**	89.0 / 87.2 / **99.5**	74.3 / 76.3 / **99.9**	91.6 / 90.7 / **99.9**	89.0 / 88.2 / **99.5**

Table 2: Distinguishing in- and out-of-distribution test set data for image classification under various validation setups. All values are percentages and the best results are indicated in bold.



(a) Small number of training data (b) Training data with random labels

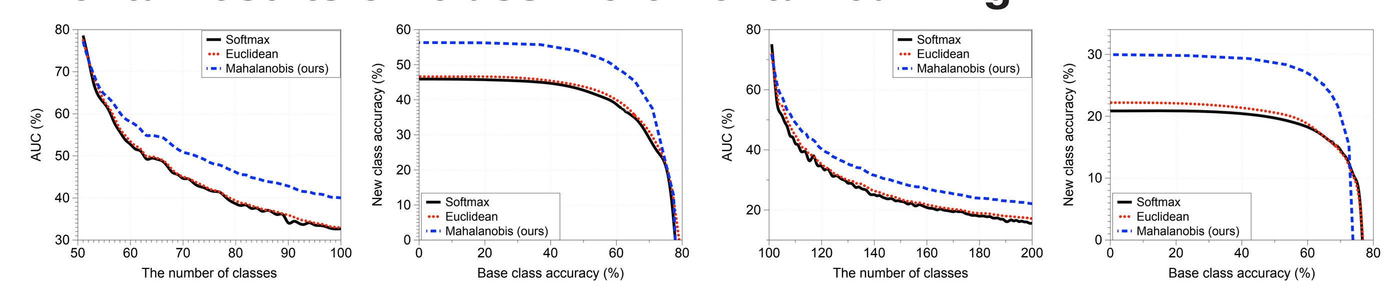
Figure 3: Comparison of AUROC (%) under extreme scenarios: (a) small number of training data, where the x-axis represents the number of training data. (b) Random label is assigned to training data, where the x-axis represents the percentage of training data with random label.

Experimental results on detecting adversarial attacks:

Model	Dataset (model)	Score	Detection of known attack: FGSM	BIM	DeepFool	CW	Detection of unknown attack: FGSM (seen)	BIM	DeepFool	CW
DenseNet	CIFAR-10	KD+PU [7]	85.96	96.80	68.05	58.72	85.96	3.10	68.34	53.21
		LID [22]	98.20	99.74	**85.14**	80.05	98.20	94.55	70.86	71.50
		Mahalanobis (ours)	**99.94**	**99.78**	83.41	**87.31**	**99.94**	**99.51**	**83.42**	**87.95**
	CIFAR-100	KD+PU [7]	90.13	89.69	68.29	57.51	90.13	66.86	65.30	58.08
		LID [22]	99.35	98.17	70.17	73.37	99.35	68.62	69.68	72.36
		Mahalanobis (ours)	**99.86**	**99.17**	**77.57**	**87.05**	**99.86**	**98.27**	**75.63**	**86.20**
	SVHN	KD+PU [7]	86.95	82.06	89.51	85.68	86.95	83.28	84.38	82.94
		LID [22]	99.35	94.87	91.79	94.70	99.35	92.21	80.14	85.09
		Mahalanobis (ours)	**99.85**	**99.28**	**95.10**	**97.03**	**99.85**	**99.12**	**93.47**	**96.95**
ResNet	CIFAR-10	KD+PU [7]	81.21	82.28	81.07	55.93	83.51	16.16	76.80	56.30
		LID [22]	99.69	96.28	88.51	82.23	99.69	95.38	71.86	77.53
		Mahalanobis (ours)	**99.94**	**99.57**	**91.57**	**95.84**	**99.94**	**98.91**	**78.06**	**93.90**
	CIFAR-100	KD+PU [7]	89.90	83.67	80.22	77.37	89.90	68.85	57.78	73.72
		LID [22]	98.73	96.89	71.95	78.67	98.73	55.82	63.15	75.03
		Mahalanobis (ours)	**99.77**	**96.90**	**85.26**	**91.77**	**99.77**	**96.38**	**81.95**	**90.96**
	SVHN	KD+PU [7]	82.67	66.19	89.71	76.57	82.67	43.21	**84.30**	67.85
		LID [22]	97.86	90.74	92.40	88.24	97.86	84.88	67.28	76.58
		Mahalanobis (ours)	**99.62**	**97.15**	**95.73**	**92.15**	**99.62**	**95.39**	72.20	**86.73**

Table 3: Comparison of AUROC (%) under various validation setups. For evaluation on unknown attack, FGSM samples denoted by "seen" are used for validation. For our method, we use both feature ensemble and input pre-processing. The best results are indicated in bold.

Experimental results on class incremental learning:



(a) Base: half of CIFAR-100 / New: the other half (b) Base: CIFAR-100 / New: ImageNet

Figure 4: Experimental results of class-incremental learning on CIFAR-100 and ImageNet datasets. In each experiment, we report (left) AUC with respect to the number of learned classes and, (right) the base-new class accuracy curve after the last new classes is added.