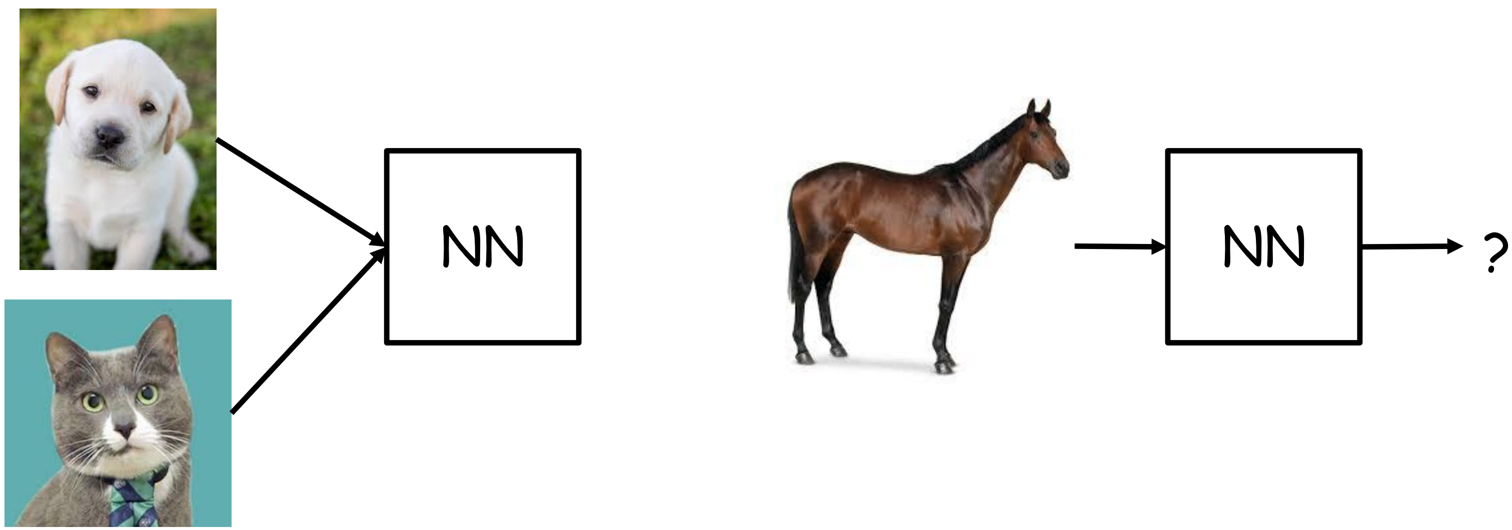
Out-of-Distribution Detection

Jee-Hyong Lee

Sungkyunkwan Univ.

Introduction

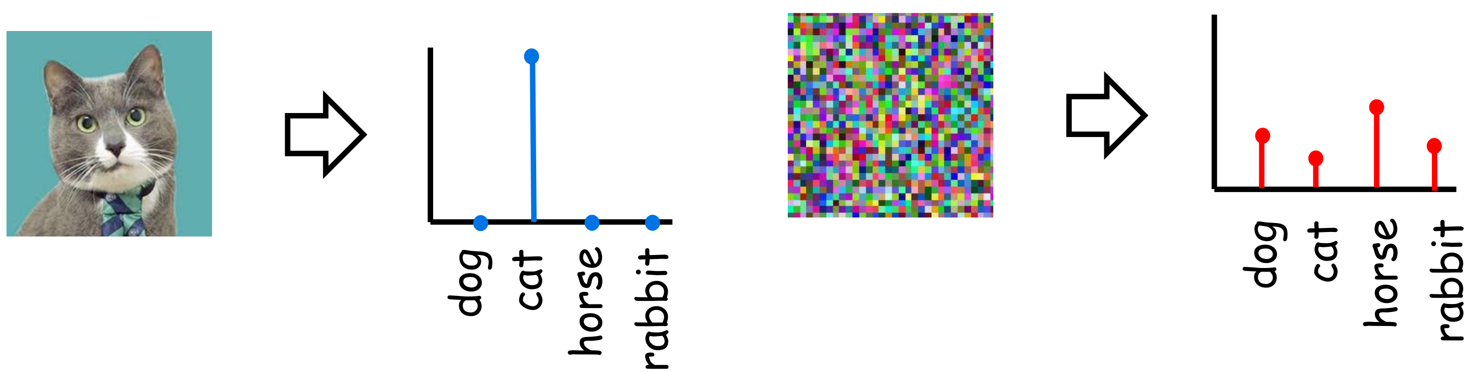
- **In the World, there are many instances which we never expect they are given.**
  - Does deep neural network can say “I don’t know” ?



Out of distribution

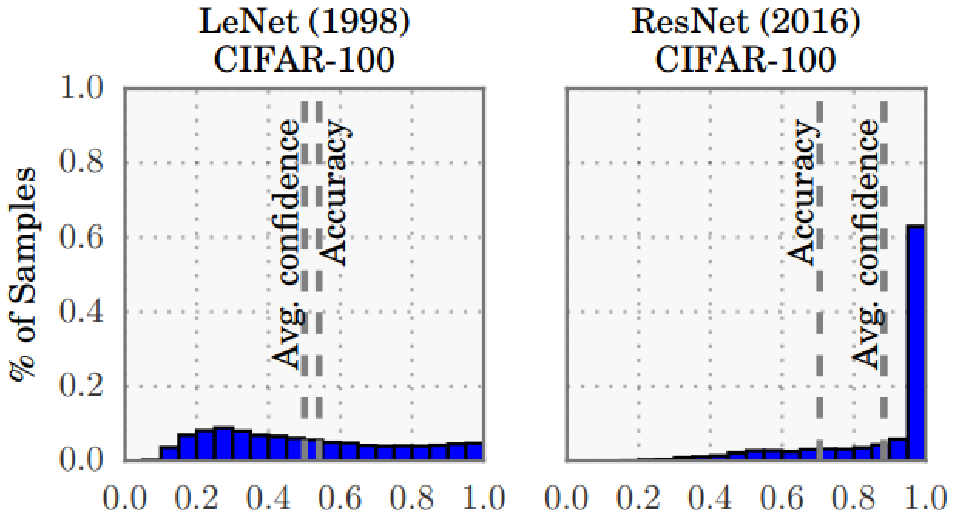
Introduction

- **A Simple Way**
  - We choose the maximum of softmax for classification
    - For an image in domain, softmax will produce a sharp output
    - For an image out of domain, softmax will produce rather a vague output
  - Let's check the value of the maximum



Introduction

- **Over Confidence**
  - Modern NN tends to output overconfident prediction
    - Confidence : Max softmax probability
  - NN returns prediction with high confidence for noise image

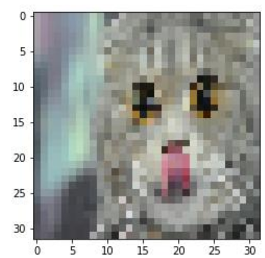


Guo, Chuan, et al. "On calibration of modern neural networks." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

Introduction

- **Over Confidence**
  - ResNet-20 trained on CIFAR10 (Test Acc: 92%)
  - Prediction & Activation before softmax

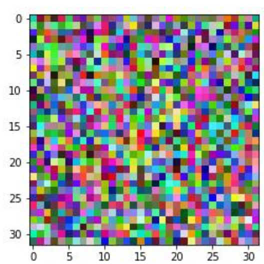
In-domain



Cat: 87%

```
array([-11.121608 , -12.295707 , -1.5396624 , 1.8473705 ,
       -4.0719457 , -0.40232527, -4.8595014 , -9.229726 ,
       -7.4466705 , -11.751272 ], dtype=float32)
```

Out-of-domain



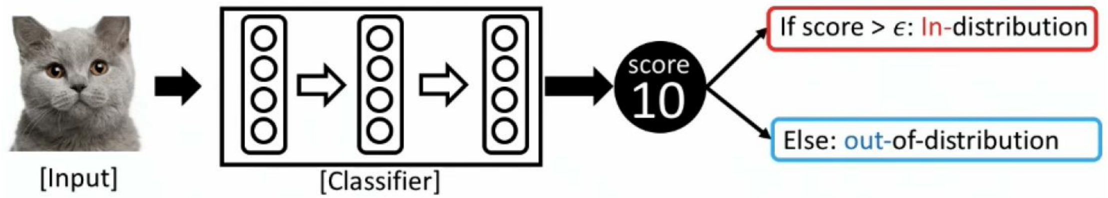
Bird: 84%

Over-confident prediction

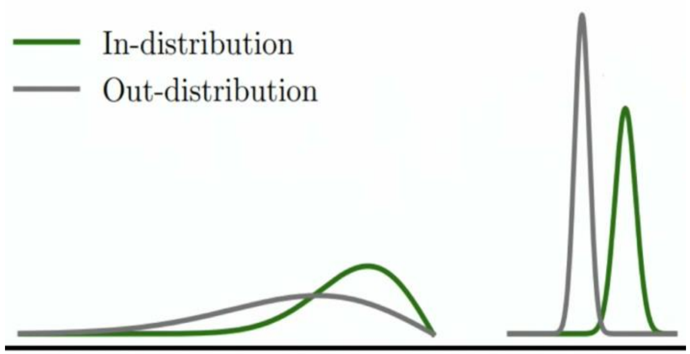
```
array([ -8.550764 , -0.03473853, 2.1666217 , -0.5177511 ,
       -9.423397 , -11.470142 , -5.384335 , -11.936867 ,
       -8.519983 , -6.6835756 ], dtype=float32)
```

Approaches

- **Threshold-based Detection**

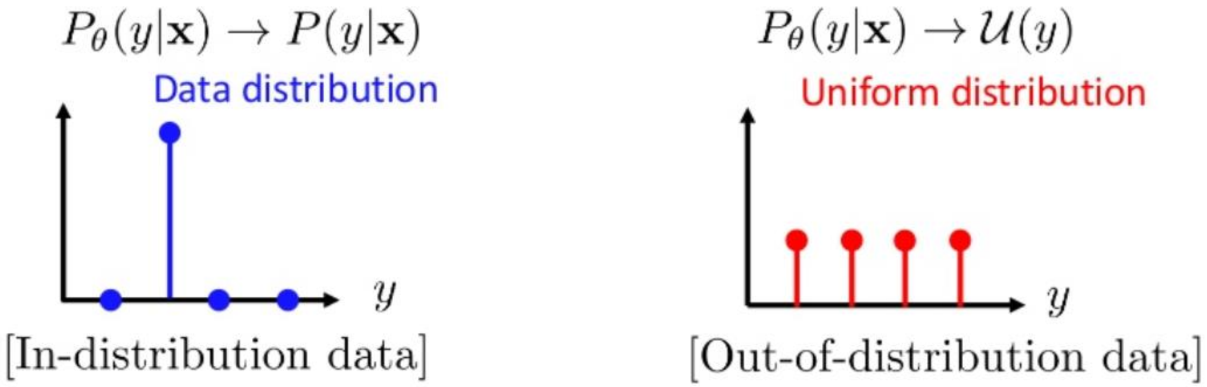


  - Limitations: Performance of prior works highly depends on how to train the classifiers



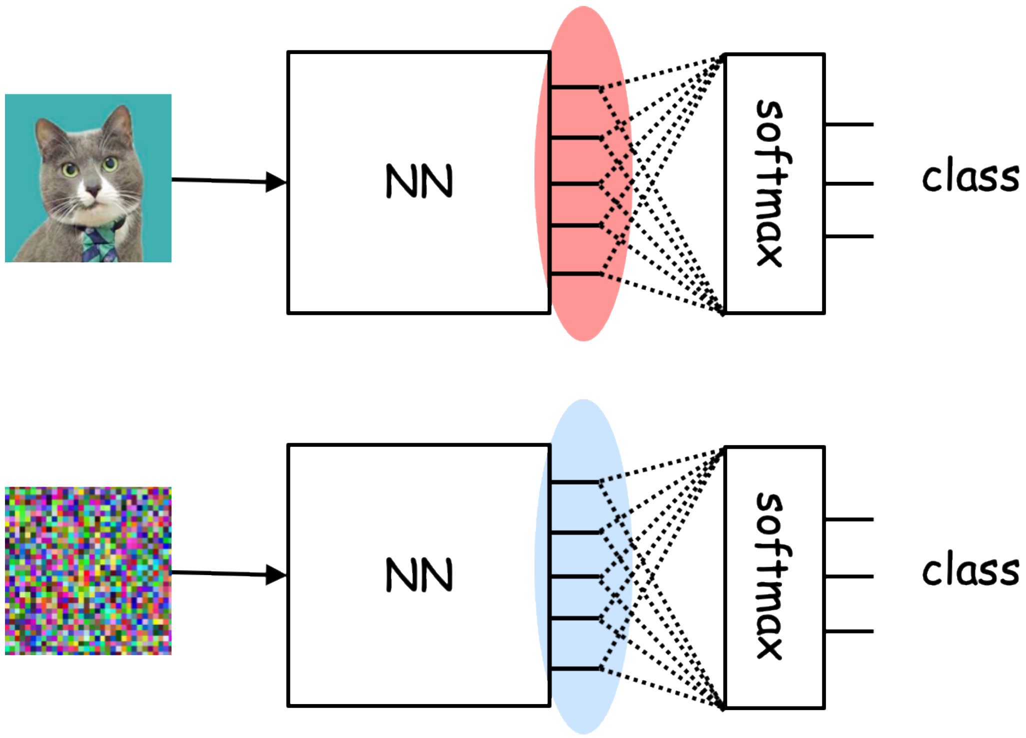
Approaches

- **Confidence Calibration**
  - Specially train a neural network so that it has low confidence for out-of-distribution samples



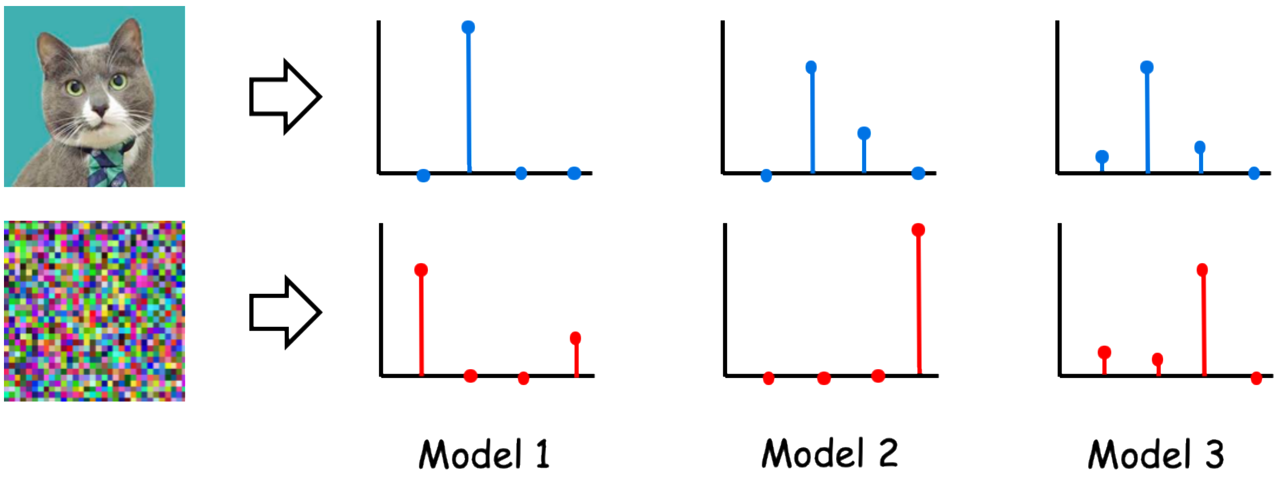
Approaches

- **Distribution-based Detection**
  - Output of each layer may be different between in-distribution and out-of-distribution samples



Approaches

- **Variance-based Detection**
  - NN cannot perform extrapolation as much as interpolation
  - Check the variance of output
    - Similar output for in-distribution data
    - Different output for out-of-distribution data



Approaches

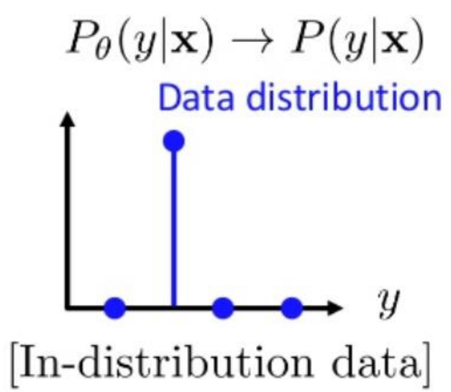
- **Confidence Calibration**
  - K. Lee, et al., Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples, ICLR 2018
- **Distribution-based Detection**
  - K. Lee, et al., A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks, NIPS 2018
- **Variance-based Detection**
  - B. Lakshminarayanan, et al., Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles, NIPS 2017

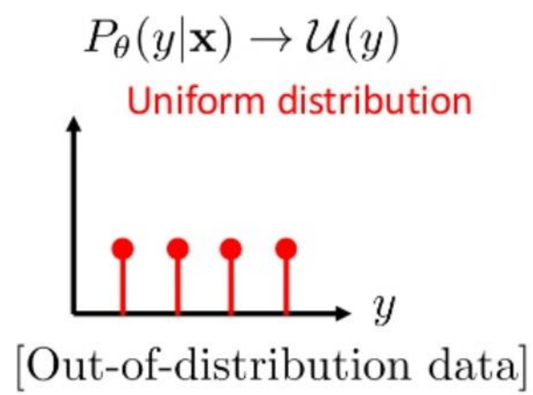
Approaches

- **Confidence Calibration**
  - K. Lee, et al., Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples, ICLR 2018
- **Distribution-based Detection**
  - K. Lee, et al., A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks, NIPS 2018
- **Variance-based Detection**
  - B. Lakshminarayanan, et al., Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles, NIPS 2017

Confidence Calibration

- **Calibrate confidence**
  - Train a neural network so that it outputs higher maximum prediction values to in-distribution samples than out-of-distribution ones
  - We need out-of-distribution data when training





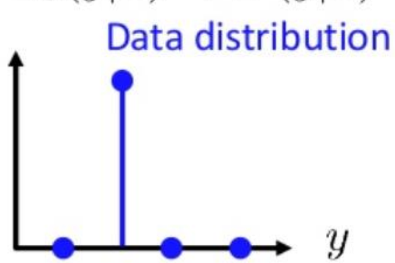
Confidence Calibration

- **Confident Loss**
  - Minimize the KL divergence on data from out-of-distribution

Uniform distribution

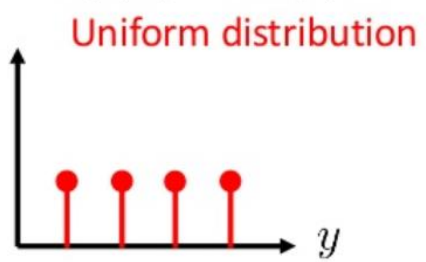
$$\min_{\theta} \; \underbrace{\mathbb{E}_{P_{\text{in}}(\widehat{\mathbf{x}},\widehat{y})}}_{\text{Data from in-distribution}} \left[-\log P_\theta \left(y = \widehat{y} | \widehat{\mathbf{x}}\right) \right] + \beta \underbrace{\mathbb{E}_{P_{\text{out}}(\mathbf{x})}}_{\text{Data from out-of-distribution}} \left[KL\left(\overline{\mathcal{U}}\left(y\right) \parallel P_\theta\left(y|\mathbf{x}\right)\right) \right]$$

$P_\theta(y|\mathbf{x}) \rightarrow P(y|\mathbf{x})$



[In-distribution data]

$P_\theta(y|\mathbf{x}) \rightarrow \mathcal{U}(y)$

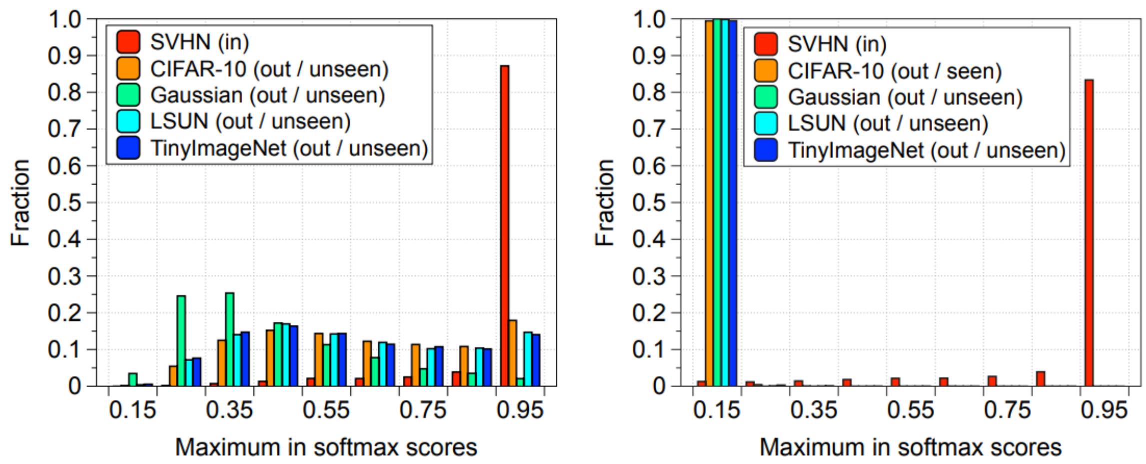


[Out-of-distribution data]

Confidence Calibration

▪ Confident Loss

– Simple test



(a) Cross entropy loss (b) Confidence loss in (1)

- Model: 2 conv + 3 FC
- Train data: SVHN(in-dist), MNIST(out-of-dist)

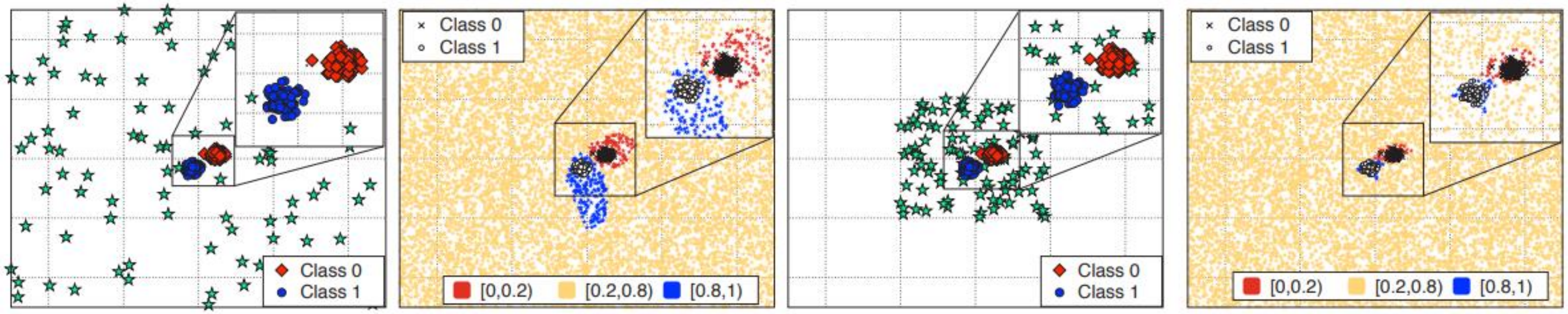
Confidence Calibration

- **Confidence Loss**
  - Usually given out-of-distribution data is not enough to generally model out-of-distribution samples
  - We need more out-of-distribution samples

- **What about generating out-of-distribution data with GAN?**

Confidence Calibration

- **Generating out-of-distribution data with GAN**
  - (a) & (b) out-of-distribution data is sparse around in-dist.
  - (c) & (d) out-of-distribution data is dense around in-dist.



(a) (b) (c) (d)

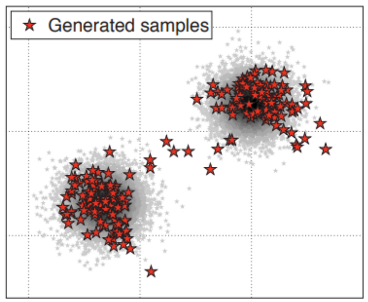
Red & blue : in-distribution data
Green : out-of-distribution data
Yellow: synthetic out-of-distribution data

=> We need to densely generate synthetic OOD around ID

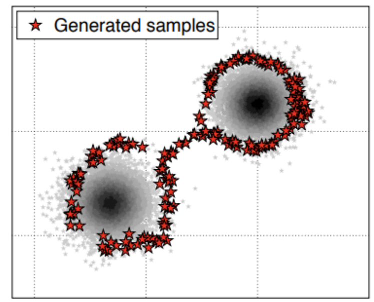
Confidence Calibration

- **Generating out-of-distribution data with GAN**

In-distribution



In-distribution
(on border line)



Confidence Calibration

- **Generating out-of-distribution data with GAN**
  - GAN loss to generate synthetic in-distribution samples on border lines

Output of classifier
(Need to be trained)

$$\min_G \max_D \; \beta \underbrace{\mathbb{E}_{P_G(\mathbf{x})}\Big[KL\left(\mathcal{U}\left(y\right) \,\|\, P_\theta\left(y|\mathbf{x}\right)\right)\Big]}_{\text{(a)}}$$

$$+ \underbrace{\mathbb{E}_{P_{\text{in}}(\mathbf{x})}\Big[\log D\left(\mathbf{x}\right)\Big] + \mathbb{E}_{P_G(\mathbf{x})}\Big[\log\left(1 - D\left(\mathbf{x}\right)\right)\Big]}_{\text{(b)}}$$

(a) Forces the generator to generate low-density samples
(b) Original GAN loss

Confidence Calibration

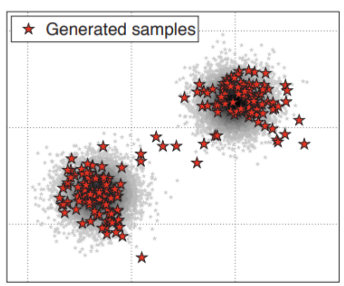
- **Generating out-of-distribution data with GAN**
  - GAN loss to generate synthetic in-distribution samples on border lines

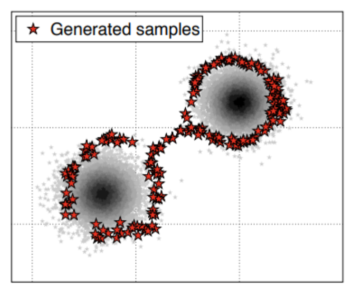
Output of classifier
(Need to be trained)

$$\min_{G}\max_{D}\ \beta\,\underbrace{\mathbb{E}_{P_G(\mathbf{x})}\left[KL\left(\mathcal{U}\left(y\right)\,\|\,P_\theta\left(y|\mathbf{x}\right)\right)\right]}_{\text{(a)}}$$

$$+\underbrace{\mathbb{E}_{P_{\text{in}}(\mathbf{x})}\left[\log D\left(\mathbf{x}\right)\right]+\mathbb{E}_{P_G(\mathbf{x})}\left[\log\left(1-D\left(\mathbf{x}\right)\right)\right]}_{\text{(b)}}$$

Original GAN loss





New GAN loss

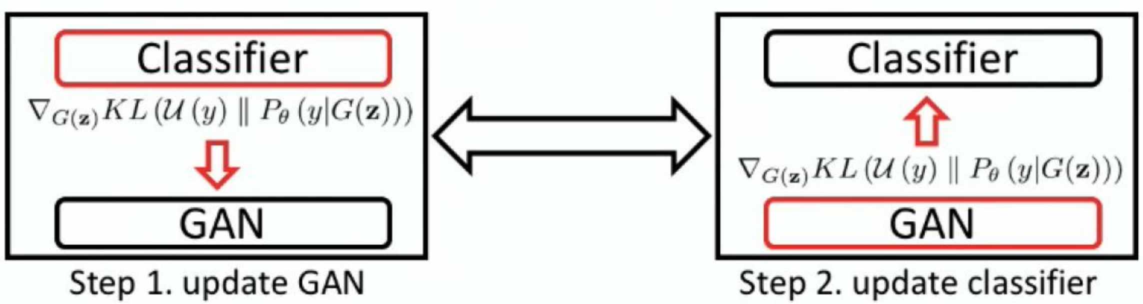
Confidence Calibration

- **Joint Loss: Confidence Loss + GAN Loss**

$$\min_G \max_D \min_\theta \quad \underbrace{\mathbb{E}_{P_{\text{in}}(\widehat{\mathbf{x}},\widehat{y})}\Big[-\log P_\theta\left(y=\widehat{y}|\widehat{\mathbf{x}}\right)\Big]}_{\text{(c)}} + \beta\underbrace{\mathbb{E}_{P_G(\mathbf{x})}\Big[KL\left(\mathcal{U}\left(y\right)\parallel P_\theta\left(y|\mathbf{x}\right)\right)\Big]}_{\text{(d)}}$$

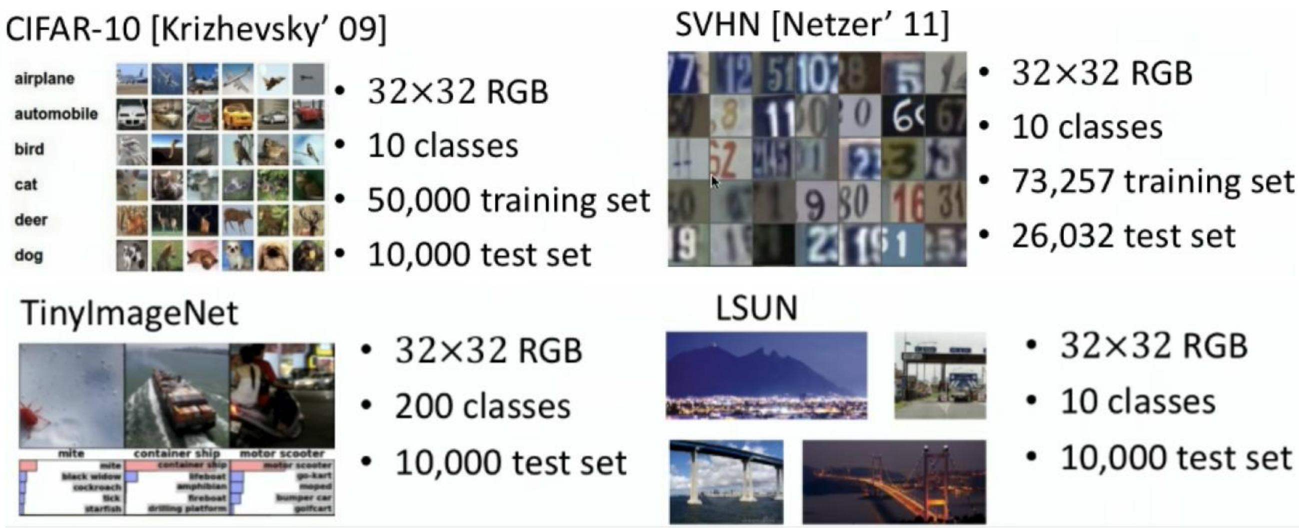
$$\underbrace{+\mathbb{E}_{P_{\text{in}}(\widehat{\mathbf{x}})}\Big[\log D\left(\widehat{\mathbf{x}}\right)\Big] + \mathbb{E}_{P_G(\mathbf{x})}\Big[\log\left(1-D\left(\mathbf{x}\right)\right)\Big]}_{\text{(e)}}.$$

  - Classifier's confidence loss: (c) + (d)
  - GAN loss: (d) + (e)



Confidence Calibration

- **Experiment**
  - Data set



  - Used model: VGGNet

Confidence Calibration

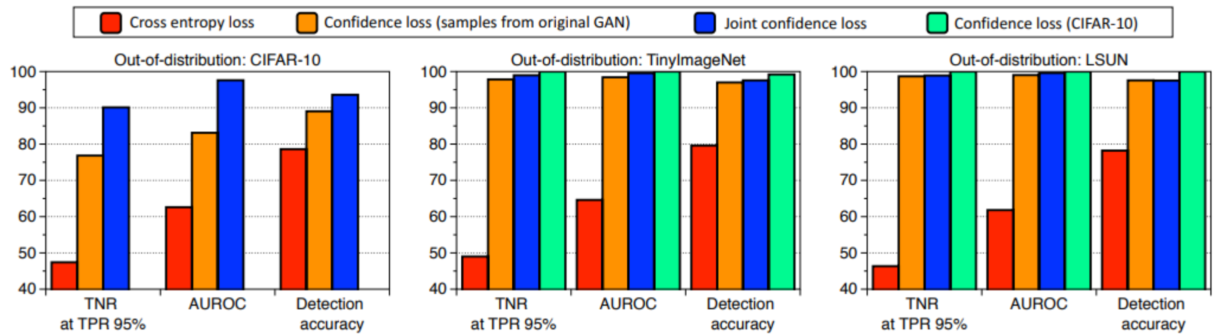
- **Experiment**
  - Result with Confident loss (without GAN loss)

In-dist	Out-of-dist	Classification accuracy	TNR at TPR 95%	AUROC	Detection accuracy	AUPR in	AUPR out
		Cross entropy loss / Confidence loss					
SVHN	CIFAR-10 (seen)	93.82 / **94.23**	47.4 / **99.9**	62.6 / **99.9**	78.6 / **99.9**	71.6 / **99.9**	91.2 / **99.4**
	TinyImageNet (unseen)		49.0 / **100.0**	64.6 / **100.0**	79.6 / **100.0**	72.7 / **100.0**	91.6 / **99.4**
	LSUN (unseen)		46.3 / **100.0**	61.8 / **100.0**	78.2 / **100.0**	71.1 / **100.0**	90.8 / **99.4**
	Gaussian (unseen)		56.1 / **100.0**	72.0 / **100.0**	83.4 / **100.0**	77.2 / **100.0**	92.8 / **99.4**
CIFAR-10	SVHN (seen)	80.14 / **80.56**	13.7 / **99.8**	46.6 / **99.9**	66.6 / **99.8**	61.4 / **99.9**	73.5 / **99.8**
	TinyImageNet (unseen)		**13.6** / 9.9	**39.6** / 31.8	**62.6** / 58.6	**58.3** / 55.3	**71.0** / 66.1
	LSUN (unseen)		**14.0** / 10.5	**40.7** / 34.8	**63.2** / 60.2	**58.7** / 56.4	**71.5** / 68.0
	Gaussian (unseen)		2.8 / **3.3**	10.2 / **14.1**	50.0 / 50.0	48.1 / **49.4**	39.9 / **47.0**

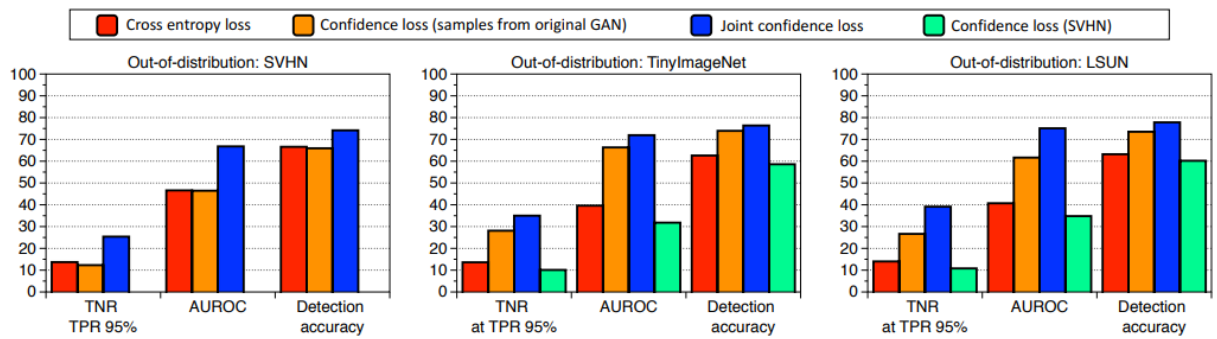
Sometimes good but sometime bad

Confidence Calibration

- **Experiment**
  - Result with Joint Loss (with GAN Loss)



(a) In-distribution: SVHN



(b) In-distribution: CIFAR-10