OUTLIER EXPOSURE WITH CONFIDENCE CONTROL FOR OUT-OF-DISTRIBUTION DETECTION

Aristotelis-Angelos Papadopoulos & Mohammad Reza Rajati & Nazim Shaikh & Jiamian Wang
University of Southern California
Los Angeles, CA 90089, USA
{aristop,rajati,nshaikh,jiamianw}@usc.edu

[Submitted on 8 Jun 2019 (v1), last revised 5 Jun 2020 (this version, v3)]

https://arxiv.org/abs/1906.03509

- 1. Intro
- 2. Related works
- 3. Methodology
- 4. Experiments and Results
- 5. Conclusion

Intro

Anomaly Detection의 필요성

Anomaly Detection Task는 보편적인 패턴(representation into train-set)에서 벗어나는 패턴(Outlier)을 감지하여

-> 전체 시스템의 Totally Performance를 높이는데 이용

Intro

OoD(Out of Distribution)

- $P_{in}(x,y)$로 부터 먼 거리에 존재하는 sample의 Distribution

  $input\ x \in X, lable\ y \in \{1, \ldots, K\}$

- 특정 학습시점(time step t시점)에 보유하고 있는 In-distribution 外 나머지 모든 sample에서 distance가 먼(outbound)

  \+ 시인성이 있는 sample

  \+ 새로운 class로 분류

Solving the OoD(Out of Distribution) task

- Train-time : 보유하고 있는 In-distribution을 이용하여, multi-classification 모델 학습
- Inference-time : Unseen에 대하여 두 가지 goal을 만족
  - In-distribution class를 기존 defined class로 정확히 분류
  - (시인성이 있는) Out-of-distribution class(new class, Unknown class) 정확히 분류 - reject

-> Open-set에 대하여 전체(Totally) AI 시스템(classifier)의 Continuous Performance를 높이는데 이용

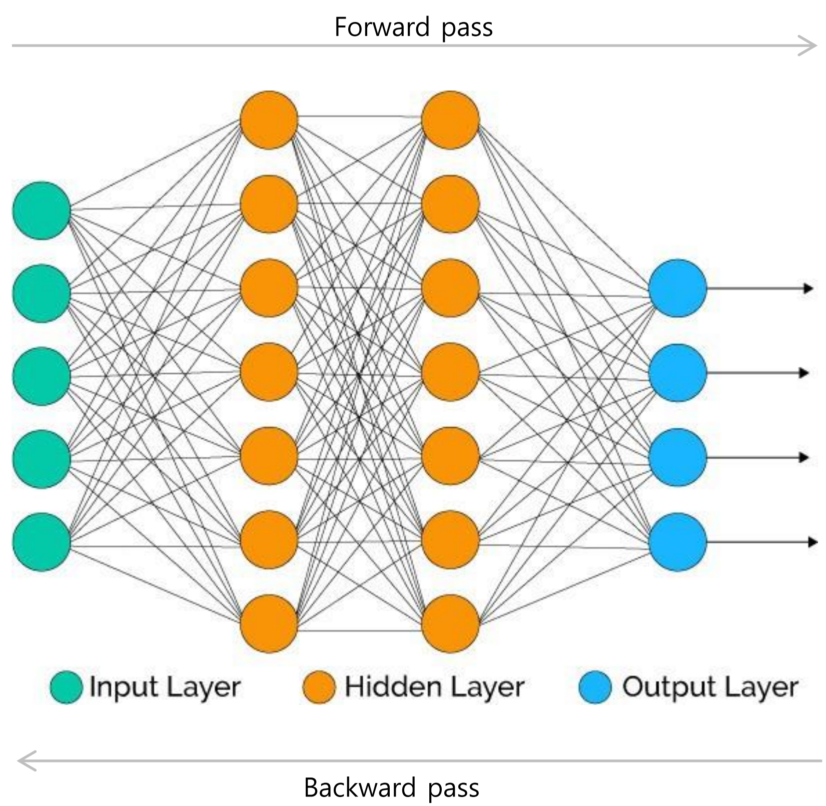
Intro

Outlier Exposure(OE)[2019] method

- Training time에 Out-of-Distribution sample을 같이 노출 -> Anomaly Detection -> Increase Model Performance !

D_{in}

D_{out}^{OE}



Intro

Confidence Score

- Entropy가 낮은 NN의 Output status(~ In Distribution)

Over Confident

- Training-set(~ In Distribution)에서 N개의 class 중 특정 1개의 Known class 높은 Prediction Probability로 분류

Confident Penalty

- Training-set에 대해 Over Confident한 Model(Over-fit)을 Regularization 하는 Term

OOD Score

- Confidence Score의 반대

-> Confidence Score가 Over Confident 한 모델에 Confident Penalty 하여
Entorpy가 높은 sample에 대하여 OOD하는 것이 바로 = Confidence Control 스킬

Intro

Skill (1) + Skill (2)

OUTLIER EXPOSURE WITH CONFIDENCE CONTROL FOR OUT-OF-DISTRIBUTION DETECTION

Solving the problem !

Aristotelis-Angelos Papadopoulos & Mohammad Reza Rajati & Nazim Shaikh & Jiamian Wang
University of Southern California
Los Angeles, CA 90089, USA
{aristop,rajati,nshaikh,jiamianw}@usc.edu

[Submitted on 8 Jun 2019 (v1), last revised 5 Jun 2020 (this version, v3)]

https://arxiv.org/abs/1906.03509

Related works

A Baseline for Detecting Misclassified and OOD in NN ICLR, 2017	Training Confidence-Calibrated Classifiers for Detecting OOD ICLR, 2018	Deep Anomaly Detection with OE ICLR, 2019
• In-Distribution vs. OOD를 판단하는 거리를 구하는 Softmax Prediction 확률을 제안 • **MSP(Maximum Softmax Prediction)** - Error and Success Prediction - In-and out-of-distribution	• **KLD 기반의 loss function** 제안 • Sample들 간 거리 - output distribution given by softmax ~ **GAN**에서 생성된 샘플들의 Uniform Distribution 간의 거리	• 최초의 **Outlier exposure** 제안 - Outlier를 사전에 정의 및 보유해야한다는 가정이 필요) • OE방식은 model의 calibration을 증가 시킴

Methodology

Objective Function

y : class given input x, L_{CE}: cross − entropy function, K: num of classes in D_{in}

z : vector representation of example $x^{(i)}$, A_{tr}: training accuracy

$$minimize_{\theta}\ E_{(x,y)\sim D_{in}}[L_{CE}(f_{\theta}(x), y)]$$

$$subject\ to\quad E_{x\sim D_{in}}\left[\max_{l=1,\dots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right] = A_{tr}$$

$$\max_{l=1,\dots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right) = \frac{1}{K'}\ \forall x^{(i)} \sim D_{out}^{OE}$$

Methodology

Objective Function

y : *class given input* x, L_{CE}: *cross − entropy function*, K: *num of classes in* D_{in}

z : *vector representation of example* $x^{(i)}$, A_{tr}: *training accuracy*

$$minimize_{\theta}\ E_{(x,y)\sim D_{in}}[L_{CE}(f_{\theta}(x), y)]$$

$$subject\ to\quad E_{x\sim D_{in}}\left[\max_{l=1,\ldots k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right] = A_{tr}$$

class(D_{in}) confident score = training accuracy

$$\max_{l=1,\ldots k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right) = \frac{1}{K'},\ \forall x^{(i)}\sim D_{out}^{OE}$$

unknown class의 sample(D_{out})에 대해서는
NN이 uncertain(=output으로
Uniform-distribution출력)

Methodology

Objective Function

y : *class given input x*, L_{CE}: *cross − entropy function*, K: *num of classes in* D_{in}

z : *vector representation of example* $x^{(i)}$, A_{tr}: *training accuracy*, λ_2: *Lagrange multiplier*

$$minimize_{\theta} \quad E_{(x,y)\sim D_{in}}[L_{CE}(f_\theta(x), y)]$$

$$+\lambda_1 \left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\ldots k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right]\right)$$

$$+\lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}} \left(\frac{1}{K} - \max_{l=1,\ldots k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right)$$

Methodology

Objective Function

y : class given input x, L_{CE}: cross − entropy function, K: num of classes in D_{in}

z : vector representation of example $x^{(i)}$, A_{tr}: training accuracy, λ_2: Lagrange multiplier

$$\text{minimize}_{\theta} \quad E_{(x,y)\sim D_{in}}[L_{CE}(f_\theta(x), y)]$$

$$+\lambda_1 \left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\dots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right]\right)$$

-이번 변형은 특별한 케이스로,
모든 $x^{(i)} \sim D_{out}^{OE}$에 대해 같은
Lagrange multiplier(λ2)를 범용적으로
사용한다.

$$+\lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}} \left(\frac{1}{K} - \max_{l=1,\dots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}} \right) \right)$$

-Validation technique를 통해 알맞은 λ1 , λ2 값 을
구한다.

Methodology

Objective Function

y : class given input x, L_{CE}: cross − entropy function, K: num of classes in D_{in}

z : vector representation of example $x^{(i)}$, A_{tr}: training accuracy, λ_2: Lagrange multiplier

$$\text{minimize}_{\theta} \quad E_{(x,y)\sim D_{in}}[L_{CE}(f_{\theta}(x), y)]$$

$$+\lambda_1 \left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\dots,k} \left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}} \right) \right] \right)$$

-Training Accuracy와 average confidence 의 차이를 최소화

$$+\lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}} \left(\frac{1}{K} - \max_{l=1,\dots,k} \left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}} \right) \right)$$

-Uniform distribution과 softmax layer를 통해 얻은 distribution 차이 를 최소화

Methodology

Objective Function

y : *class given input* x, L_{CE}: *cross − entropy function*, K: *num of classes in* D_{in}

z : *vector representation of example* $\mathrm{x}^{(\mathrm{i})}$, A_{tr}: *training accuracy*, λ_2: *Lagrange multiplier*

$$\mathit{minimize}_{\theta} \quad E_{(x,y)\sim D_{in}}[L_{CE}(f_\theta(x), y)]$$

$$+\lambda_1\left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\ldots k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right]\right)$$

$$+\boxed{\lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}}\left(\frac{1}{K} - \max_{l=1,\ldots k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right)} \quad \cdots\cdots\cdots\cdots \quad (2)$$

Methodology

Objective Function

y : class given input x, L_{CE}: cross − entropy function, K: num of classes in D_{in}

z : vector representation of example $x^{(i)}$, A_{tr}: training accuracy, λ_2: Lagrange multiplier

Objective function의 Regularization Term이

$$\boxed{\lambda_2 \sum_{x^{(i)} \sim D_{out}^{OE}} \left(\underbrace{\frac{1}{K} - \max_{l=1,\dots k} \left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}} \right)}_{} \right)} \quad \text{(2)}$$

음수가 되지 않게 하고
&&
최소화(=0) 하려면,

-> 모든 class를 동시에 1/k로 조절(regularization) 해야 함

Methodology

Objective Function

y : *class given input* x, L_{CE}: *cross − entropy function*, K: *num of classes in* D_{in}

z : *vector representation of example* $\mathrm{x}^{(\mathrm{i})}$, A_{tr}: *training accuracy*, λ_2: *Lagrange multiplier*

$$minimize_{\theta} \quad E_{(x,y)\sim D_{in}}[L_{CE}(f_{\theta}(x), y)]$$

$$+\lambda_1 \left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\dots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right) \right] \right)^2$$

$$+\lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}} \sum_{l=1}^{K} \underline{\left| \frac{1}{K} - \frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}} \right|}$$

Methodology

Objective Function

y : class given input x, L_{CE}: cross − entropy function, K: num of classes in D_{in}

z : vector representation of example $x^{(i)}$, A_{tr}: training accuracy, λ_2: Lagrange multiplier

$$minimize_{\theta} \quad E_{(x,y)\sim D_{in}}[L_{CE}(f_\theta(x),y)]$$

$$+\lambda_1\left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\ldots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right]\right)^2$$

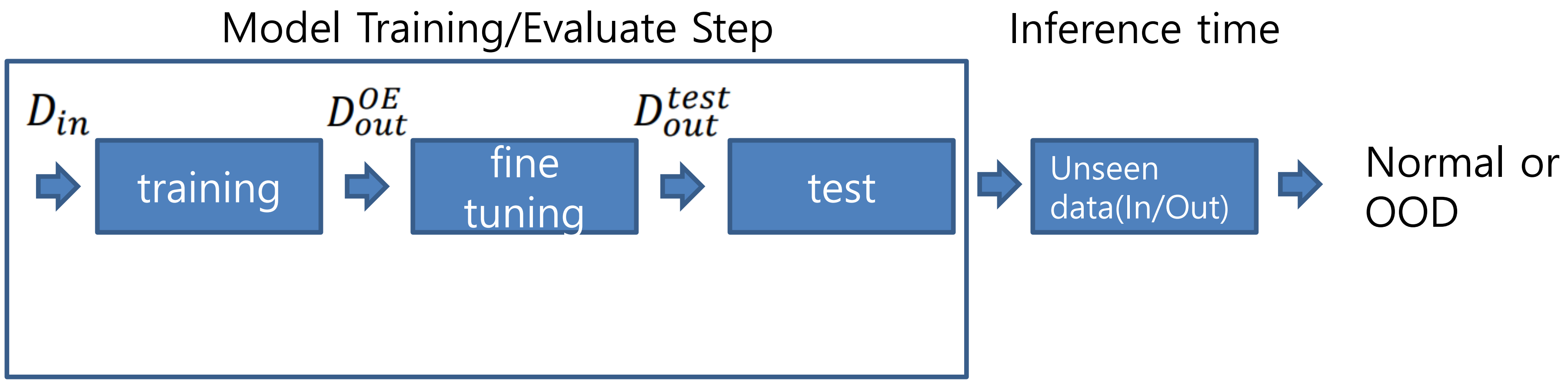
A_{tr}과 멀리 떨어진 max softmax 확률값을 강조
-> better detect

$$+\lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}} \sum_{l=1}^{K} \left|\frac{1}{K} - \frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right|$$

$l1$norm이 모든 prediction probabilities를 균등하게
$1/K$ 로 reguralrization (Uniform distribution 화)

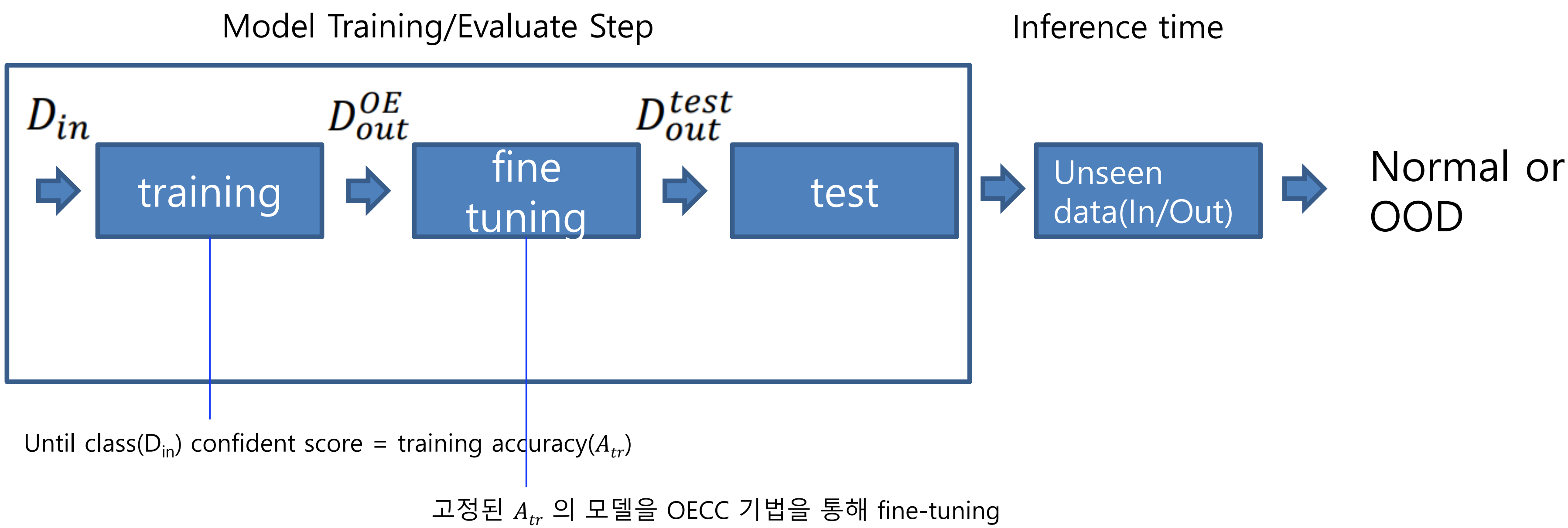
Methodology

Training



Methodology

Training



Experiments and Results

Metric

1. FPRN(False Positive Rate at N% True Positive Rate:

✓ Maximum Softmax Probability threshold가 특정 값으로 정해져 있을 때의 OOD Detector의 성능 N%의 OOD Sample이 감지되어야 한다고 가정하고 threshold를 지정.

✓ 이때의 threshold로 FPR(실제로는 In-distribution이지만 OOD로 잘못 판단한 비율) 계산

2. AUROC(Area Under the Receiver Operating Characteristic Curve):

✓ threshold 값을 다르게 할 때의 각각의 OOD detector의 성능을 표현

3. AUPR(Area Under the precision-Recall Curve):

✓ OOD와 In-distribution간의 imbalance가 있을 때 모델 성능 측정

Dataset

✓ D_{in} 분포에서 만들어진 example: 'In-Distribution'

✓ D_{out} 분포에서 만들어진 example: 'Out-of-Distribution', OOD

✓ Training에 쓰이는 sample들의 분포 : D_{in}, D_{out}^{OE} / Test에 쓰이는 sample들의 분포 : D_{out}^{test}

✓ In : out = 5: 1

Experiments and Results

Image Classification & Text Classification Out-of-distribution Detection Results with OE(SOTA)

- ✓ Metric : FPRN, AUROC, AUPR
- ✓ +OE : Fine-tuning with OE
- ✓ +OECC : Fine-tuning with OECC
- ✓ Averaged over 10 runs and over 8 OOD datasets
- ✓ densenet100, resnet34

D_{in}	D_{out}^{test}	FPR95↓ +OE	FPR95↓ OECC	AUROC↑ +OE	AUROC↑ OECC	AUPR↑ +OE	AUPR↑ OECC
SVHN	Gaussian	0.0	0.0	100.	100.	100.	99.4
	Bernulli	0.0	0.0	100.	100.	100.	99.2
	Blobs	0.0	0.0	100.	100.	100.	99.6
	Icons-50	0.3	0.1	99.8	99.9	99.2	99.5
	Textures	0.2	0.1	100.	100.	99.7	99.6
	Places365	0.1	0.0	100.	100.	99.9	99.7
	LSUN	0.1	0.0	100.	100.	99.9	99.7
	CIFAR-10	0.1	0.0	100.	100.	99.9	99.7
	Mean	0.10	**0.03**	99.98	**99.99**	**99.83**	99.55
CIFAR-10	Gaussian	0.7	0.7	99.6	99.8	94.3	99.0
	Rademacher	0.5	1.1	99.8	99.6	97.4	97.6
	Blobs	0.6	1.5	99.8	99.1	98.9	91.7
	Textures	12.2	4.0	97.7	98.9	91.0	95.0
	SVHN	4.8	1.4	98.4	99.6	89.4	97.9
	Places365	17.3	13.3	96.2	96.9	87.3	89.5
	LSUN	12.1	6.7	97.6	98.4	89.4	91.9
	CIFAR-100	28.0	23.8	93.3	94.9	76.2	82.0
	Mean	9.50	**6.56**	97.81	**98.40**	90.48	**93.08**
CIFAR-100	Gaussian	12.1	0.7	95.7	99.7	71.1	97.2
	Rademacher	17.1	0.7	93.0	99.7	56.9	96.2
	Blobs	12.1	1.3	97.2	99.6	86.2	96.3
	Textures	54.4	50.1	84.8	87.8	56.3	61.5
	SVHN	42.9	16.7	86.9	94.9	52.9	74.1
	Places365	49.8	47.8	86.5	88.1	57.9	58.5
	LSUN	57.5	56.6	83.4	85.9	51.4	53.0
	CIFAR-10	62.1	57.2	75.7	78.7	32.6	35.2
	Mean	38.50	**28.89**	87.89	**91.80**	58.15	**71.50**

TEXT OOD DETECTION RESULTS

D_{in}	D_{out}^{test}	FPR90↓ +OE	FPR90↓ OECC	AUROC↑ +OE	AUROC↑ OECC	AUPR↑ +OE	AUPR↑ OECC
20 Newsgroups	SNLI	12.5	2.1	95.1	97.1	86.3	93.0
	IMDB	18.6	2.5	93.5	98.2	74.5	92.9
	Multi30K	3.2	0.1	97.3	99.4	93.7	98.6
	WMT16	2.0	0.2	98.8	99.8	96.1	99.4
	Yelp	3.9	0.4	97.8	99.6	87.9	97.9
	EWT-A	1.2	0.2	99.2	99.8	97.3	98.4
	EWT-E	1.4	0.1	99.2	99.9	97.2	98.9
	EWT-N	1.8	0.5	98.7	99.2	95.7	94.5
	EWT-R	1.7	0.1	98.9	99.4	96.6	98.3
	EWT-W	2.4	0.1	98.5	99.4	93.8	98.3
	Mean	4.86	**0.63**	97.71	**99.18**	91.91	**97.02**
TREC	SNLI	4.2	0.8	98.1	99.1	91.6	94.9
	IMDB	0.6	0.6	99.4	98.9	97.8	97.1
	Multi30K	0.3	0.2	99.7	99.9	99.0	99.6
	WMT16	0.2	0.2	99.8	99.9	99.4	99.6
	Yelp	0.4	0.8	99.7	99.1	96.1	92.9
	EWT-A	0.9	4.0	97.7	98.0	96.1	95.6
	EWT-E	0.4	0.3	99.5	99.2	99.1	98.1
	EWT-N	0.3	0.2	99.6	99.9	99.2	99.6
	EWT-R	0.4	0.2	99.5	99.6	98.8	98.9
	EWT-W	0.2	0.2	99.7	99.6	99.4	98.9
	Mean	0.78	**0.75**	99.28	**99.32**	**97.64**	97.52
SST	SNLI	33.4	7.4	86.8	95.8	52.0	76.4
	IMDB	32.6	10.8	85.9	95.8	51.5	77.6
	Multi30K	33.0	5.1	88.3	97.9	58.9	86.9
	WMT16	17.1	3.6	92.9	98.3	68.8	88.1
	Yelp	11.3	15.6	92.7	95.2	60.0	81.1
	EWT-A	33.6	21.4	87.2	92.7	53.8	70.8
	EWT-E	26.5	22.6	90.4	92.4	63.7	67.7
	EWT-N	27.2	19.2	90.1	93.6	62.0	67.4
	EWT-R	41.4	36.7	85.6	88.1	54.7	62.5
	EWT-W	17.2	36.7	92.8	88.1	66.9	62.5
	Mean	27.33	**17.91**	89.27	**93.79**	59.23	**74.10**

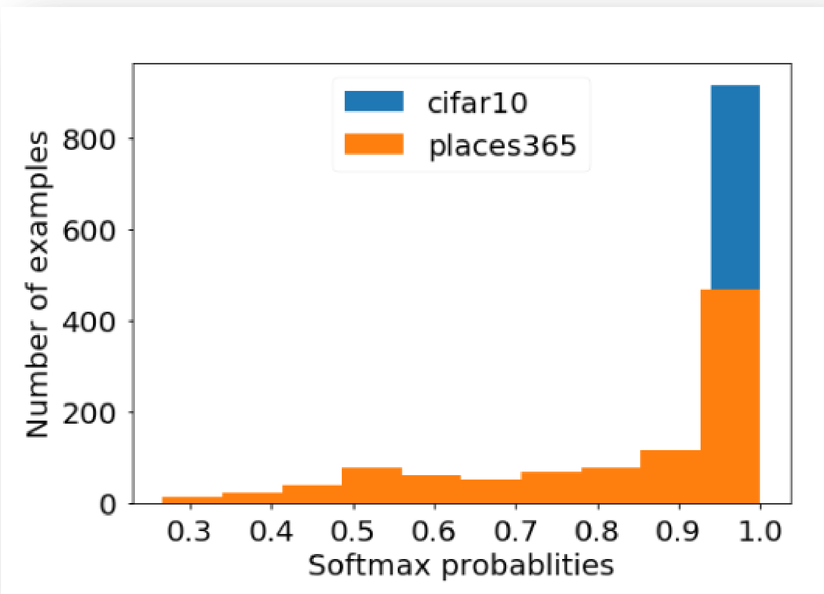
-> SOTA를 달성한 OE보다 거의 모든 데이터셋의 3가지 지표에 대해 월등한 높은 성능

Experiments and Results

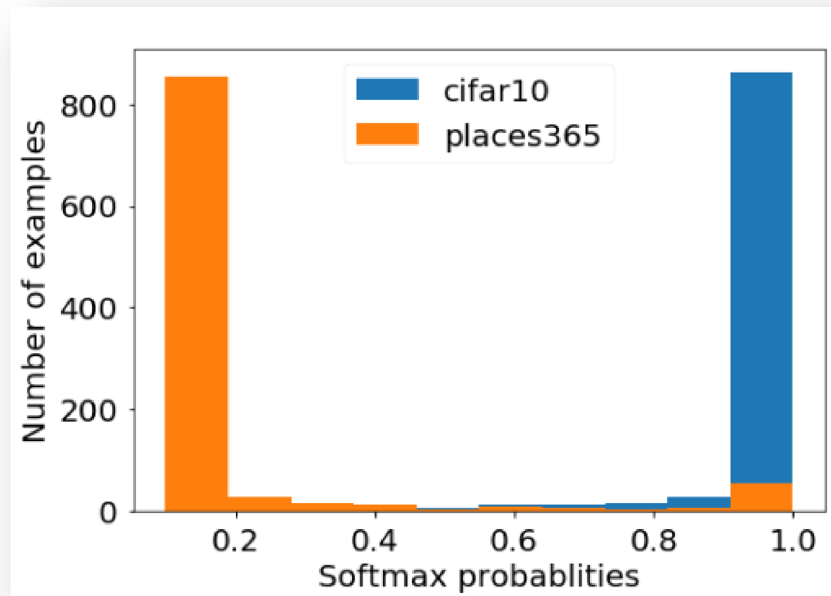
$$\underset{\theta}{\text{minimize}}\ \mathbb{E}_{(x,y)\sim D_{in}}[\mathcal{L}_{CE}(f_\theta(x),y)] + \lambda_1 \left(A_{tr} - \mathbb{E}_{x\sim D_{in}}\left[\max_{l=1,\dots,K}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right]\right)^2 + \lambda_2 \sum_{x^{(i)}\sim D_{out}^{OE}} \sum_{l=1}^{K} \left|\frac{1}{K} - \frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right| \tag{3}$$

Image Out-of-distribution Detection Results with OE(SOTA)

- D_{in} : CIFAR-10 데이터 셋, 비행기/자동차/새 등의 동물/배/트럭 총 10개의 레이블의 사진을 담고 있음.
- D_{out}^{test} : Places365 데이터 셋, 풍경 사진을 담고 있음.
- 결과 : OECC로 fine-tuning후, 성격이 다른 두 데이터를 잘 구분해내고 있음



[MSP Baseline Detector]



[MSP Detector Fine-tuned with OECC]

Figure 1: Histogram of soft-max probabilities with CIFAR-10 asDinand Places365 asDtestout(1,000 samples fromeach dataset).Top: MSP base-line detector.Bottom: MSP de-tector fine-tuned with (3).

Experiments and Results

Regularization Term

D_{in}	λ_1	λ_2	FPR95↓	AUROC↑	AUPR↑	Test Accuracy(D_{in})
CIFAR-10	-	-	34.94	89.27	59.16	94.65
	-	√	8.87	96.72	77.65	92.72
	√	√	6.56	98.40	93.08	93.86
CIFAR-100	-	-	62.66	73.11	30.05	75.73
	-	√	26.75	91.59	68.27	71.29
	√	√	28.89	91.80	71.50	73.14

- λ_1과 λ_2를 순차적으로 추가하며 성능을 측정
- 추가로 D_{in}의 sample들을 test sample들로 실험 수행
- 결과 : Test Accuracy(D_{in})의 성능이 조금 줄어든 것에 비해 OECC(λ_1과 λ_2 모두 존재)가 다른 모델보다 성능이 월등하게 증가한 것을 확인할 수 있음.

Experiments and Results

OECC + post-training method – Mahalanobis Detection

✓ OECC와 post-training method를 결합:
Standard cross-entropy loss function으로 DNN을 학습 시키고 OECC 로 fine-tuning한 후, post-training method 적용

✓ **Post-training method : Mahalanobis Detection Method**
: Confidence Score를 Mahalanobis 거리로 측정

: $M(x) = \max_c -(f(x) - \widehat{\mu_c})^T \widehat{\Sigma}^{-1}(f(x) - \widehat{\mu_c})$

x : test sample, f(x) : softmax neural classifier

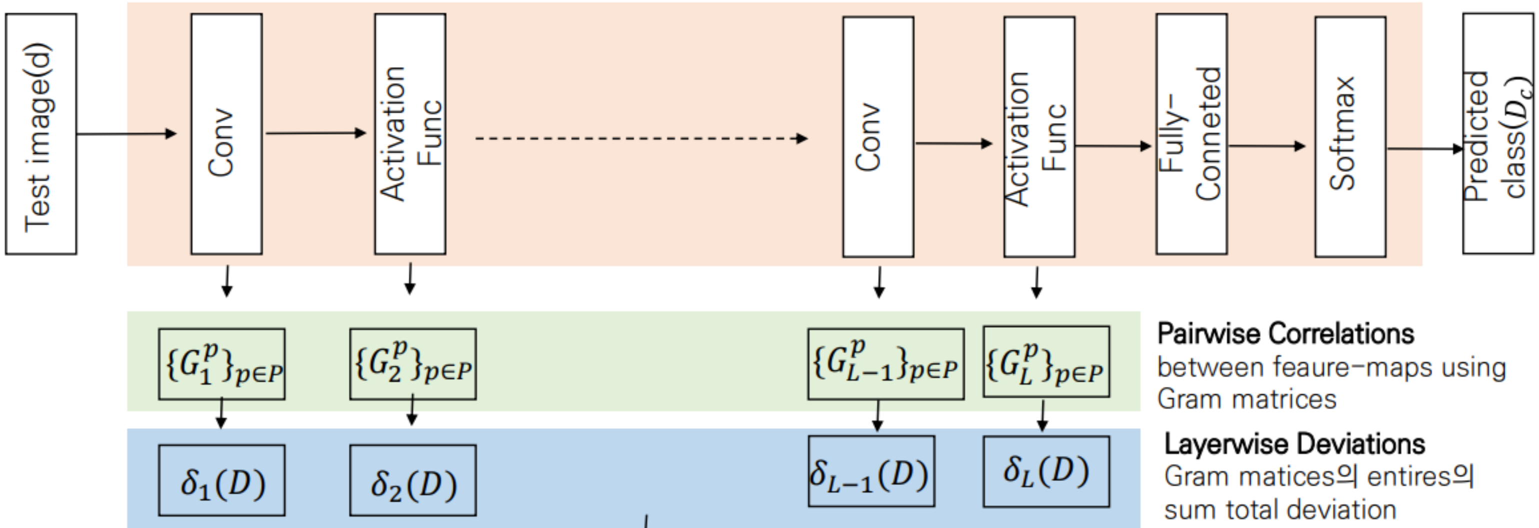
$\widehat{\mu_c}$: mean of multivariate Gaussian distribution of class c∈{1,…,C}

D_{in}	D_{out}^{test}	TNR95↑ GM	TNR95↑ OECC+GM	AUROC↑ GM	AUROC↑ OECC+GM	DAcc↑ GM	DAcc↑ OECC+GM
CIFAR-10	SVHN	96.0	**98.5**	99.1	**99.6**	95.8	**97.4**
	TinyImageNet	98.8	**99.3**	99.7	**99.8**	97.9	**98.3**
	LSUN	99.5	**99.8**	99.9	99.9	97.9	**99.0**
CIFAR-100	SVHN	**89.4**	88.9	**97.4**	97.0	**92.4**	92.1
	TinyImageNet	95.8	**96.2**	99.0	99.0	95.6	**95.7**
	LSUN	97.3	**98.1**	**99.4**	99.3	96.4	**97.0**
SVHN	CIFAR-10	80.2	**98.5**	95.5	**99.6**	89.0	**97.5**
	TinyImageNet	99.1	**99.9**	99.7	**100.0**	97.9	**99.7**
	LSUN	99.5	**100.0**	99.8	**100.0**	98.5	**99.9**

-> Mahalanobis-distance based Detector 보다 모든 결과에 대해 우수한 성능
-> 모든 pre-trained softmax classifier에 적용 가능

Experiments and Results

OECC + post-training method – Gram Detection Method



Pairwise Correlations
between feaure-maps using Gram matrices

Layerwise Deviations
Gram matices의 entires의 sum total deviation

Total Deviation : $$\sum_{l=1}^{L} \frac{\delta_1(D)}{E(\delta_1(D))}$$

D_{in}	D_{out}^{test}	TNR95↑		AUROC↑		DAcc↑	
		GM	OECC+GM	GM	OECC+GM	GM	OECC+GM
CIFAR-10	SVHN	96.0	**98.5**	99.1	**99.6**	95.8	**97.4**
	TinyImageNet	98.8	**99.3**	99.7	**99.8**	97.9	**98.3**
	LSUN	99.5	**99.8**	99.9	99.9	97.9	**99.0**
CIFAR-100	SVHN	**89.4**	88.9	**97.4**	97.0	**92.4**	92.1
	TinyImageNet	95.8	**96.2**	99.0	99.0	95.6	**95.7**
	LSUN	97.3	**98.1**	**99.4**	99.3	96.4	**97.0**
SVHN	CIFAR-10	80.2	**98.5**	95.5	**99.6**	89.0	**97.5**
	TinyImageNet	99.1	**99.9**	99.7	**100.0**	97.9	**99.7**
	LSUN	99.5	**100.0**	99.8	**100.0**	98.5	**99.9**

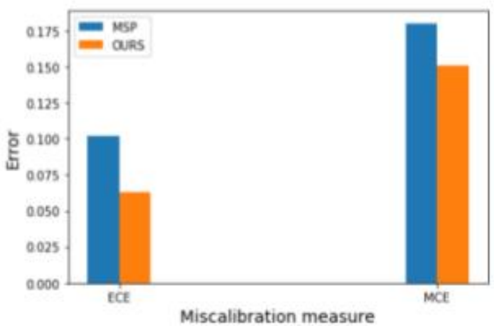
-> Gram-Detector 보다 모든 결과에 대해 우수한 성능

Experiments and Results

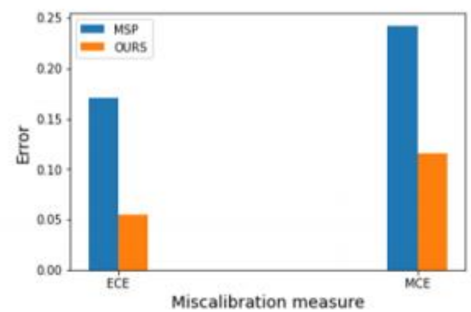
Calibration Experiment

- Calibration : 모델의 출력 값이 실제 confidence(=calibrated confidence)를 반영하도록 만들어 confidence와 accuracy가 일치하도록 하는 것.

  예시) X의 Y1에 대한 모델의 출력이 0.8 -> 80% 확률로 Y1일 거라는 의미를 갖도록 만드는 것



[CIFAR-100으로 측정한 Image data의 ECE와 MCE]



[SST으로 측정한 Text data의 ECE와 MCE]

- ECE(Expected Calibration Error) : confidence의 평균값과 accuracy의 평균값의 차이
- MCE(Maximum Calibrated Error) : worst-case일 때의 confidence와 accuracy의 차의 분산

- 결과 : OURS(OECC로 fine-tuning한 모델)이 ECE, MCE 면에서 월등하게 성능이 좋아짐을 알 수 있음

- 결론 : $\left(A_{tr} - E_{x\sim D_{in}}\left[\max_{l=1,\dots,k}\left(\frac{e^{z_l}}{\sum_{j=1}^{K} e^{z_j}}\right)\right]\right)^2$ 값을 최소화하면 OOD의 탐지 성능 뿐만 아니라 더 'Calibrated'한 모델을 얻을 수 있음

Conculsion

Experiments and Result

- 기존 OE에 단순한 2개의 가정을 추가한 OECC기법을 적용하였고 OOD에 우수한 성능
- OECC + SOTA Post-training 조합에 대한 baseline을 제시
- calibration 문제 또한 OE SOTA보다 좋아짐을 실험으로 증명

감사합니다

Conclusion and Follow-ups

- Demonstrated a softmax prediction probability baseline for error, out-of-distribution detect
- Presented the abnormality module (+ gain)
- Presented Evaluation Metric in OOD task(property)

Deep Anomaly Detection with Outlier Exposure, 2019 ICLR

$\mathcal{D}_{\text{in}}$	FPR95 ↓ MSP	FPR95 ↓ +OE	AUROC ↑ MSP	AUROC ↑ +OE	AUPR ↑ MSP	AUPR ↑ +OE
SVHN	6.3	0.1	98.0	100.0	91.1	99.9
CIFAR-10	34.9	9.5	89.3	97.8	59.2	90.5
CIFAR-100	62.7	38.5	73.1	87.9	30.1	58.2
Tiny ImageNet	66.3	14.0	64.9	92.2	27.2	79.3
Places365	63.5	28.2	66.5	90.6	33.1	71.0

Table 1: Out-of-distribution image detection for the maximum softmax probability (MSP) baseline detector and the MSP detector after fine-tuning with Outlier Exposure (OE). Results are percentages and also an average of 10 runs. Expanded results are in Appendix A.

$\mathcal{D}_{\text{in}}$	FPR95 ↓ MSP	FPR95 ↓ +GAN	FPR95 ↓ +OE	AUROC ↑ MSP	AUROC ↑ +GAN	AUROC ↑ +OE	AUPR ↑ MSP	AUPR ↑ +GAN	AUPR ↑ +OE
CIFAR-10	32.3	37.3	11.8	88.1	89.6	97.2	51.1	59.0	88.5
CIFAR-100	66.6	66.2	49.0	67.2	69.3	77.9	27.4	33.0	44.7

Table 4: Comparison among the maximum softmax probability (MSP), MSP + GAN, and MSP + GAN + OE OOD detectors. The same network architecture is used for all three detectors. All results are percentages and averaged across all $\mathcal{D}_{\text{out}}^{\text{test}}$ datasets.

- Outlier Exposure는 기존 방법들에 독립적으로 추가가 가능한 아이디어
- 기존 detector들에 Outlier Exposure를 추가하였을 때 얼마나 성능이 향상되는지를 논문에서 결과로 제시
- 다만 **Outlier Exposure**로 **어떤 데이터 셋**을 사용하는지에 따라 성능이 크게 달라질 수 있다는 점이 풀어야 할 문제(Future work)
- Gaussian noise나 GAN으로 생성한 sample 등을 활용하는 것은 크게 효과적이지 않음
- 반면, Outlier Exposure로 사용하는 데이터 셋을 최대한 realistic 하면서 size도 크고, 다양하게 구축하는 것이 좋은 성능을 달성하는 데 도움을 준다고 가이드를 제시해주고 있음
- 기존에 존재하던 Out-of-distribution Detection 알고리즘들에 **추가로 적용**이 가능하면서도 **손쉽게 구현**이 가능한 방법론을 제안하였고, 실제로 효과적인 성능 향상