Recent Advances in Anomaly Detection and Adversarial Robustness

Mohammad Hossein Rohban, Ph.D.

Computer Engineering Department

Sharif University of Technology

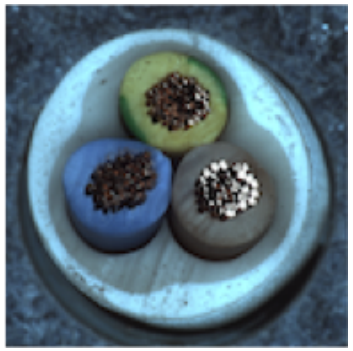
rohban@sharif.edu

Joint work with: Mohammadreza Salehi Dehnavi, Atrin Arya, Barbod Pajoum, Mohammad Otoofi, Amirreza Shaeiri, Niousha Sadjadi, Ainaz Eftekhar, and Hamid R. Rabiee

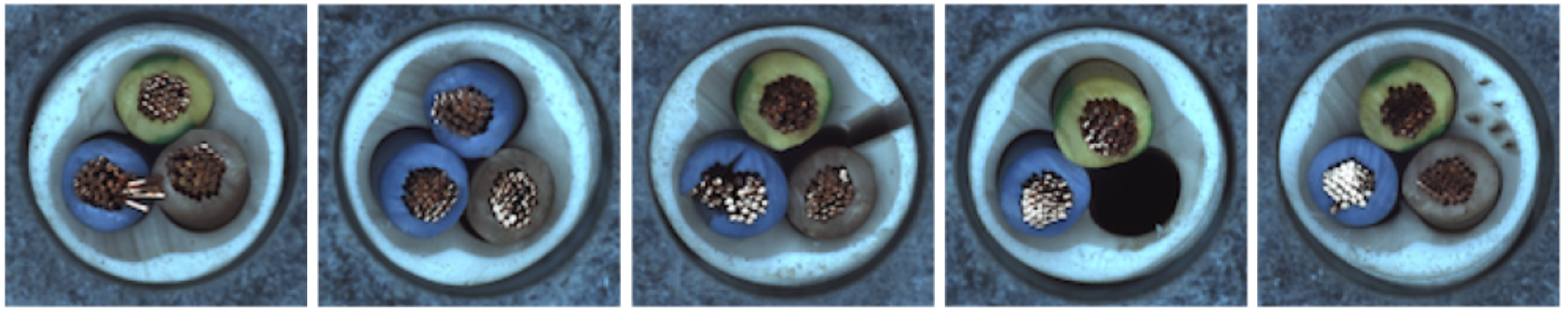
Motivation

- Collected data about the normal behavior.
- Infer at the test time about being normal vs. anomalous.
- Why?
  - could be hard to sample from the anomalous class (e.g. healthy subjects imaged in a clinical setup in CMR images)
  - Anomaly could span a wide range of possibilities.

normal



anomalous

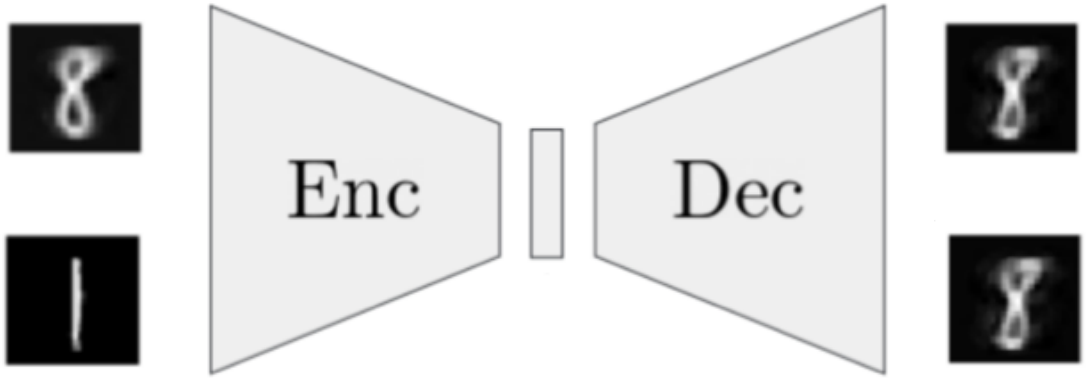


Desirable properties of a solution

- High precision/recall, or sometimes AUC.
- Low False Positive rate when True Positive Rate = 99%.
- Data efficiency (under small sample size)
- Test time efficiency (e.g. real-time detection)
- General and adaptable to various datasets.
- Agnostic and insensitive to choices of hyperparameters, early stopping criteria.

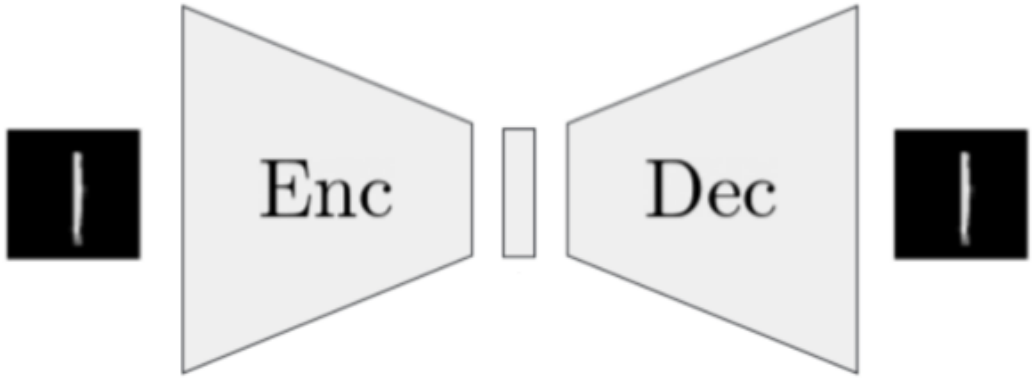
A common good practice

- Train an autoencoder on the normal samples.
- Threshold the reconstruction error.
- E.g. let the digit 8 be the normal class in MNIST. We expect:



Criticisms

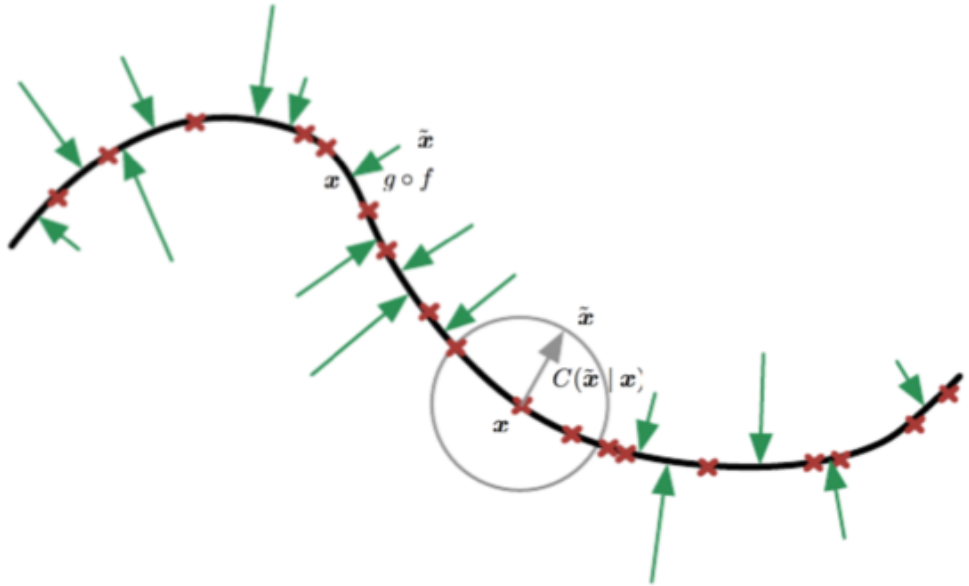
- AE generalization : it may reconstruct anomalies as well!



- Reconstruction is poor for the the more complex datasets, e.g. CIFAR-10.

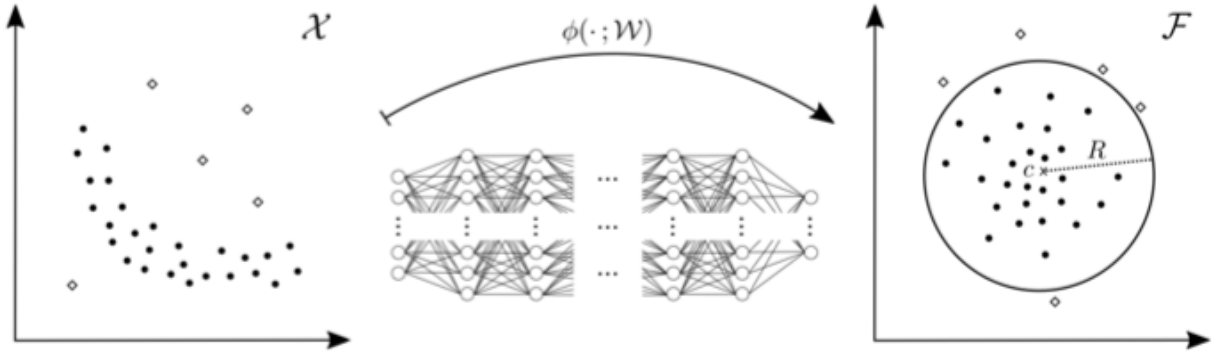
How to address the generalization issue?

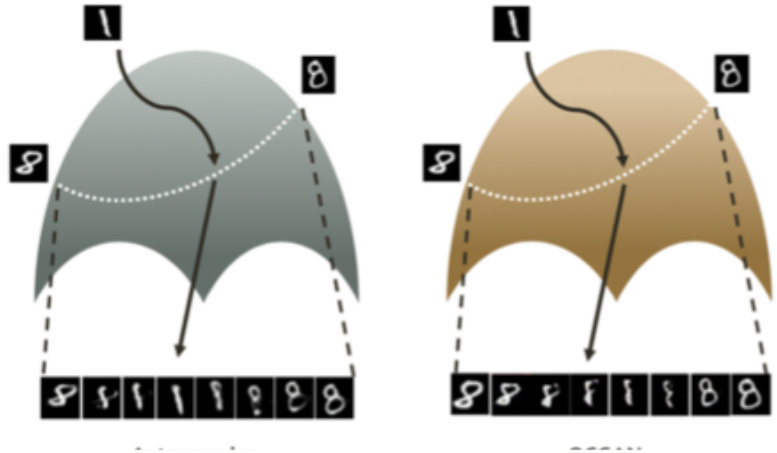
- The latent space is not **compact** for the normal samples.
- It encodes many low-level features in the image that can be used to reconstruct *any arbitrary* image.
- Simplest possible solution: Denoising Autoencoder

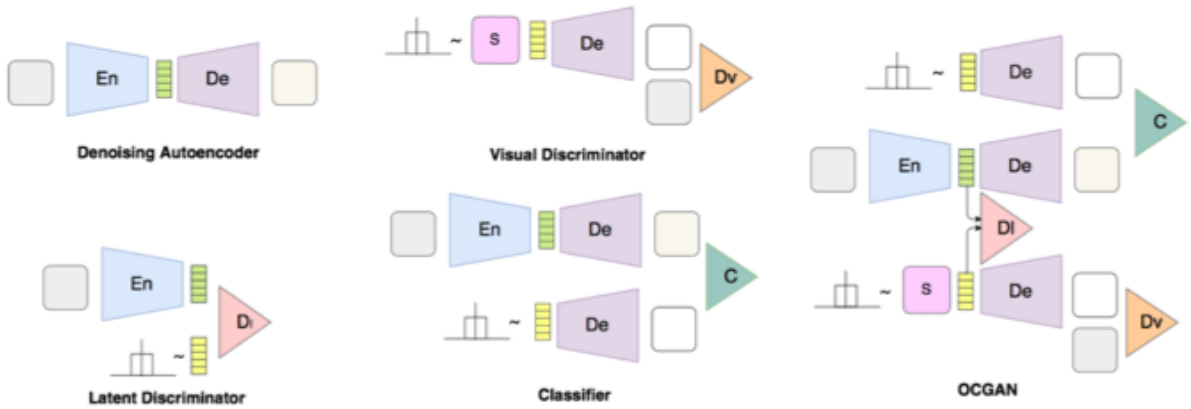


Other options

- Deep SVDD: Force the latent layer be bounded in a hypersphere.
- OCGAN: Force the latent layer to follow a uniform distribution through GANs.

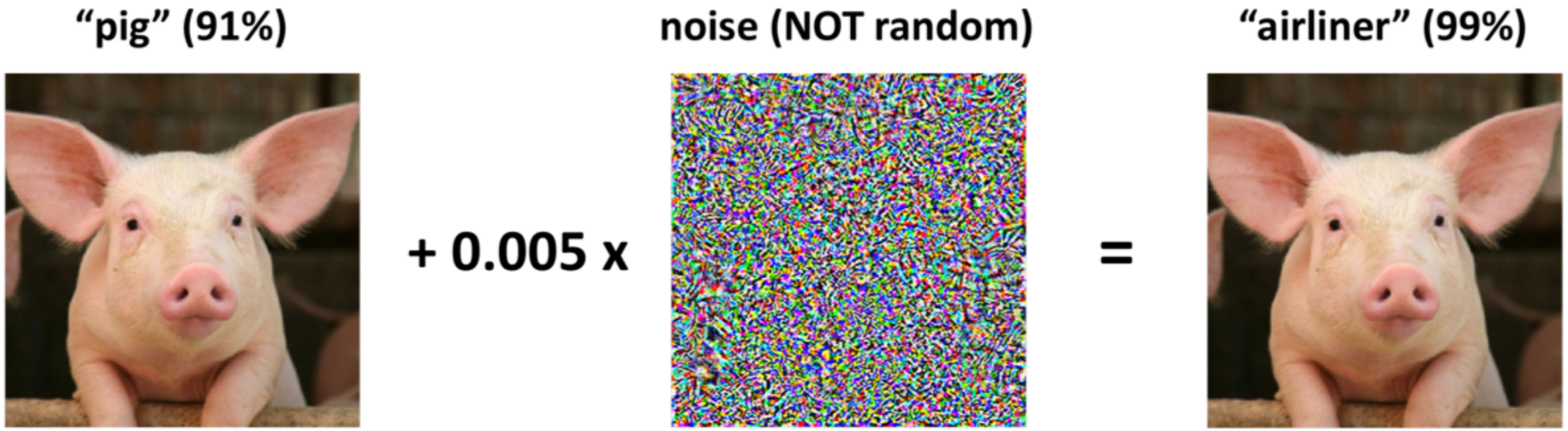






Our solution: Adversarial Robustness

- One could find small noises that change the model output drastically:



How to find such noises?

- Solve the optimization:

$$\max_{\delta \in \Delta} l(f(x+\delta), y)$$

which can be achieved by the gradient ascent w.r.t. x.

What choices of Δ do we have?

- Typical choices are when some norm of δ is restricted to be a small value:

$$\Delta = \{\delta: \|\delta\|_p \leq \epsilon\}$$

- Typically $p = \infty$ is used, which allows all pixels of the image be perturbed by an ϵ amount.
- For $p = 0$, only few pixels are perturbed by arbitrary amounts.

How to make the model robust?

- We ought to change the training criterion.
- Standard training:

$$\min_f \mathbb{E}_{(x,y)\sim D}\, l(f(x), y)$$

- Adversarially robust training:

$$\min_f \mathbb{E}_{(x,y)\sim D} \max_{\delta\in\Delta} l(f(x+\delta), y)$$

How to do it in practice? [Alex Madry 2017]

- Estimate δ_i for each data point x_i in the training dataset.
  - Through projected gradient ascent.
- Update weights of the network by a single step of gradient descent on the new dataset $\{(x_1 + \delta_1, y_1), \ldots, (x_n + \delta_n, y_n)\}$.
- Go to step 1 and iterate.

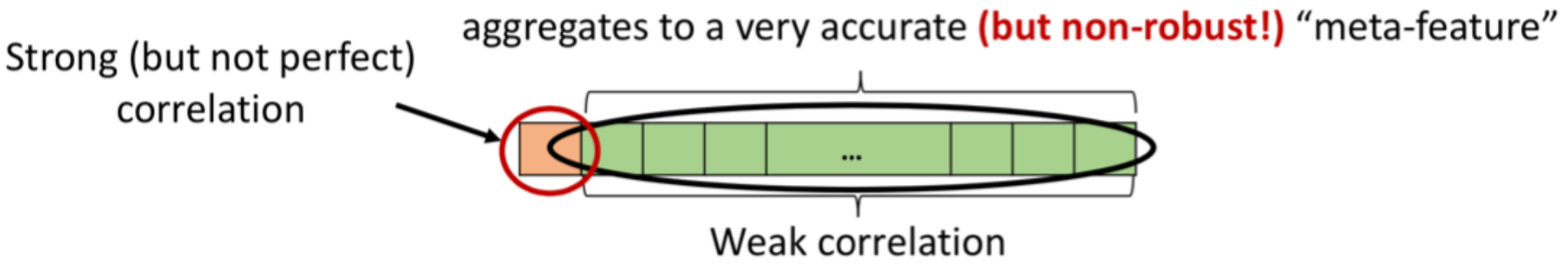
Does Being Robust Help "Standard" Generalization? (cont.)

Theorem [Tsipras Santurkar Engstrom Turner M 2018]:
No "free lunch": can exist a trade-off between accuracy and robustness

Basic intuition:

→ In standard training, **all correlation is good correlation**

→ If we want robustness, **must avoid** weakly correlated features



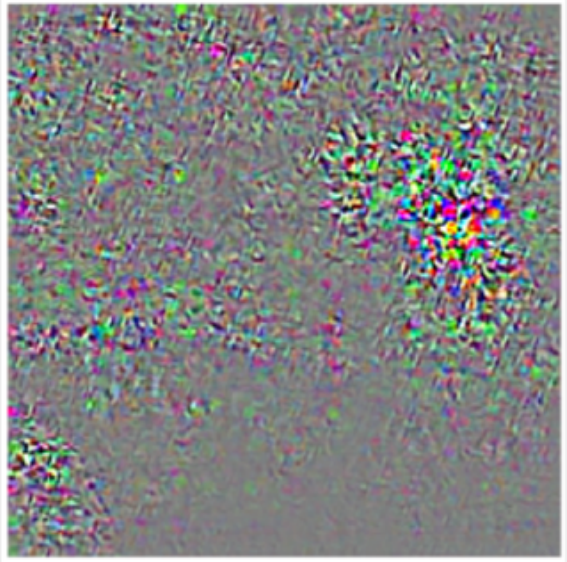
Standard training: use all of features, maximize accuracy

Adversarial training: use only single robust feature (**at the expense of accuracy**)

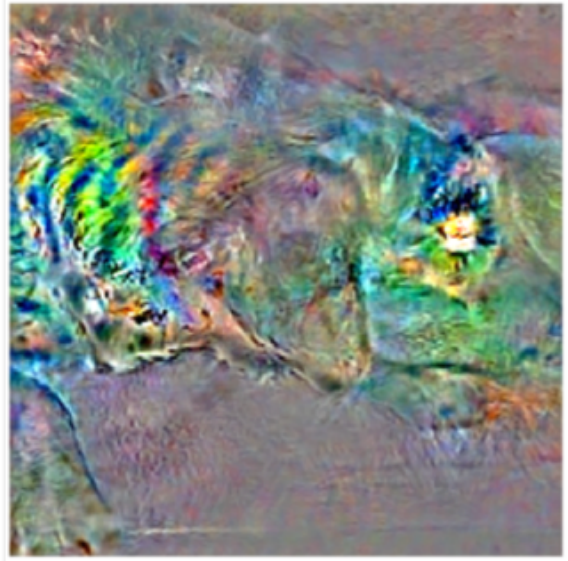
But There Are (Unexpected?) Benefits Too!

Models become more **semantically meaningful**

Input



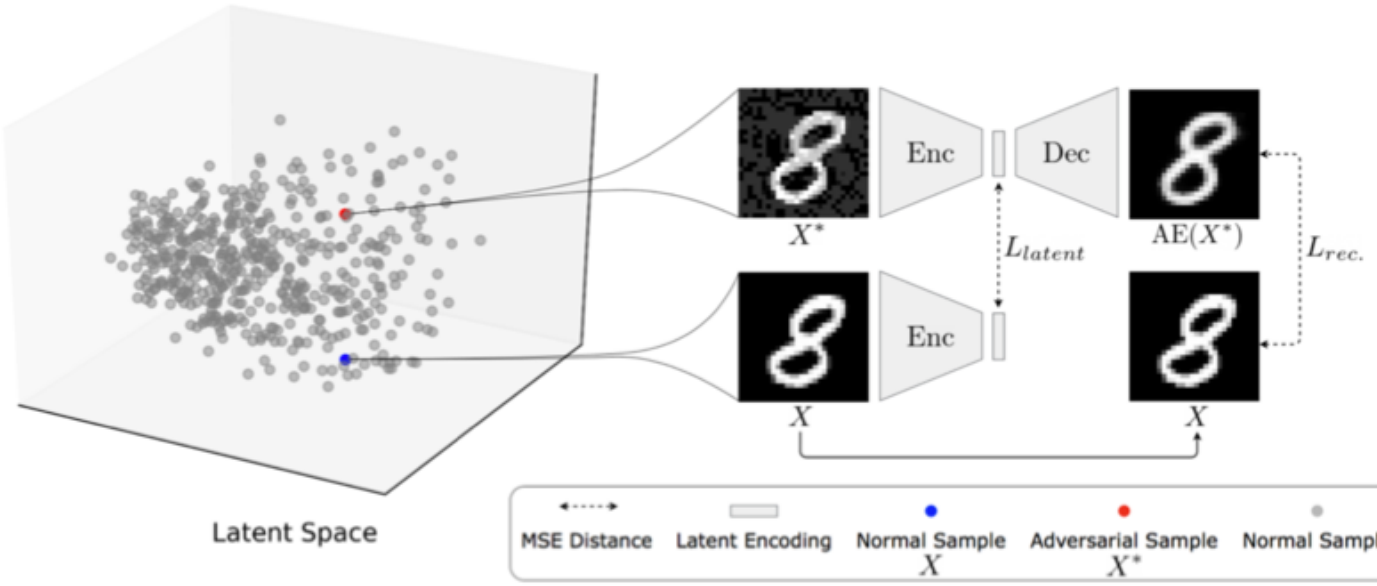
Gradient of
standard model



Gradient of
adv. robust model

Basic idea (Adversarially Robust AE) or ARAE

- Use adversarial training in the autoencoder to get rid of the meaningless non-robust features.



$$L_{\text{AE}} = L_{\text{rec.}} + \gamma L_{\text{latent}}$$

$$L_{\text{rec.}} = \|X - \text{Dec}(\text{Enc}(X^*))\|_2^2$$

$$L_{\text{latent}} = \|\text{Enc}(X + \delta_X) - \text{Enc}(X)\|_2^2$$

$$\max_{\delta_X} L_{\text{latent}} \text{ s.t. } \|\delta_X\|_\infty \leq \epsilon, \text{ where } L_{\text{latent}} = \|\text{Enc}(X + \delta_X) - \text{Enc}(X)\|_2^2$$

ARAE on MNIST

Input (Anomalous)

DAE reconstruction

ARAE reconstruction

Fig. 1: Unlike DAE, ARAE that is trained on the normal class, which is the digit 8, reconstructs a normal instance when it is given an anomalous digit, from the class 1. The first row shows the input images. The second and third rows show DAE and ARAE reconstructions of the corresponding inputs, respectively. ARAE is trained based on bounded ℓ_∞, ℓ_2, rotation, and translation perturbations.

Experimental Setup

- Encoder = 3 dense layers (512, 256, 128), with sigmoid activation.
- epsilon = 0.2 for MNIST and 0.05 for other datasets.
- gamma = 0.1 for all datasets.

$$\max_{\delta_X} L_{\text{latent}} \text{ s.t. } \|\delta_X\|_\infty \leq \epsilon, \text{ where } L_{\text{latent}} = \|\text{Enc}(X + \delta_X) - \text{Enc}(X)\|_2^2$$

- One class is normal, the others are anomalous
- Testing is done on the originally specified test set of each dataset.

Results on MNIST

Table 1: AUC values for the MNIST dataset. The values were obtained for each class using protocol 2.

Method	0	1	2	3	4	5	6	7	8	9	Mean
VAE [16]	98.5	99.7	94.3	91.6	94.5	92.9	97.7	97.5	86.4	96.7	95.0
OCSVM [7]	99.5	99.9	92.6	93.6	96.7	95.5	98.7	96.6	90.3	96.2	96.0
AnoGAN [29]	96.6	99.2	85.0	88.7	89.4	88.3	94.7	93.5	84.9	92.4	91.3
DSVDD [26]	98.0	99.7	91.7	91.9	94.9	88.5	98.3	94.6	93.9	96.5	94.8
MTQM [36]	99.5	99.8	95.3	96.3	96.6	96.2	99.2	96.9	95.5	97.7	97.3
OCGAN [24]	99.8	99.9	94.2	96.3	97.5	98.0	99.1	98.1	93.9	98.1	**97.5**
LSA [1]	99.3	99.9	95.9	96.6	95.6	96.4	99.4	98.0	95.3	98.1	**97.5**
DAE	99.6	99.9	93.9	93.5	96.4	94.3	99.0	95.8	89.1	97.5	95.9
ARAE	99.8	99.9	96.0	97.2	97.0	97.4	99.5	96.9	92.4	98.5	**97.5**

FPR at TPR = 99.5%

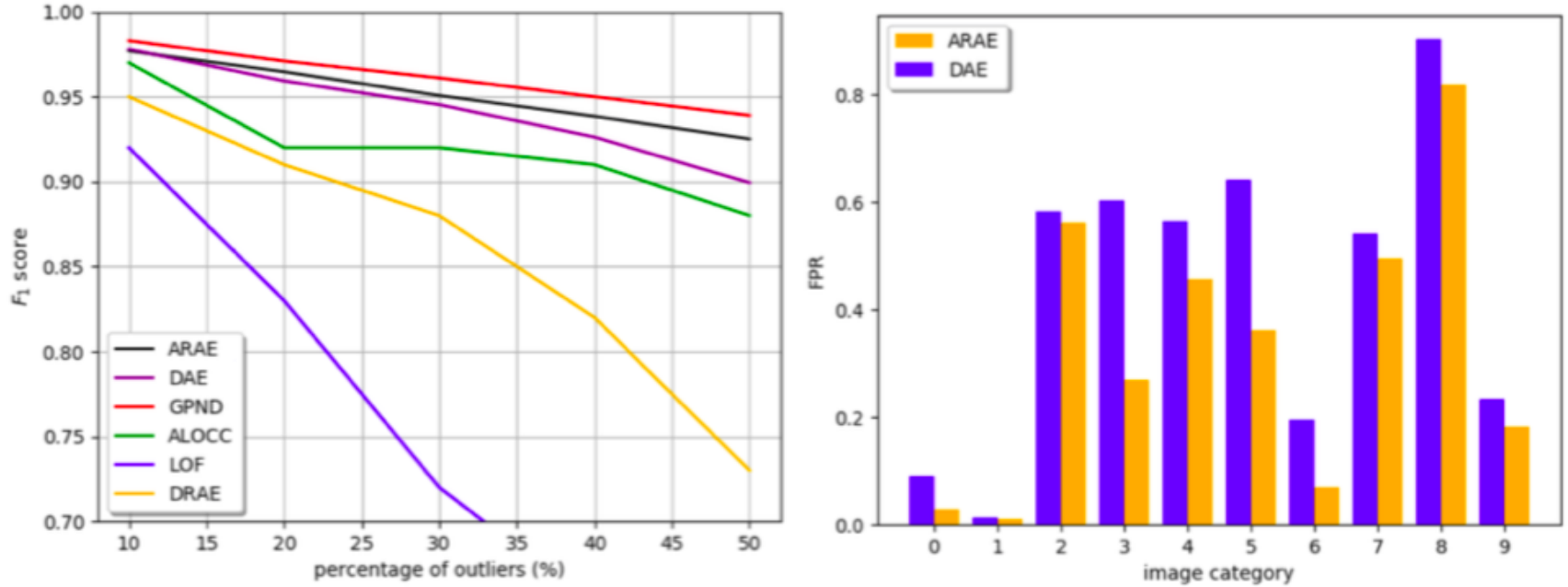


Fig. 3: Left: F_1 scores for the MNIST dataset. Right: FPR at 99.5% TPR of the MNIST dataset.

Results on Fashion-MNIST

Table 2: AUC values for the Fashion-MNIST dataset. The values were obtained for each class using protocol 2.

Method	0	1	2	3	4	5	6	7	8	9	Mean
VAE [16]	87.4	97.7	81.6	91.2	87.2	91.6	73.8	97.6	79.5	96.5	88.4
OCSVM [7]	91.9	99.0	89.4	94.2	90.7	91.8	83.4	98.8	90.3	98.2	92.8
DAGMM [45]	30.3	31.1	47.5	48.1	49.9	41.3	42.0	37.4	51.8	37.8	41.7
DSEBM [42]	89.1	56.0	86.1	90.3	88.4	85.9	78.2	98.1	86.5	96.7	85.5
MTQM [36]	92.2	95.8	89.9	93.0	92.2	89.4	84.4	98.0	94.5	98.3	92.8
LSA [1]	91.6	98.3	87.8	92.3	89.7	90.7	84.1	97.7	91.0	98.4	92.2
DAE	92.6	99.2	90.3	93.8	91.8	91.6	83.4	98.7	90.7	97.6	92.9
ARAE	93.7	99.1	91.1	94.4	92.3	91.4	83.6	98.9	93.9	97.9	**93.6**

Results on COIL-100

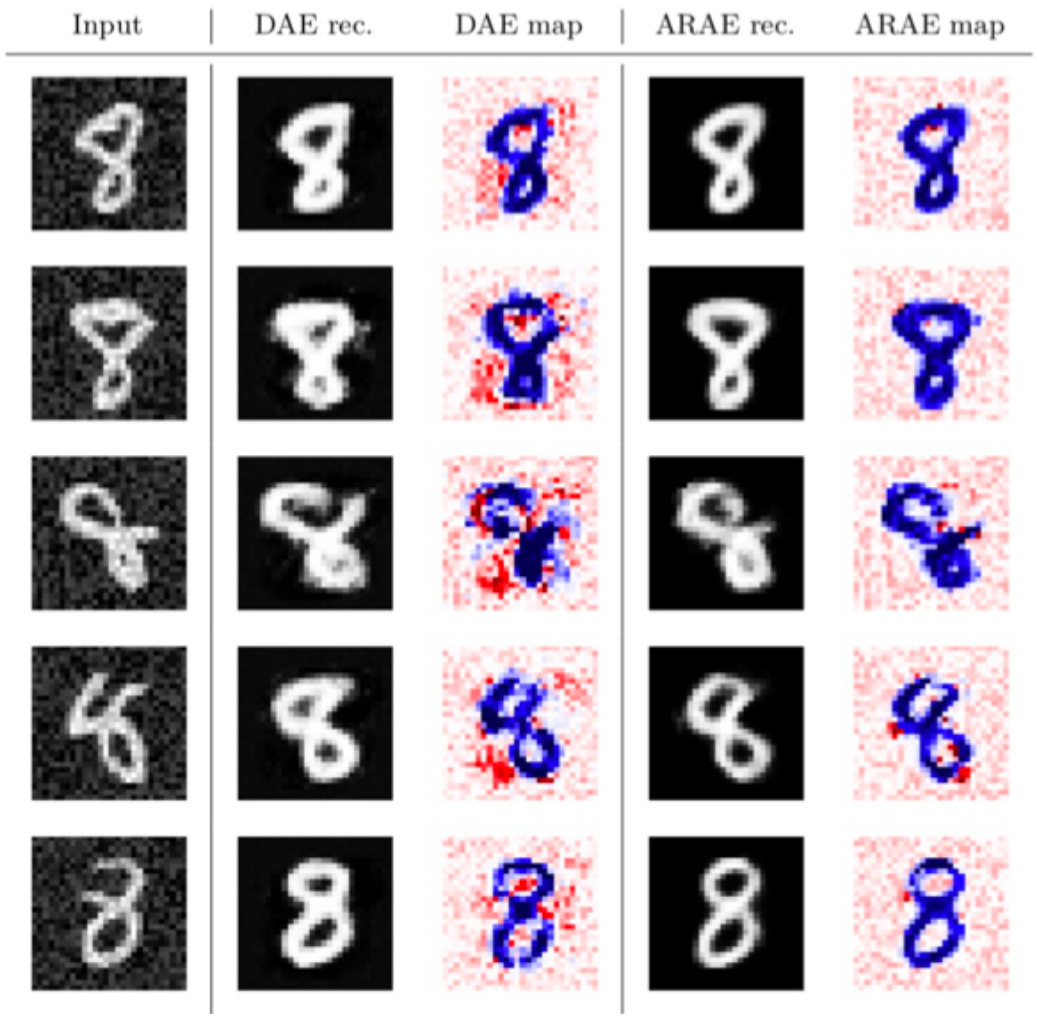
- 80% of normal is used for training
- Test set is sampled from the remaining 20%.
- n = number of normal classes

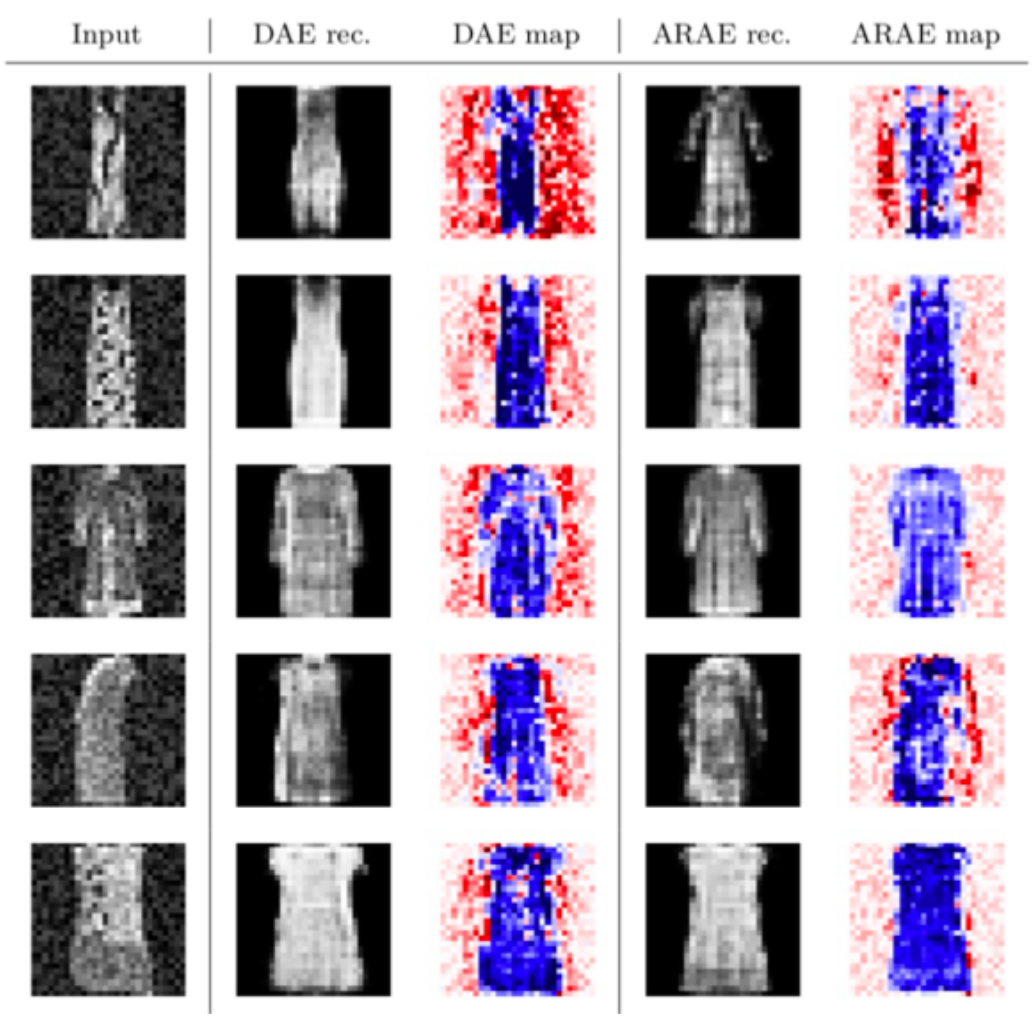
Table 3: AUC and F_1 values for the COIL-100 dataset. The values were obtained using protocol 1 for $n \in \{1, 4, 7\}$ and different anomaly percentages.

	OutlierPursuit [39]	DPCP [33]	l_1 thresholding [30]	R-graph [40]	GPND [25]	DAE	ARAE
Normal samples: **one** category of images, Anomalous samples: 50%							
AUC	0.908	0.900	0.991	0.997	0.968	0.997	**0.998**
F_1	0.902	0.882	0.978	0.990	0.979	**0.994**	0.993
Normal samples: **four** category of images, Anomalous samples: 25%							
AUC	0.837	0.859	0.992	0.996	0.945	0.990	**0.997**
F_1	0.686	0.684	0.941	0.970	0.960	0.950	**0.973**
Normal samples: **seven** category of images, Anomalous samples: 15%							
AUC	0.822	0.804	0.991	**0.996**	0.919	0.986	0.993
F_1	0.528	0.511	0.897	**0.955**	0.941	0.901	0.941

Where do ARAE and DAE attend?

- Used occlusion-1 to determine saliency maps:





Disadvantages

- AE-based approaches often can not model complex datasets (e.g. CIFAR-10).
- Long training time for large networks to due the PGD attack iterative nature.

Self-Supervised Learning comes to rescue!

- AE works at the pixel level abstraction.
- Use SSL to learn better representations in the unsupervised settings.
- Solving a **puzzle** or **colorization** on the input to learn the *input structure* rather than the pixel level relationships.
- Shortcuts is a big issue in SSL methods
  - The network could cheat in solving the puzzle, but looking at the input artifacts.
  - E.g. horizontal vs. vertical edges

Shortcuts

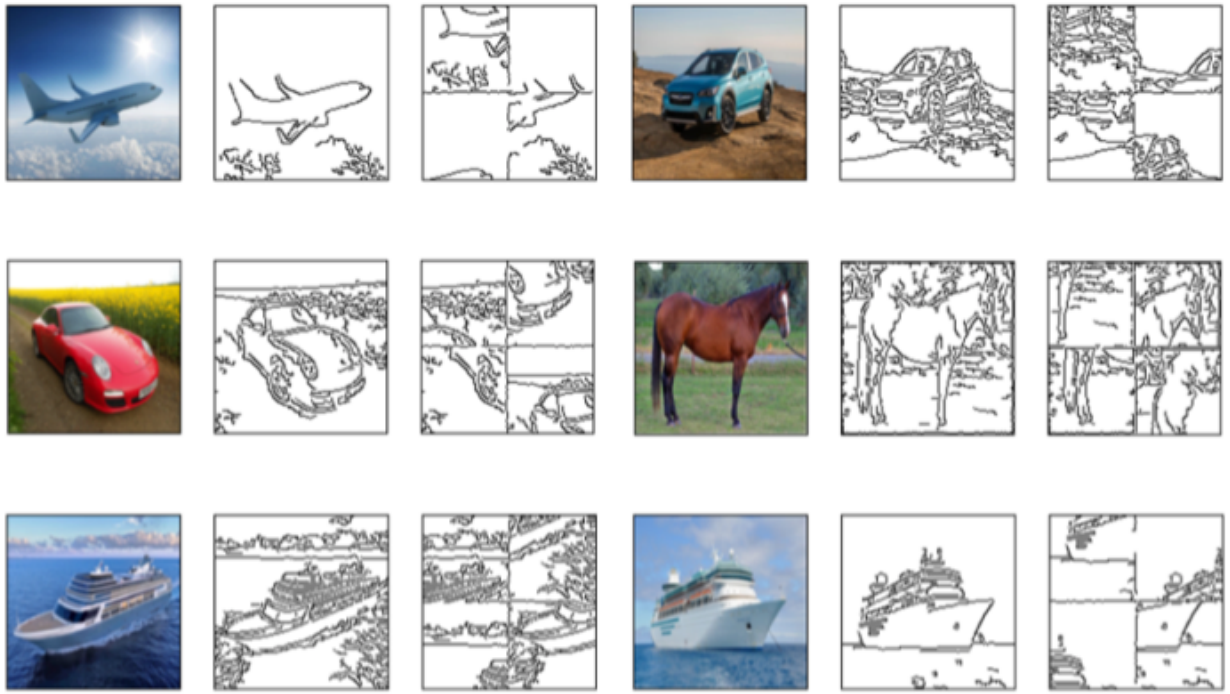
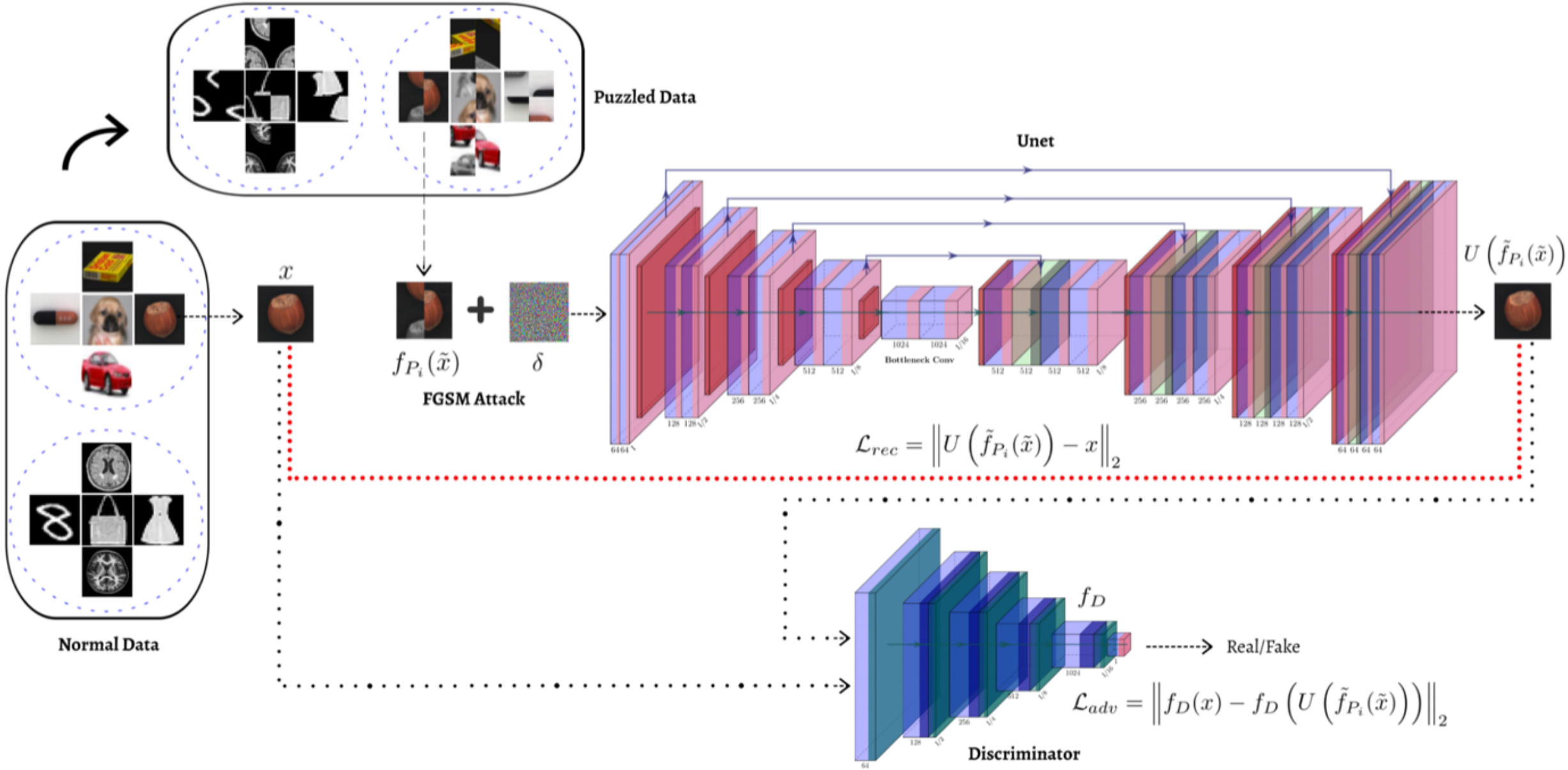


Figure 6: As it is shown, by puzzling an image, some trivial features are produced that conduct the model to learn trivial solutions of doing the puzzle. For example, model could understand the number of displacements just by noticing the vertical and horizontal lines.


Puzzled Data
Unet
Normal Data
x
$f_{P_i}(\tilde{x})$
δ
FGSM Attack
Bottleneck Conv
$U\left(\tilde{f}_{P_i}(\tilde{x})\right)$
$\mathcal{L}_{rec} = \left\| U\left(\tilde{f}_{P_i}(\tilde{x})\right) - x \right\|_2$
f_D
Real/Fake
Discriminator
$\mathcal{L}_{adv} = \left\| f_D(x) - f_D\left(U\left(\tilde{f}_{P_i}(\tilde{x})\right)\right) \right\|_2$

Test time

- Apply *various random permutations* on input and aggregate the reconstruction error.
- E.g. of aggregation functions: min, max, avg.
- We adopted avg.

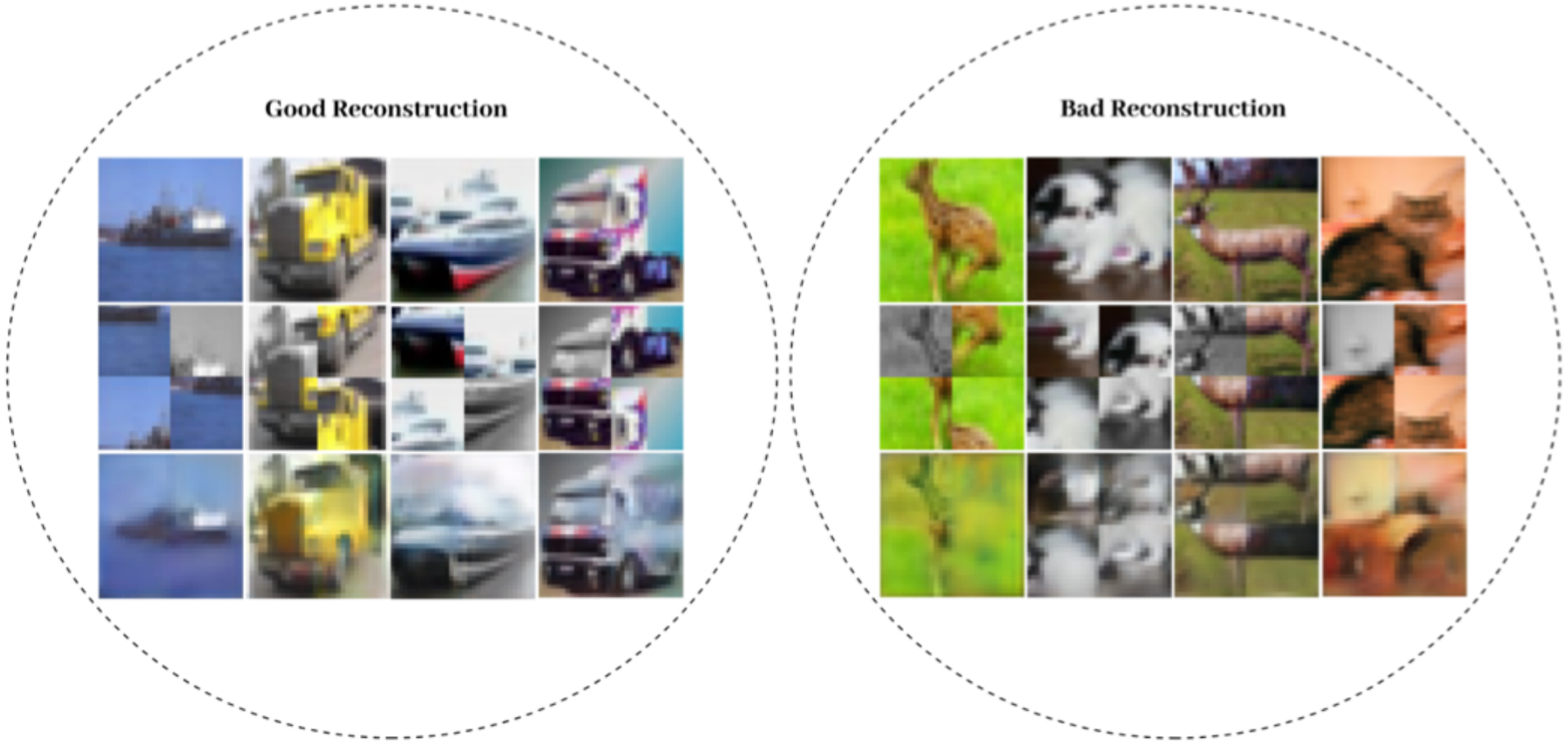


Figure 8: Outputs of the model trained on the class car of the CIFAR-10 [16] dataset for anomalous inputs. As it shows, some samples of the truck class are also unpuzzled well, which could be for high similarity between main features of car and truck. The same implications exist for some of the ship class samples.

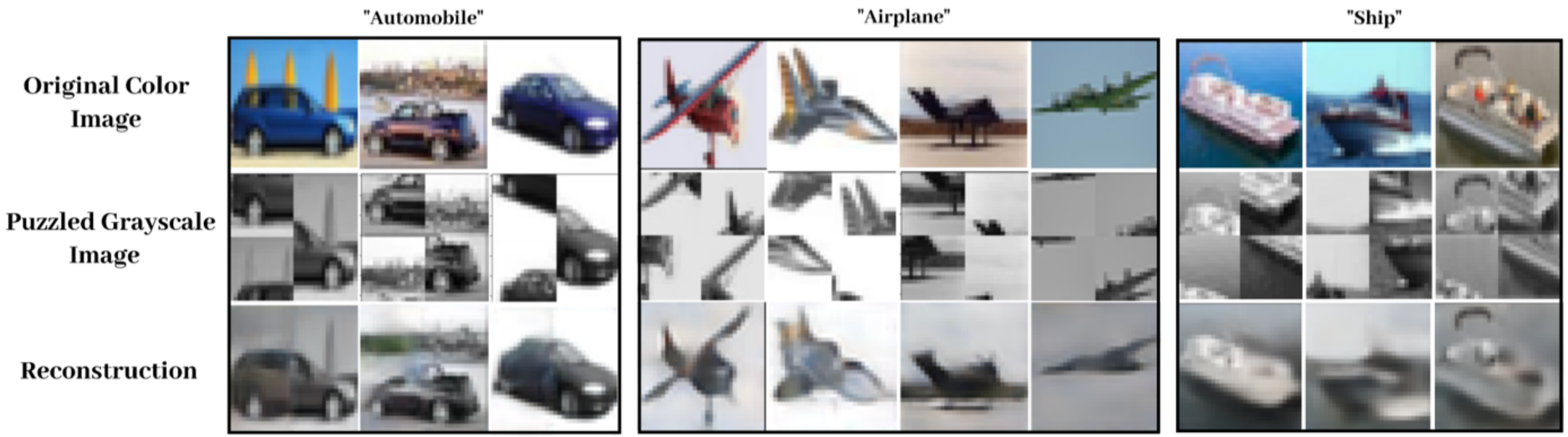


Figure 7: Effect of converting images to gray-scale on the model learned features for some classes like car, plane and ship of the CIFAR-10 [16] dataset. As it is shown, the model can produce perfect outputs even for gray-scale inputs.

Results on various datasets

Dataset	Method	Mean
MNIST [37]	ARAE [15]	97.5
	OCSVM [6]	96.0
	AnoGAN [31]	91.3
	DSVDD [5]	94.8
	CapsNet$_{PP}$ [41]	97.7
	OCGAN [30]	97.5
	LSA [29]	97.5
	OURS	**98.00**
Fashion-MNIST [42]	ARAE [15]	**93.6**
	OCSVM [6]	92.8
	DAGMM [43]	41.7
	DSEBM [44]	85.5
	DSVDD [5]	92.8
	LSA [29]	92.2
	OURS(4-parts)	92.26
	OURS(9-parts)^3	93.00
CIFAR-10 [16]	ARAE [15]	60.23
	OCSVM [6]	58.56
	AnoGAN [31]	61.79
	DSVDD [5]	64.81
	CapsNet$_{PP}$ [41]	61.2
	OCGAN [30]	65.66
	LSA [29]	64.1
	OURS	**72.47**

Effect of each component

TABLE 9: Effect of each component or algorithm is illustrated separately. As it is shown, CPAE-G has the best results on sample dataset CIFAR-10 [16].

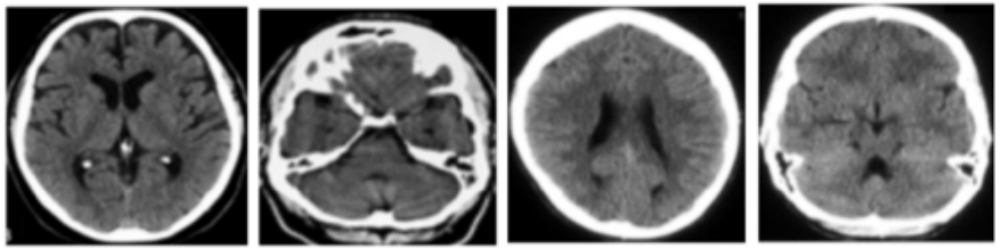
		0	1	2	3	4	5	6	7	8	9	mean
MIN	puzzle AE (PAE)	76.32	69.69	68.70	54.08	75.30	62.91	72.72	69.61	80.87	64.88	69.51
	colorization + puzzle (CPAE)	79.51	69.77	68.51	54.75	72.56	63.24	67.86	68.30	82.09	65.57	69.22
	colorization + puzzle + GAN (CPAE-G)	79.42	73.00	69.48	53.00	73.98	65.13	68.98	70.65	83.28	66.73	70.37
MAX	puzzle AE (PAE)	76.29	69.07	68.19	52.14	75.84	60.84	73.66	68.61	78.26	66.20	68.91
	colorization + puzzle (CPAE)	78.87	76.56	67.97	54.33	74.69	62.32	75.72	71.59	81.06	70.92	71.40
	colorization + puzzle + GAN (CPAE-G)	77.21	77.31	69.3	54.10	74.76	64.10	76.02	70.42	81.04	66.91	71.12
AVG	puzzle AE (PAE)	76.59	68.84	68.54	53.00	76.00	61.8	73.32	68.87	79.54	66.12	69.26
	colorization + puzzle (CPAE)	79.72	75.86	68.56	54.74	74.87	64.21	74.02	72.97	82.64	70.62	71.82
	colorization + puzzle + GAN (CPAE-G)	78.93	78.05	69.95	54.88	75.46	66.04	74.76	73.30	83.34	69.96	**72.47**

Results on the QC dataset: MVTec AD

Method	Bottle	Hazelnut	Capsule	Metal Nut	Leather	Pill	Wood	Carpet
AVID [39]	88.0	86.0	85.0	63.0	58.0	86.0	83.0	70.0
AE_{SSIM} [50]	88.0	54.0	61.0	54.0	46.0	60.0	83.0	67.0
AE_{L2} [50]	80.0	88.0	62.0	73.0	44.0	62.0	74.0	50.0
AnoGAN [31]	69.0	50.0	58.0	50.0	52.0	62.0	68.0	49.0
LSA [29]	86.0	80.0	71.0	67.0	70.0	85.0	75.0	74.0
OURS	94.24 ± 0.10	91.21 ± 0.13	66.88 ± 0.23	66.33 ± 0.10	72.86 ± 0.86	71.63 ± 0.11	89.51 ± 0.63	65.73 ± 0.37

Method	Tile	Grid	Cable	Transistor	Toothbrush	Screw	Zipper	Mean
AVID [39]	66.0	59.0	64.0	58.0	73.0	66.0	84.0	73.0
AE_{SSIM} [50]	52.0	69.0	61.0	52.0	74.0	51.0	80.0	63.0
AE_{L2} [50]	77.0	78.0	56.0	71.0	98.0	69.0	80.0	71.0
AnoGAN [31]	51.0	51.0	53.0	67.0	57.0	35.0	59.0	55.0
LSA [29]	70.0	54.0	61.0	50.0	89.0	75.0	88.0	73.0
OURS	65.48 ± 0.12	75.35 ± 0.60	87.90 ± 0.10	85.96 ± 0.12	97.79 ± 0.05	57.81 ± 1.12	75.74 ± 0.13	**77.63**

Results on Medical Imaging Datasets

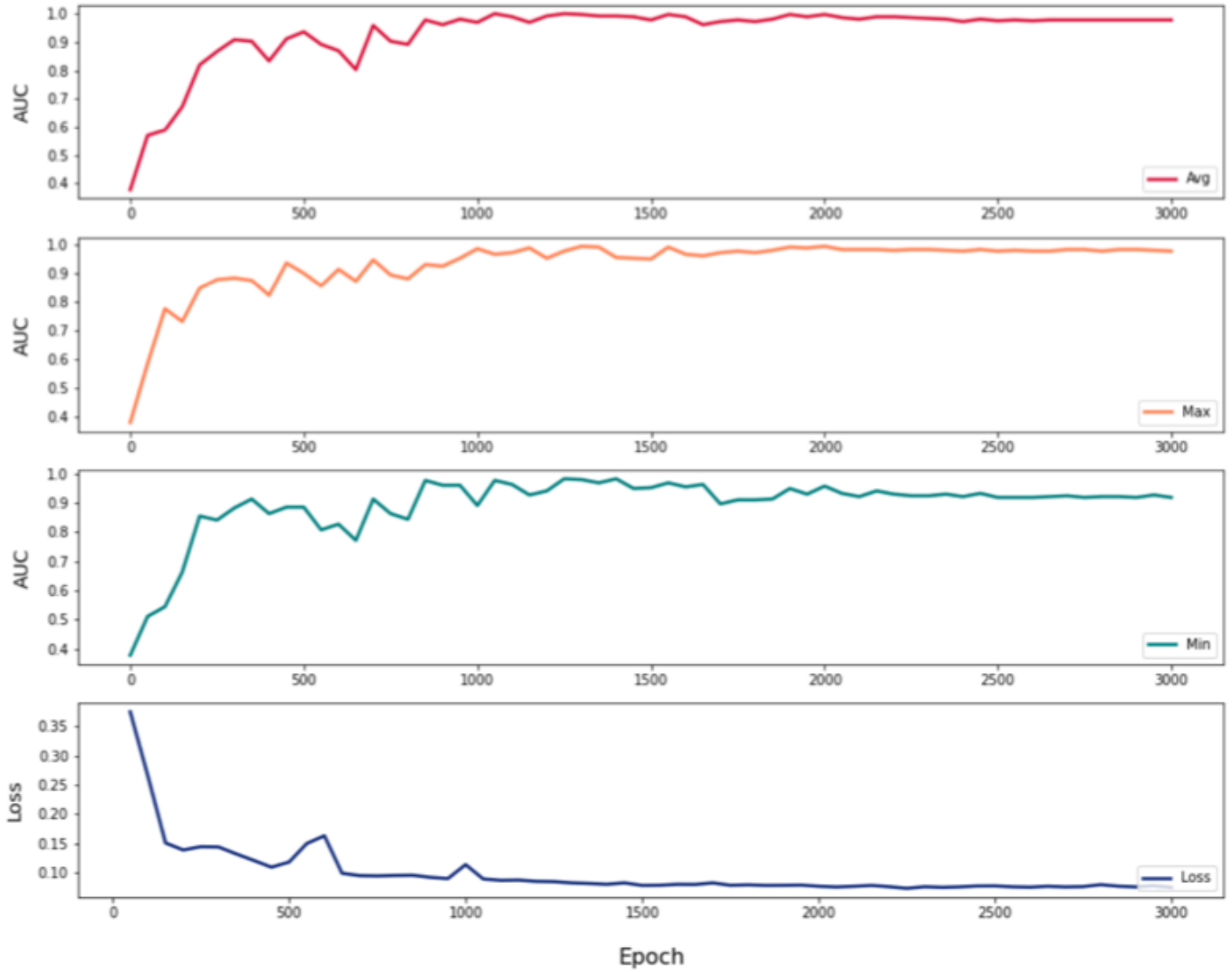


		LSA* [29]	OCGAN* [30]	OURS
head_ct_hemorrhage	AUC	81.67 ± 0.358	51.22 ± 3.626	$\mathbf{86.43 \pm 0.04}$
	FPR	0.81	1.00	**0.70**
brain_tumor_hemorrhage	AUC	95.61 ± 1.433	91.74 ± 3.050	$\mathbf{96.34 \pm 0.031}$
	FPR	**0.40**	0.60	0.50

Generality of the method

	MNIST [37]	CIFAR-10 [16]	MVTec [23]	head_ct_hemorrhage	brain_tumor_hemorrhage
GT [22]	**98.00**	**82.30**	67.20	44.70*	82.07*
OURS	**98.00**	72.47	**77.63**	**86.43**	**96.34**

Stability of the method



Robustness of the model

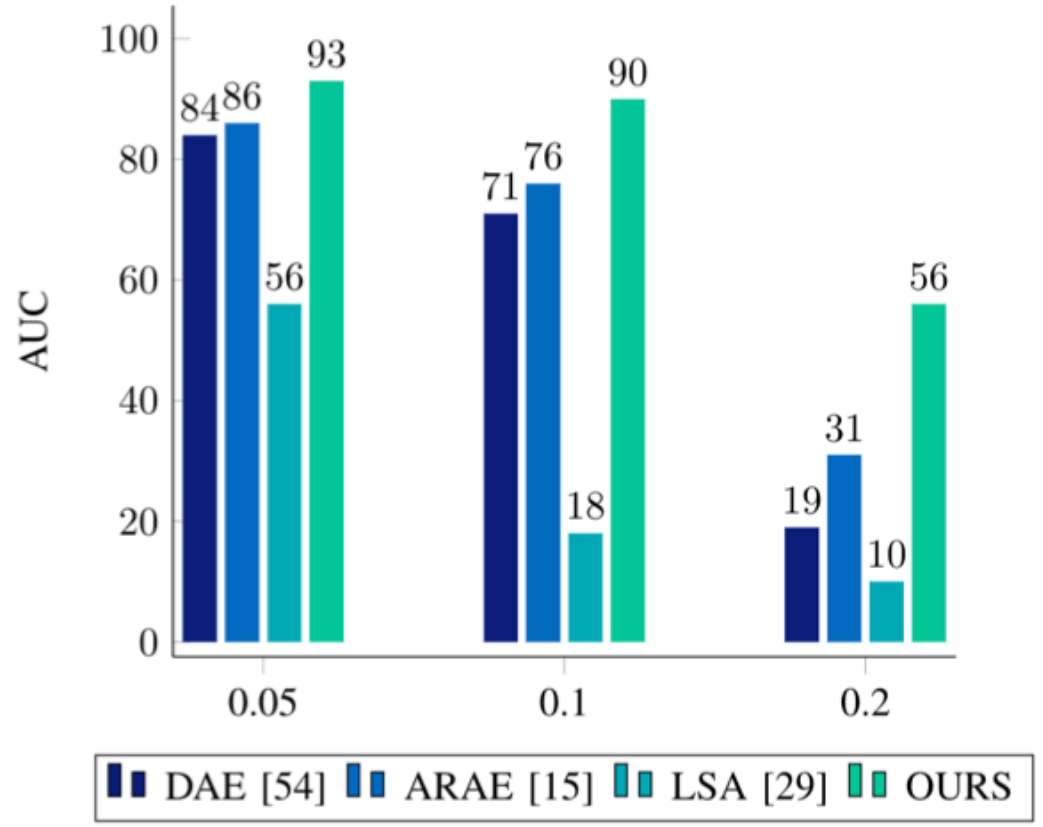
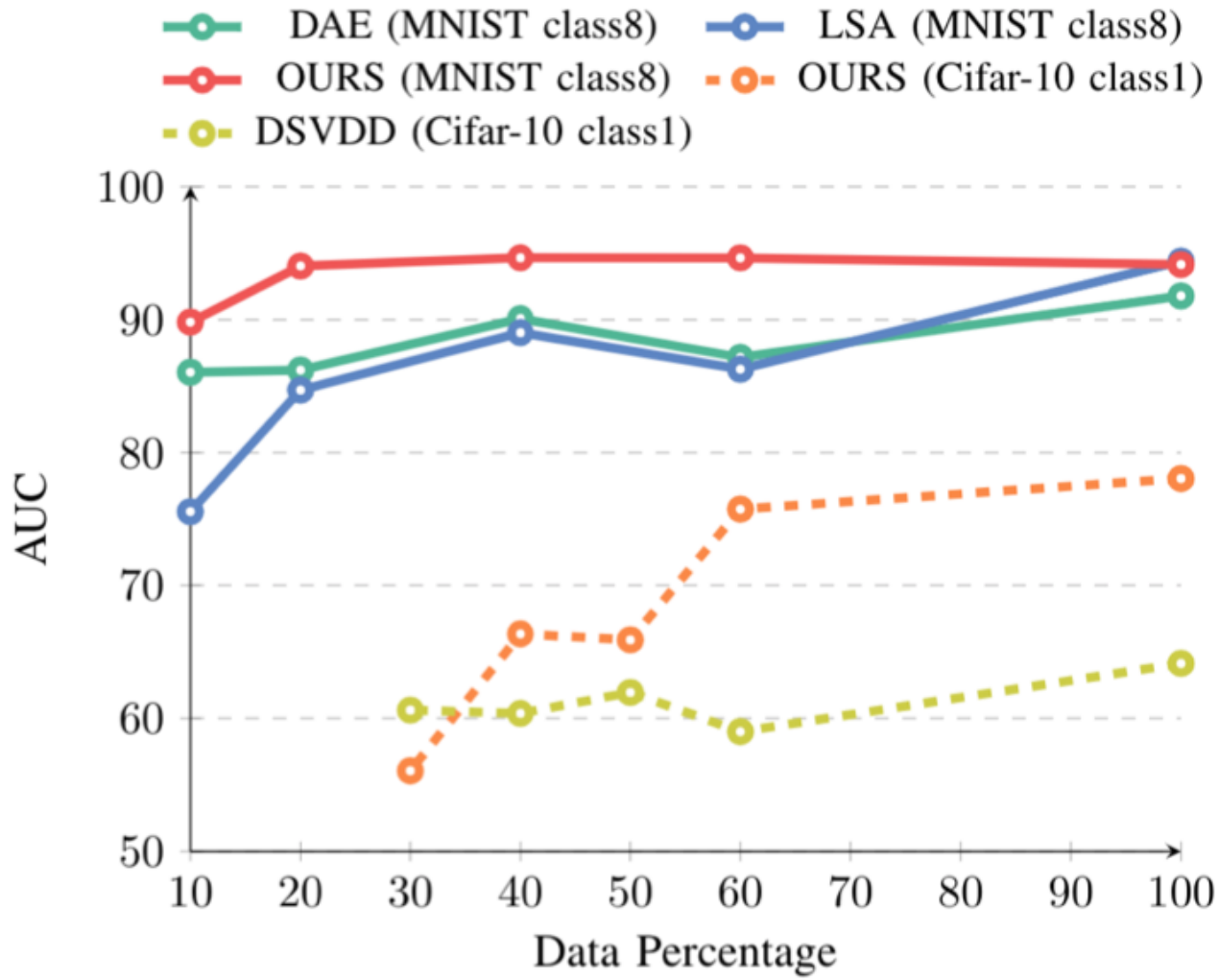


Figure 10: Robustness to adversarial attack on normal data at testing time. The results are shown for three different ϵ and the model is trained on the class 8 of the MNIST [37] dataset. Clearly, Puzzle AE is significantly more robust than other SOTAs on the most challenging class of the MNIST [37] dataset.

Data efficiency of the method



A funded Postdoctoral position available at my lab.

Email to me if you are interested:

rohban@sharif.edu

Thanks!

Any questions?