Model Selection for Self-supervised Anomaly Detection

Jaemin Yoo

School of Electrical Engineering

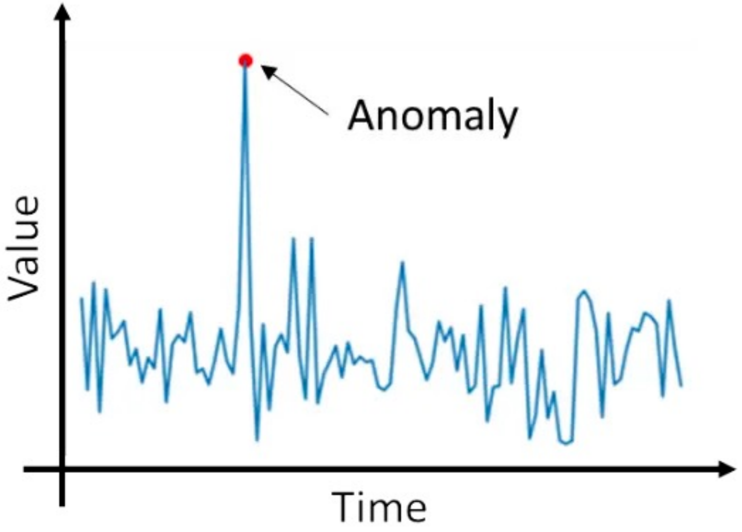
KAIST

Outline

1. **<u>Introduction</u>**
2. Preliminaries: Benchmark Study
3. Approach 1: Offline Selection
4. Approach 2: End-to-end Learning
5. Conclusion

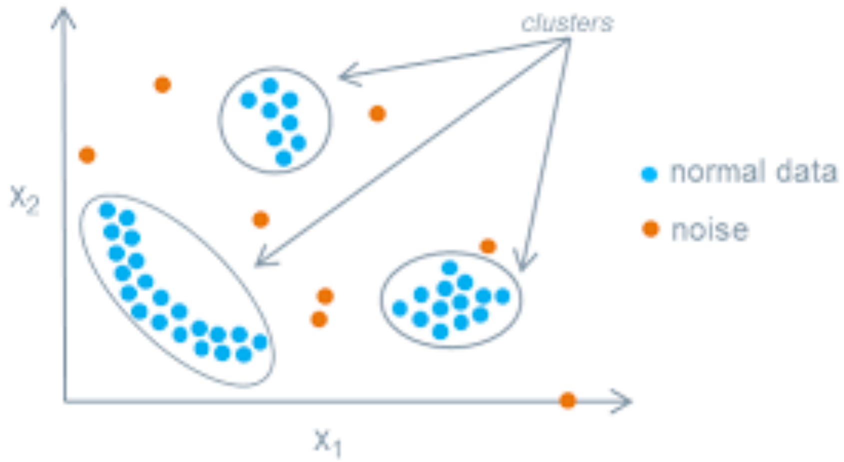
Anomaly Detection

- **Anomaly detection (AD)** is a problem to find **anomalies** from data.
  - Anomalies are common in many real-world datasets.
  - Essential to improve the reliability and efficiency of a system.



Anomalies in a time series

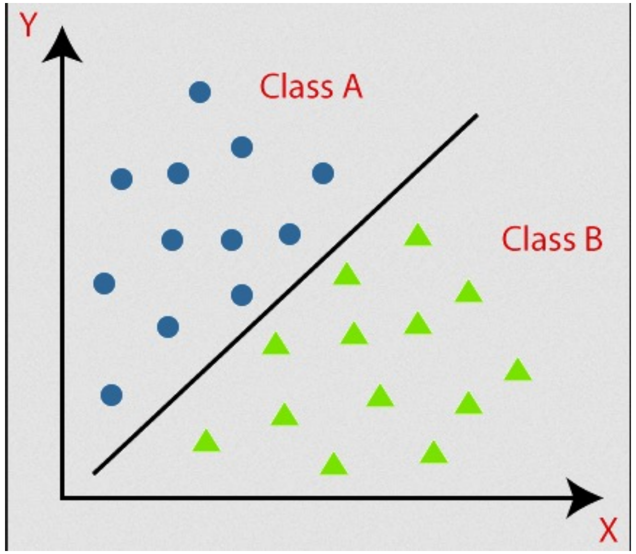
A red fish as an anomaly



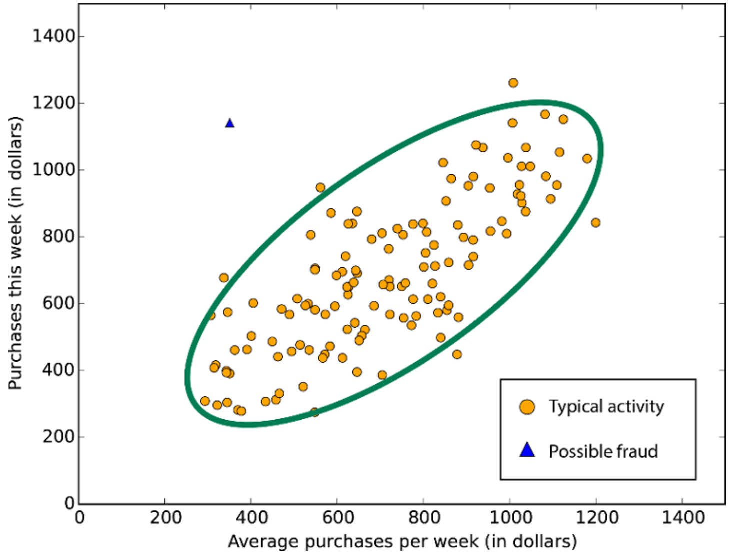
Outliers in a data distribution

Challenges of Anomaly Detection

- **Challenge 1:** Training labels are insufficient or even **nonexistent**.
- **Challenge 2:** Anomalies are **scattered** without creating a cluster.



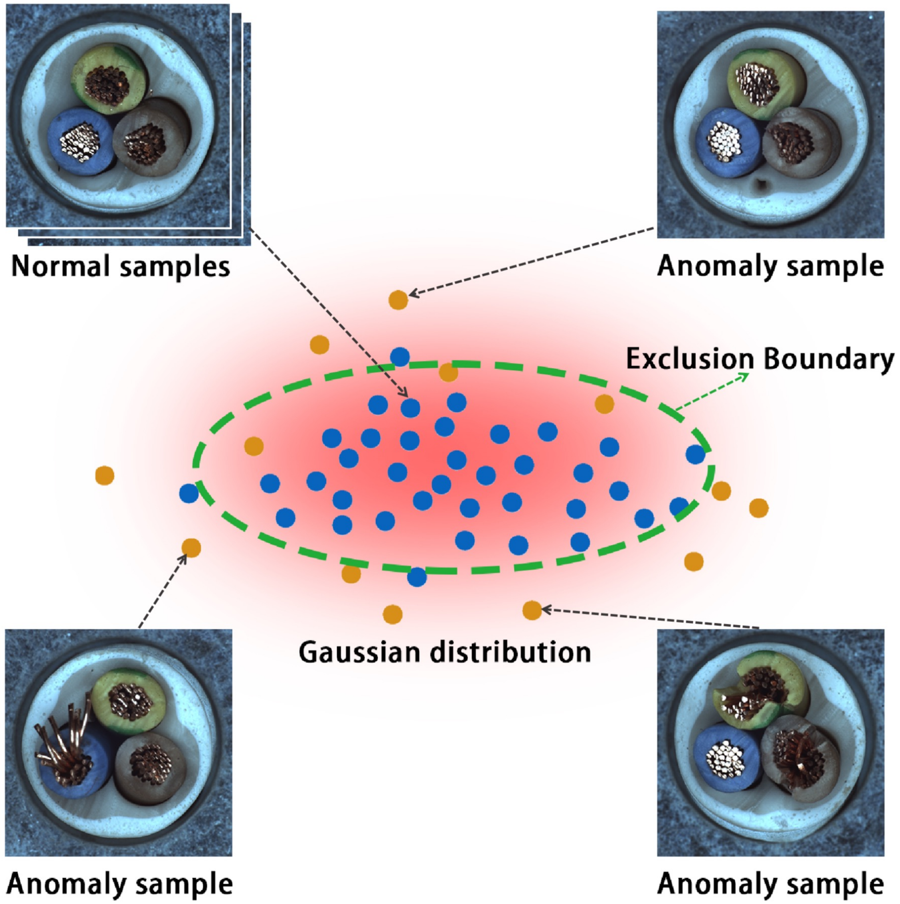
Typical binary classification



Anomaly detection

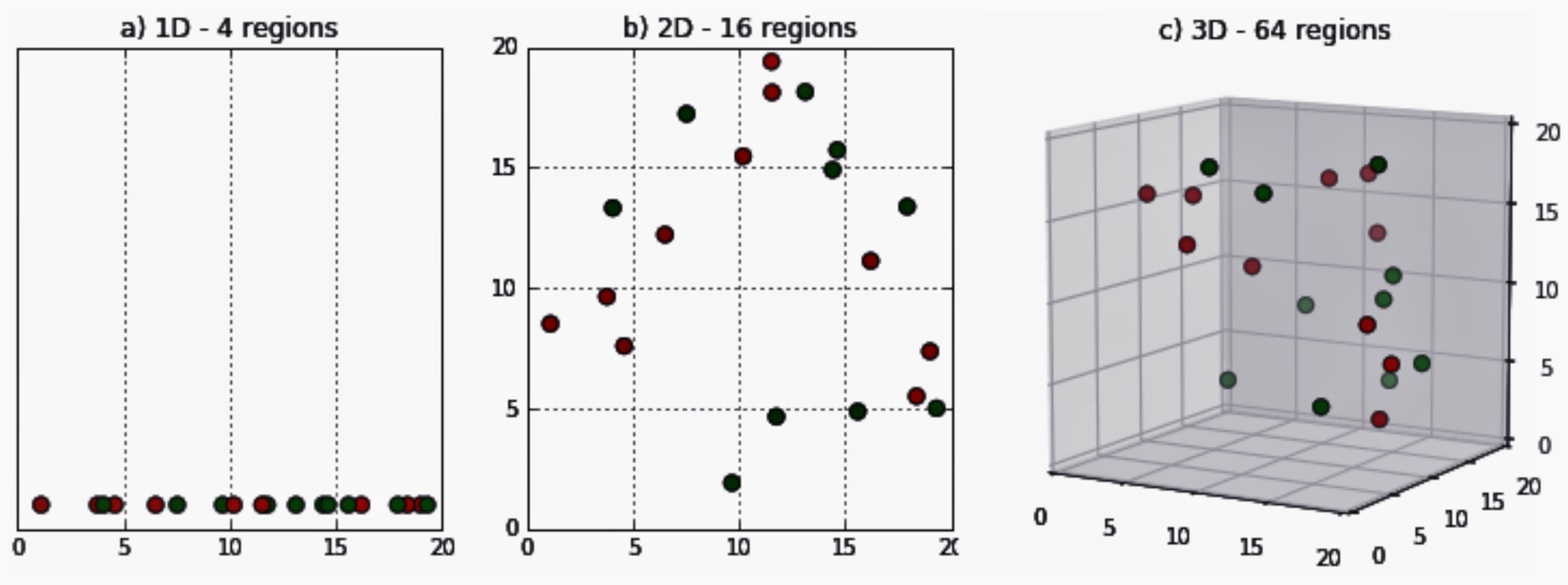
Unsupervised Solutions

- **Traditional approaches** on AD rely on **unsupervised learning**
  1. Find a probability distribution p_X
     - that describes normal data.
  2. A point x is anomalous if
     - it is far from the center (i.e., low $p_X(x)$).
- Also known as *density estimation*.



Limitations of Unsupervised Solutions

- **Limitation 1:** Need a lot of data samples to accurately learn $p_X(x)$.
  - This is problematic especially with **high dimensionality**.
  - Also known as the *curse of dimensionality*; # of data samples $\propto 2^D$.



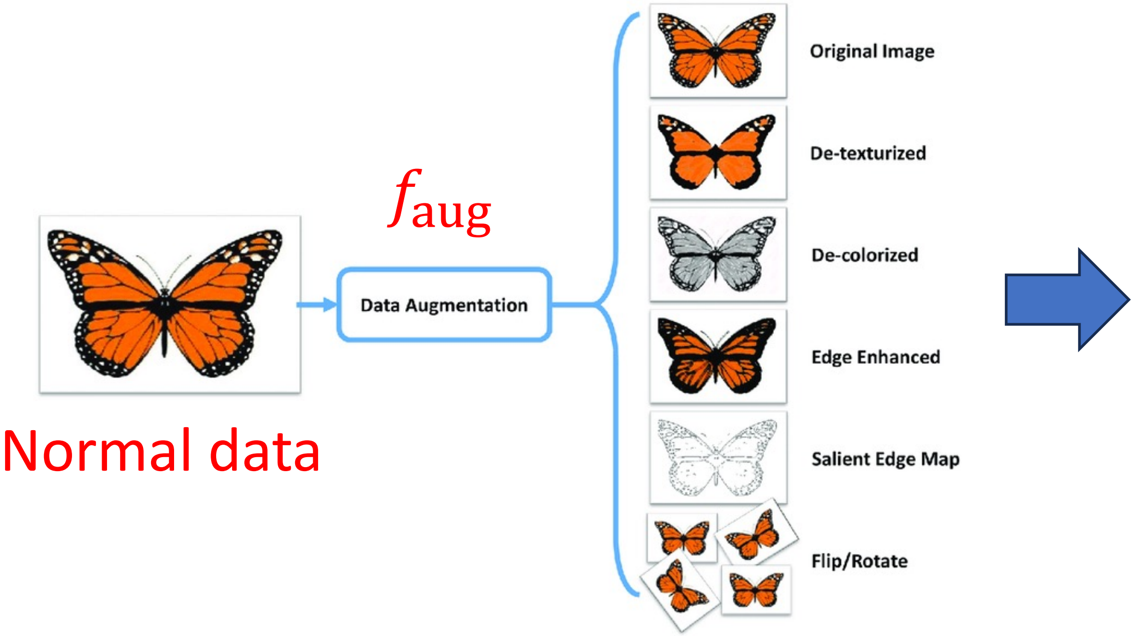
Limitations of Unsupervised Solutions

- **Limitation 2:** It is hard to incorporate domain knowledge of data.
  - What if we have **some knowledge** on how anomalies look like?
  - What if we know that there can be some cracks on the surface of a pill?

Limitations 1 & 2 suggest we need a *better paradigm* for AD!

Self-supervised Anomaly Detection

- **Self-supervised anomaly detection (SSAD)** can be the future.
  - **Idea:** Train a classifier that can detect **pseudo anomalies** from the inliers.
  - Put differently, train a classifier that can detect the **artificial differences**.



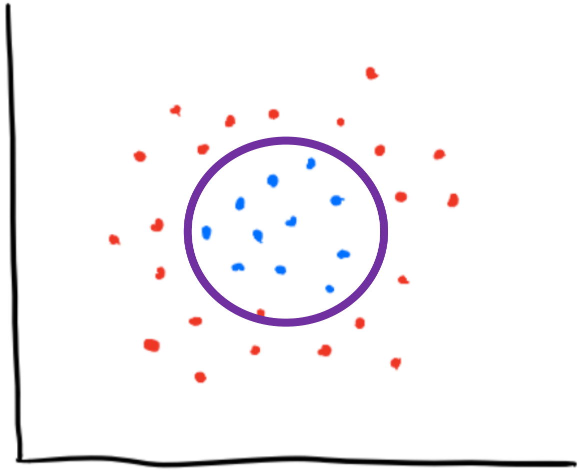
Treat as anomalies!
(learn the differences)

Example of SSAD

- **How to train a detector with SSAD:**
  1. Suppose we have a (training) set $\mathcal{D}_{\text{trn}}$ of normal data.
  2. Create an augmented set $\mathcal{D}_{\text{aug}} = \{f_{\text{aug}}(x) \mid x \in \mathcal{D}_{\text{trn}}\}$.
  3. Create a labeled dataset $\mathcal{D}_{\text{new}} = \{(x, +1) \mid x \in \mathcal{D}_{\text{trn}}\} \cup \{(x, -1) \mid x \in \mathcal{D}_{\text{aug}}\}$.
  4. Train a **binary classifier** ϕ from $\mathcal{D}_{\text{new}}$.
- **How to actually use ϕ in test time:**
  - For each test data $x \in \mathcal{D}_{\text{test}}$, we say that x is an anomaly if $\phi(x) \approx -1$.

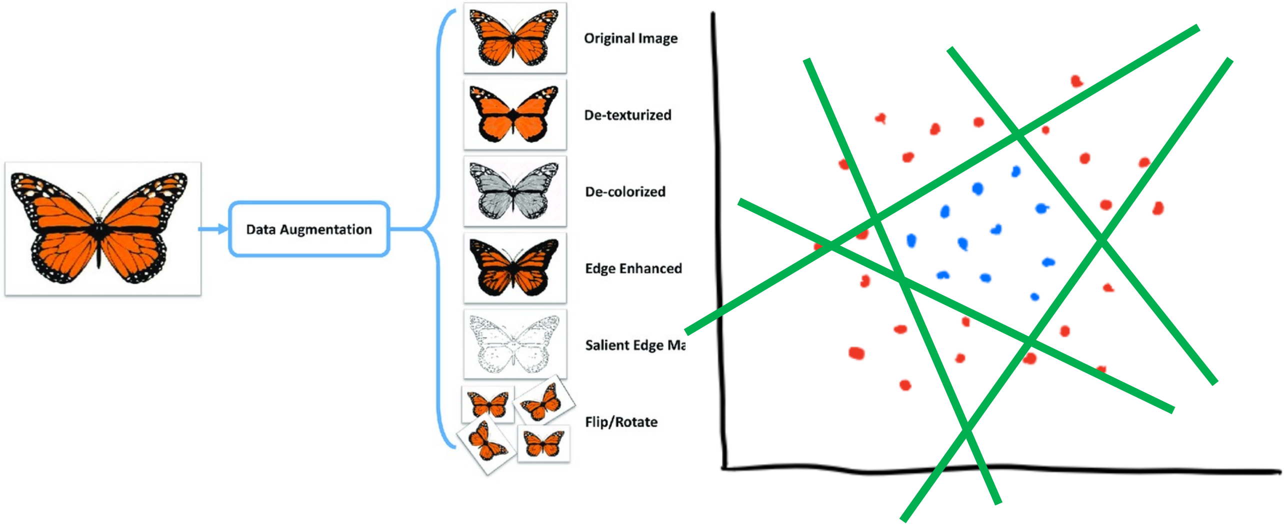
Meaning of Self-supervision

- **Density estimation** is *filling in the space* with dots (= data).
- **Self-supervision** is *drawing many boundaries* around the data.
  - It is less sensitive to the data dimensionality.



Unsupervised approach

VS

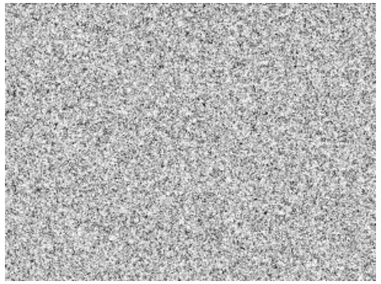


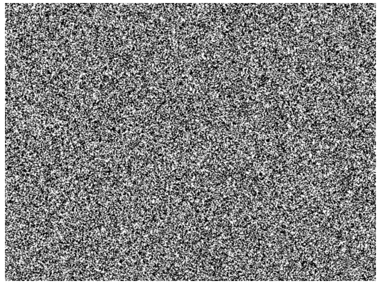
Self-supervised approach with data augmentation

Why SSAD is Successful

- SSAD allows us to focus on a **plausible subspace**.
  - It is not necessary to consider every possible data $x \in \mathbb{R}^D$.
  - E.g., we won't expect white noise as an actual anomaly.
  - Only a few possible types of pseudo anomalies are enough.

vs.





/

Research Motivation

- Note that SSAD contains many important *hyperparameters*.
  - The augmentation function f_{aug}, the objective function, etc.
  - Hyperparameter choice determines the success of SSAD.
- However, **model selection** is especially challenging in SSAD.
  - Since no labeled validation data are given.

Q: How can we perform HP search on SSAD without labels?

Outline

1. Introduction
2. <u>**Preliminaries: Benchmark Study**</u>
3. Approach 1: Offline Selection
4. Approach 2: End-to-end Learning
5. Conclusion

Research Goal

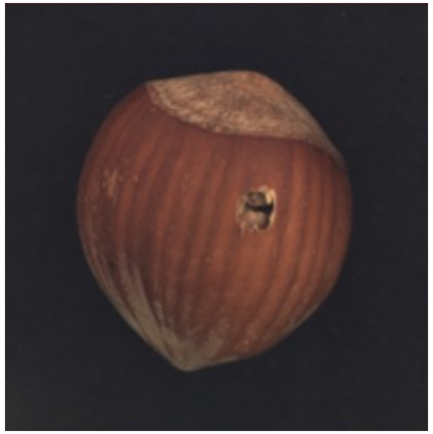
- **Goal:** Study how important the choice of f_{aug} is on SSAD.
- **Idea:** Introduce the **alignment** between f_{gen} and f_{aug}.
  - f_{gen} is the internal anomaly-generating function in test data.
  - E.g., if x is a normal image, $f_{\mathrm{gen}}(x)$ is an anomalous image.
- **Hypothesis:** SSAD works better if f_{gen} and f_{aug} are aligned well.

Anomaly-Generating Function

- f_{gen} transforms a normal sample x into an anomaly $f_{\text{gen}}(x)$.
  - Hard to formally define in real data.

Normal data

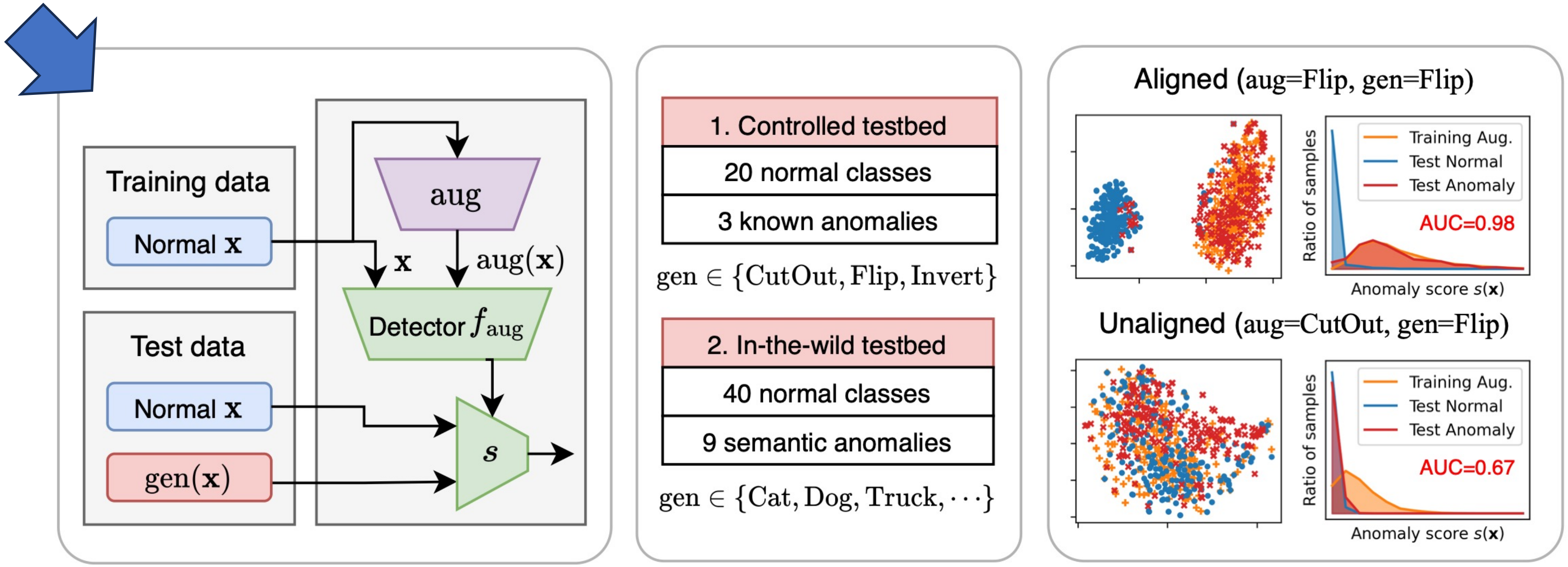
f_{gen}



Anomaly

Overview

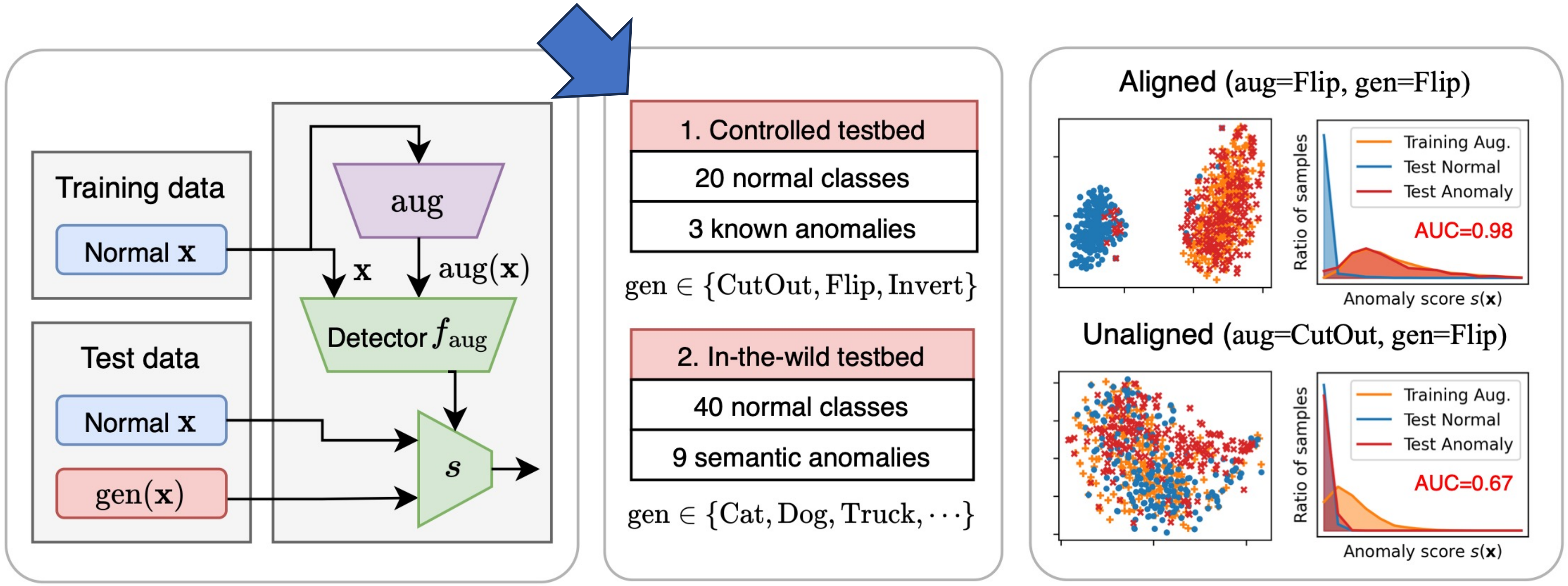
- **(Left) Simple illustration on how SSAD works:**
  - The score function s is used to produce the final output.



(a) SSAD Framework (b) Testbeds (c) Main finding: alignment matters

Overview

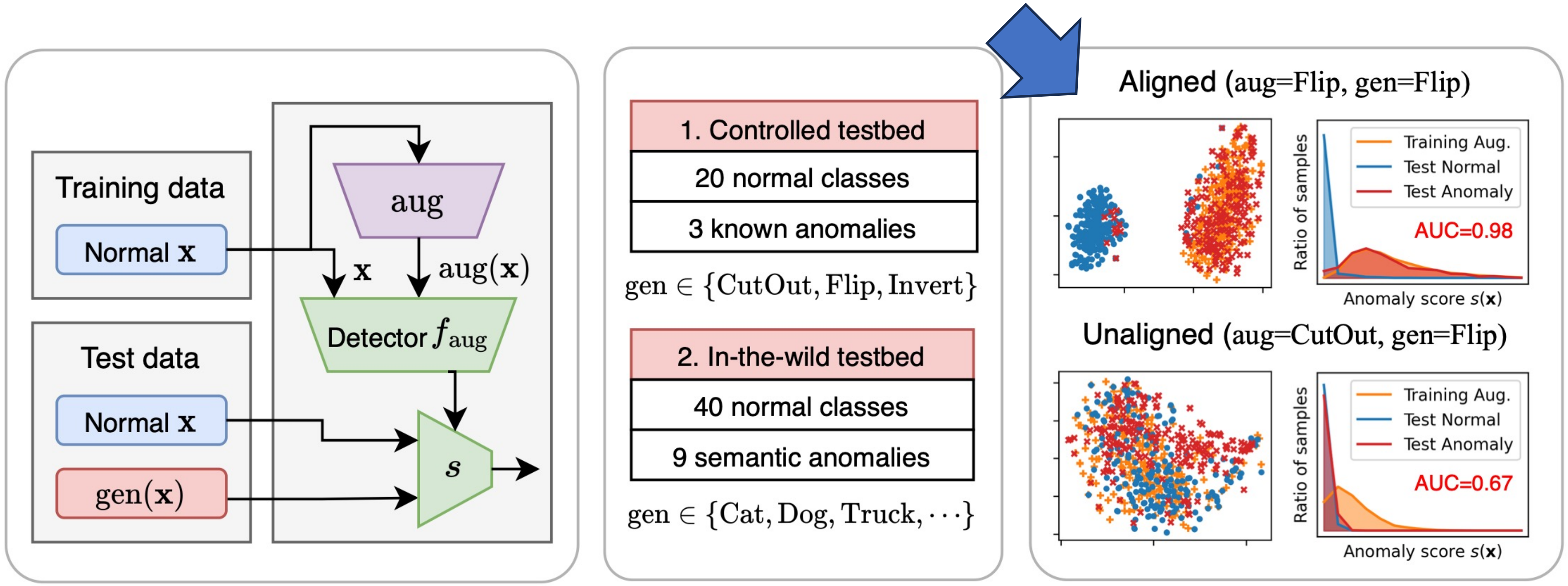
- **(Center) Two testbeds:** **Controlled** and **in-the-wild** testbeds.
  - We create anomalies with known f_{gen} in the controlled testbed.



(a) SSAD Framework (b) Testbeds (c) Main finding: alignment matters

Overview

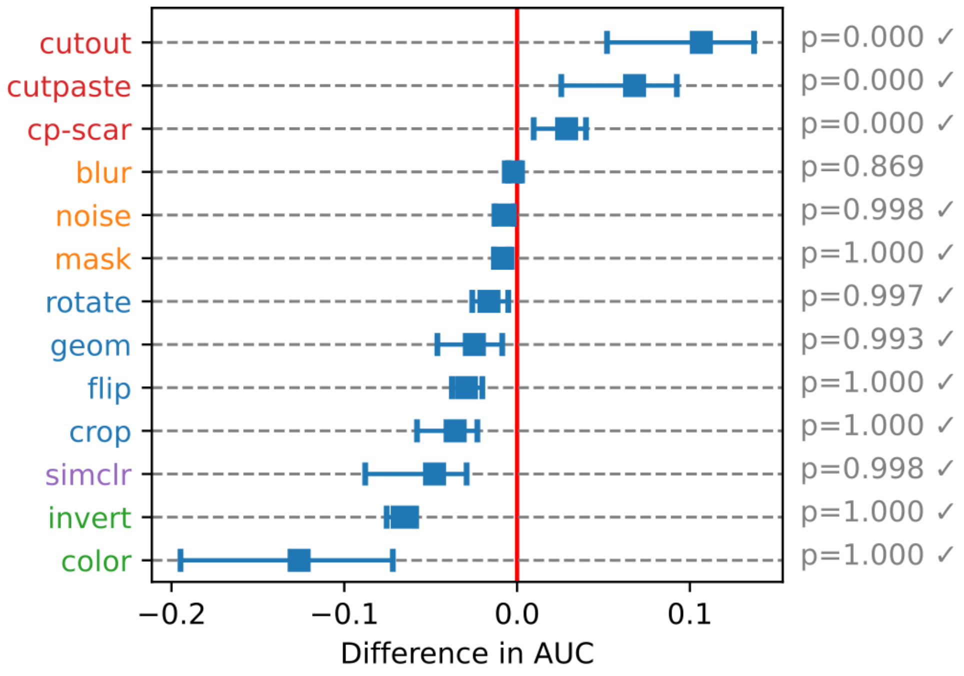
- **(Right) Test AUC is high with the high alignment.**
  - Embeddings of augmented data are test anomalies are matched.



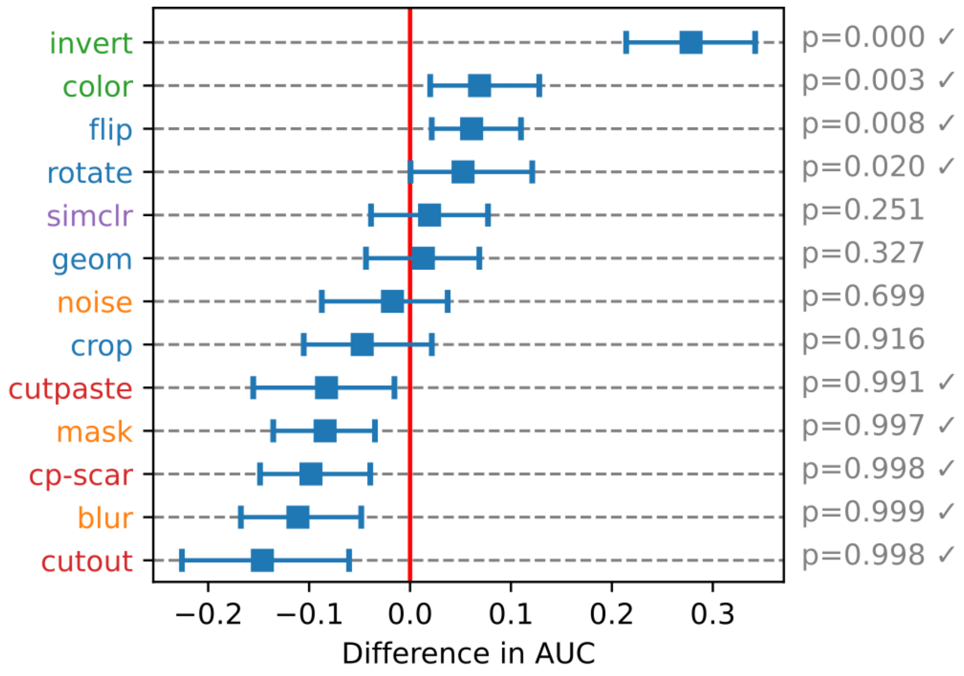
(a) SSAD Framework (b) Testbeds (c) Main finding: alignment matters

Main Result

- Performance is affected a lot by the semantic alignment with f_{gen}.



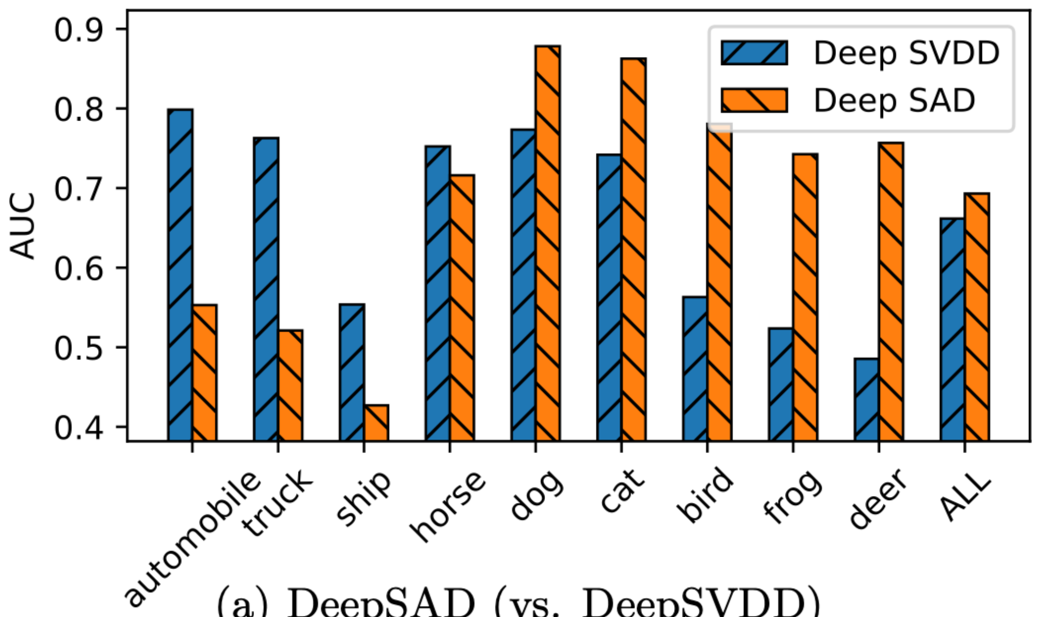
(a) DAE on **gen**:=CutOut



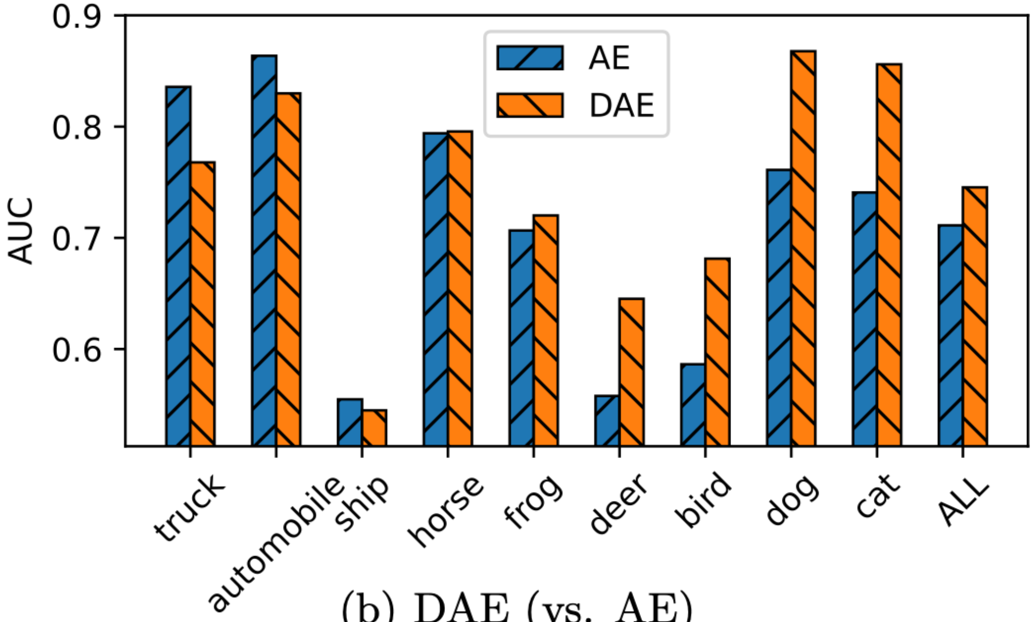
(b) DeepSAD on **gen**:=Invert

Bias in Prediction

- Self-supervision creates a bias in the prediction distribution.
  - Airplane as the inlier, and Rotation as f_{aug}.



(a) DeepSAD (vs. DeepSVDD)



(b) DAE (vs. AE)

Outline

1. Introduction
2. Preliminaries: Benchmark Study
3. **Approach 1: Offline Selection**
4. Approach 2: End-to-end Learning
5. Conclusion

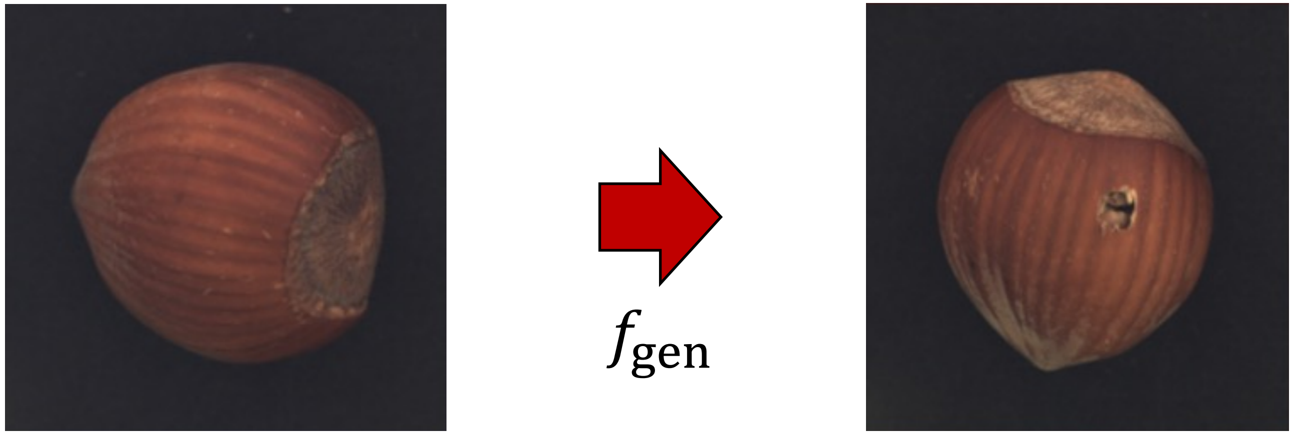
Research Question

- **Problem:** We focus on **transductive** anomaly detection.
- **Given:**
  1. Normal-only training data $\mathcal{D}_{\mathrm{trn}}$.
  2. Set $\{\phi_i\}_i$ of detectors trained with f_{aug} having different HPs.
  3. **Unlabeled test data** $\mathcal{D}_{\mathrm{test}}$ containing normal data and anomalies.
- **Goal:** Find a loss function $\mathcal{L}$ such that

$$\phi^* = \mathrm{argmin}_{\phi}\, \mathcal{L}(\cdot) \text{ is the best detector.}$$

Motivation 1

- **Idea 1:** Let's measure the alignment between f_{aug} and f_{gen}.
  - Then, we will select the detector with the best alignment.
  - We know from the preliminary work that alignment ≈ accuracy.

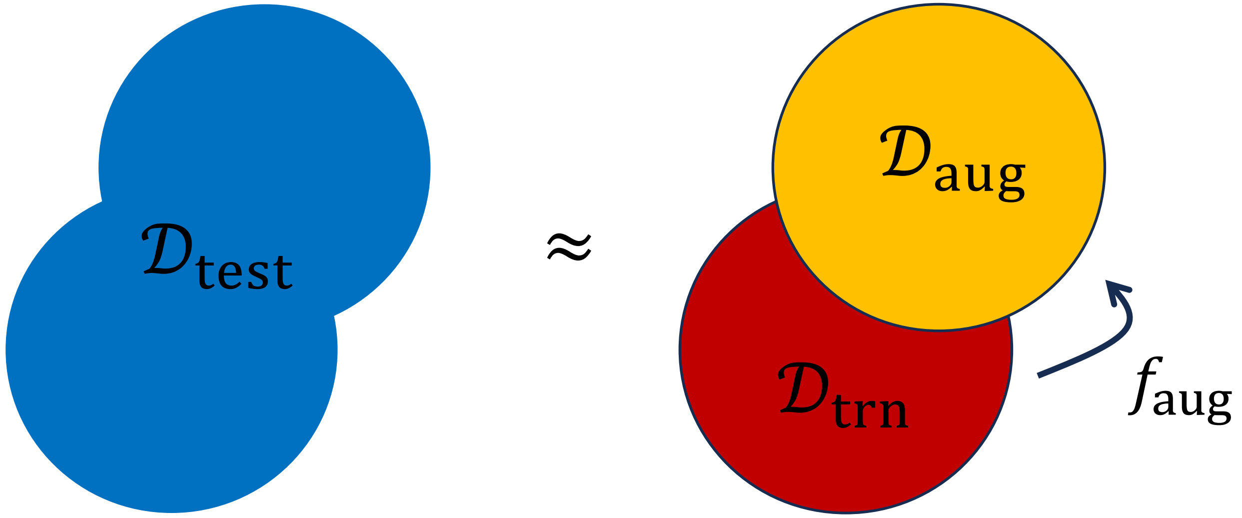


Normal data

Anomaly

Motivation 2

- **Idea 2:** Let's utilize the assumption of transductive learning:
  - We are given unlabeled test data $\mathcal{D}_{\text{test}}$ at training.
  - Suppose that f_{aug} and f_{gen} are aligned well.
  - Then, $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{test}}^{\text{n}} \cup \mathcal{D}_{\text{test}}^{\text{a}}$ should be similar to $\mathcal{D}_{\text{trn}} \cup \mathcal{D}_{\text{aug}}$.



Main Approach

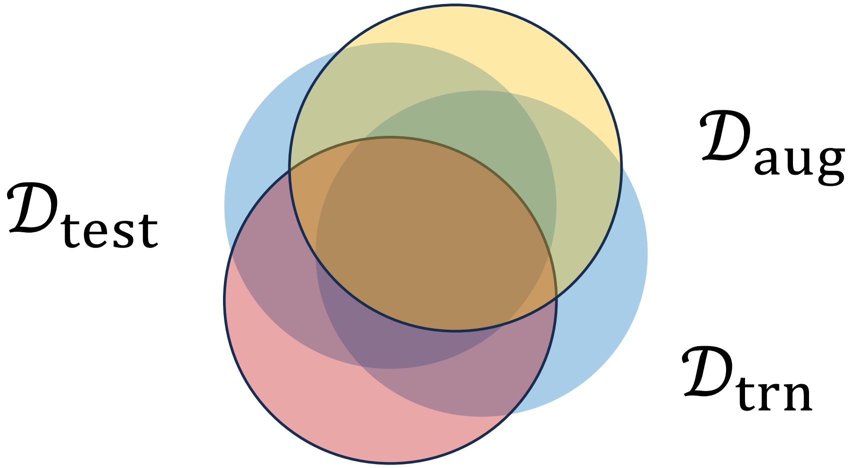
- **Approach:** Let's approximate the alignment with $\tilde{\mathcal{L}}$:

$$\boxed{\tilde{\mathcal{L}}(\alpha) = d\left(\mathcal{Z}_{\mathrm{trn}} \cup \mathcal{Z}_{\mathrm{aug}}, \mathcal{Z}_{\mathrm{test}}\right)}$$

  - α is a set of hyperparameters (HPs) which we want to evaluate.
  - $d(\cdot,\cdot)$ is a distance function between sets.
  - $\mathcal{Z}$ refers to the set of embeddings created with hyperparameters α.
- Smaller $\tilde{\mathcal{L}}(\alpha)$ represents that α makes better alignment.

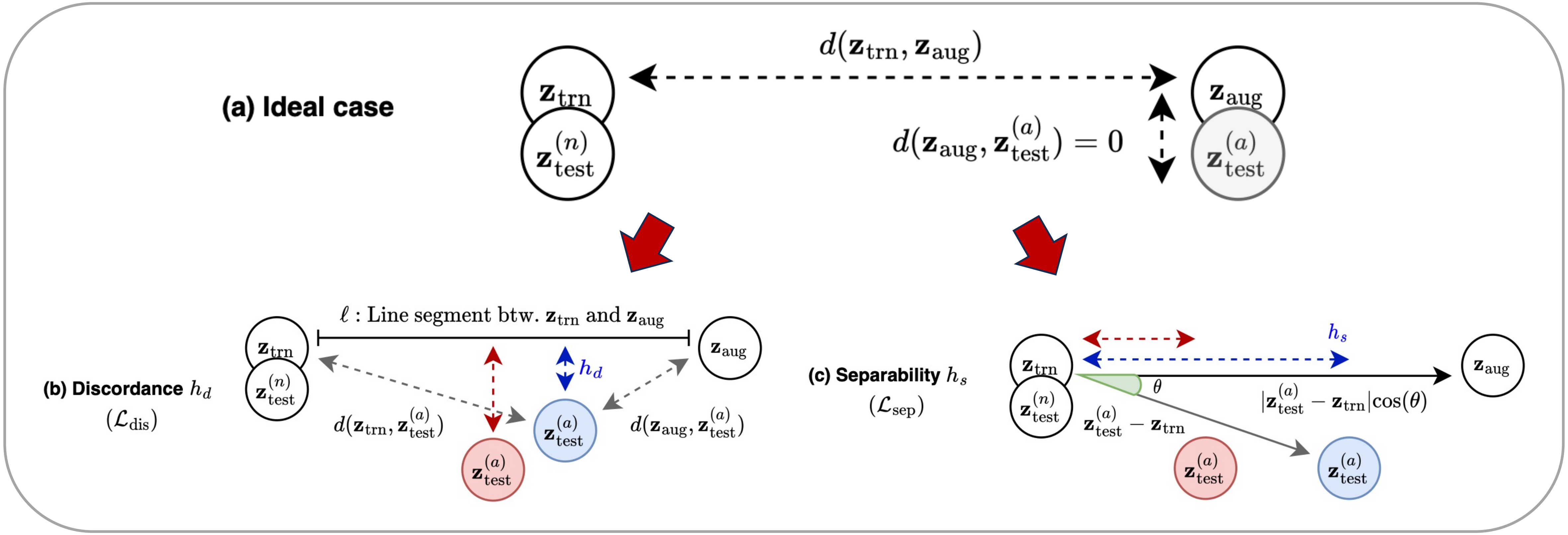
Important Limitation

- The problem is that there can be **false positives**:
  - If f_{aug} and f_{gen} are aligned well, then $\tilde{\mathcal{L}}$ should be small.
  - If $\tilde{\mathcal{L}}$ is small, then f_{aug} and f_{gen} are *not necessarily aligned*.
- One example is when everything is mixed around.



Final Approach

- We study how to make $\tilde{\mathcal{L}}$ more accurate by avoiding false positives.



Experiments

- Average AUC and rank across 21 different tasks in 2 datasets.
- Our DSV outperforms all competitors in 6 of the 8 cases.

AUC:

f_{aug}	Avg.	Rand.	Base	MMD	STD	MC	SEL	HITS	**DSV**
CutOut	0.739	<u>0.776</u>	0.741	0.735	0.739	0.749	0.727	0.757	**0.813**
CutAvg	0.739	**0.817**	0.721	0.692	0.745	0.751	0.744	0.742	<u>0.806</u>
CutDiff	0.743	0.711	0.739	0.730	0.744	0.747	0.741	<u>0.777</u>	**0.811**
CutPaste	0.788	0.841	0.694	0.756	0.818	<u>0.862</u>	0.830	0.850	**0.884**

Rank:

f_{aug}	Avg.	Rand.	Base	MMD	STD	MC	SEL	HITS	**DSV**
CutOut	7.33	6.10	6.62	6.93	6.29	6.50	7.10	<u>5.43</u>	**3.79**
CutAvg	7.00	<u>5.02</u>	7.64	8.36	5.52	5.48	5.98	5.60	**4.19**
CutDiff	6.43	7.24	6.45	7.38	6.00	<u>5.64</u>	6.24	6.21	**3.60**
CutPaste	7.67	6.29	8.67	7.21	5.60	**4.33**	5.17	4.64	<u>4.57</u>

Outline

1. Introduction
2. Preliminaries: Benchmark Study
3. Approach 1: Offline Selection
4. **Approach 2: End-to-end Learning**
5. Conclusion

Research Question

- DSV has a limitation as an **offline evaluation measure.**
  - We need to train all possible N models before doing selection.
- **Q:** How can we design a framework for end-to-end learning?
- **What we need:**
  1. *Differentiable validation loss* that measures the alignment.
  2. *Differentiable augmentation functions.*
  3. *Implementation techniques* that make everything possible.

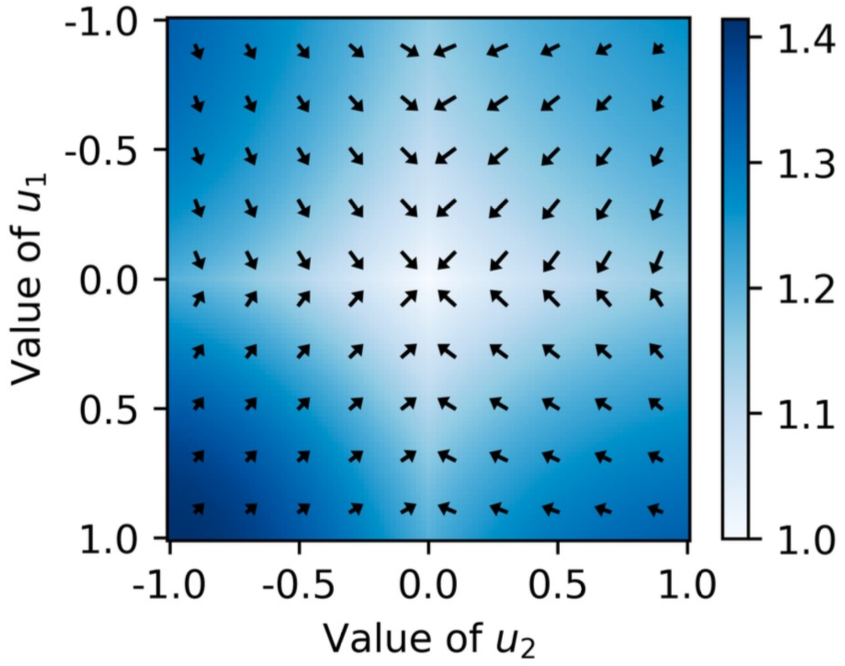
New Validation Loss

- DSV sub-optimal with respect to end-to-end optimization.
  - What we need is not just differentiability.
  - The loss function should be smooth and able to lead to local optima.
- We design a much simpler loss:

$$\mathcal{L}_{\mathrm{val}}(\mathcal{Z}_{\mathrm{trn}}, \mathcal{Z}_{\mathrm{aug}}, \mathcal{Z}_{\mathrm{test}}) = \frac{1}{2} \sum_{\mathbf{z}' \in \mathcal{Z}'_{\mathrm{test}}} \|\mathbf{z}' - \mathrm{mean}(\mathcal{Z}'_{\mathrm{trn}})\|_2 + \|\mathbf{z}' - \mathrm{mean}(\mathcal{Z}'_{\mathrm{aug}})\|_2 \ ,$$

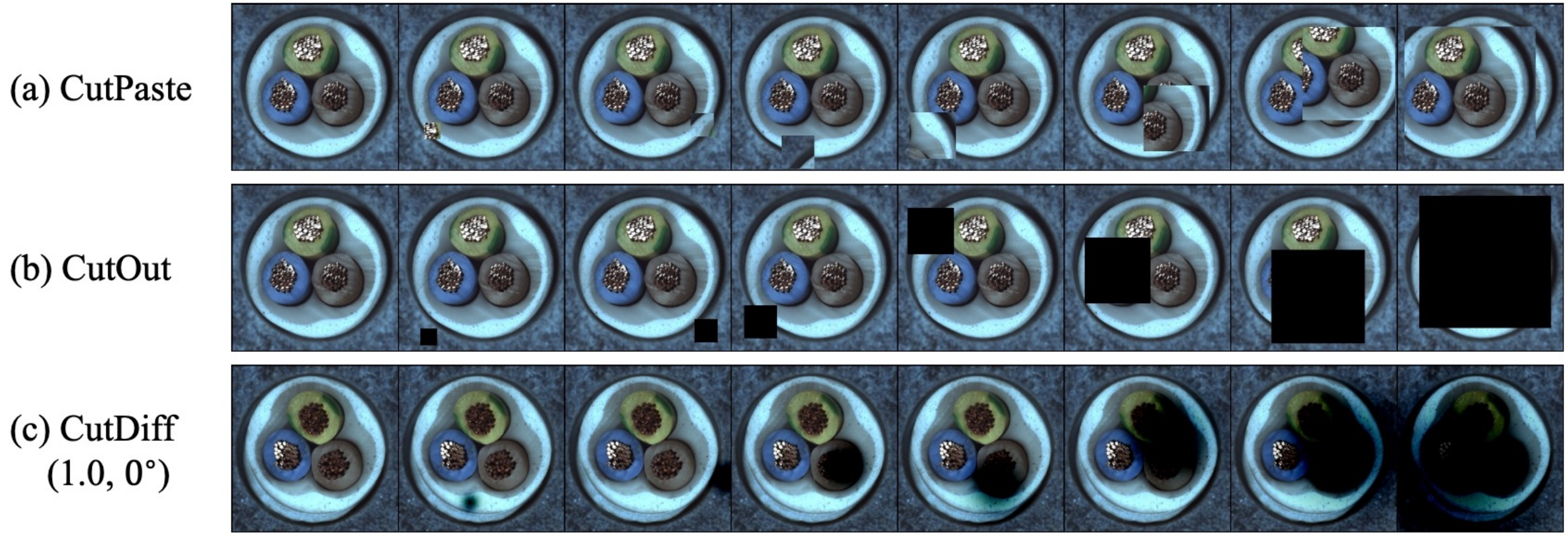
Nice Property of the Validation Loss

- The (negative) gradients nicely point to a local optimum.
  - u_1 and u_2 are parameters in this example that determine the alignment.



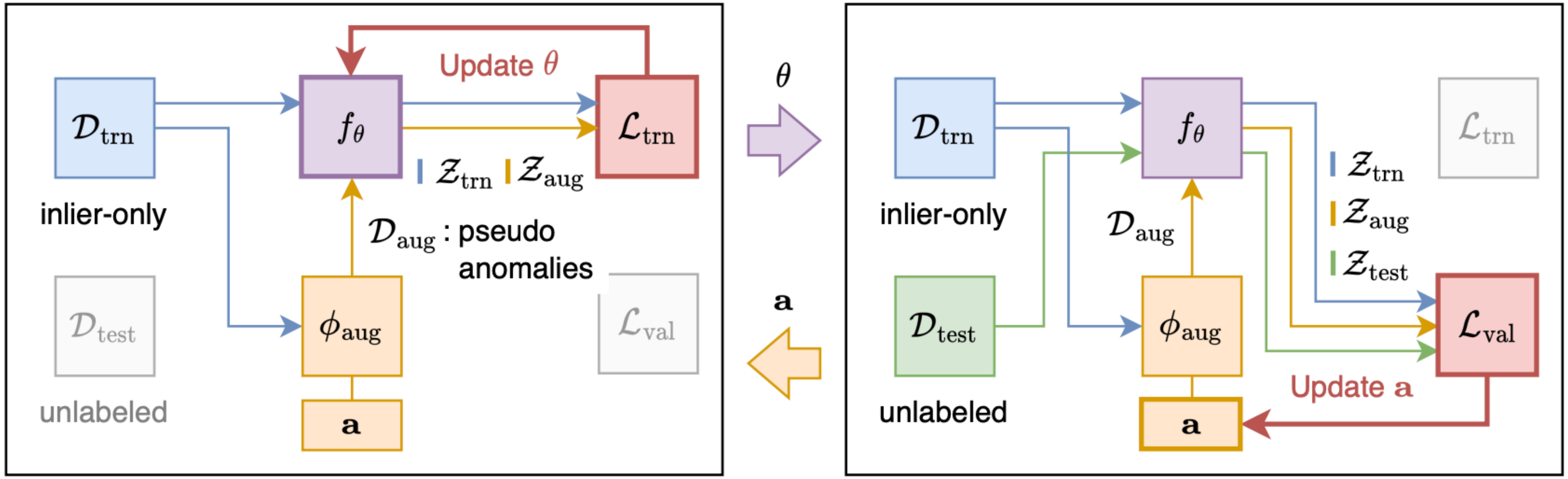
Differentiable Augmentation

- We propose CutDiff, a new differentiable variant of CutOut.



Overview

- Our framework aims to solve bilevel optimization for θ and $\mathbf{a}$.
  - θ is the set of parameters for a detector f_θ.
  - $\mathbf{a}$ is the set of (hyper)parameters for f_{aug}.



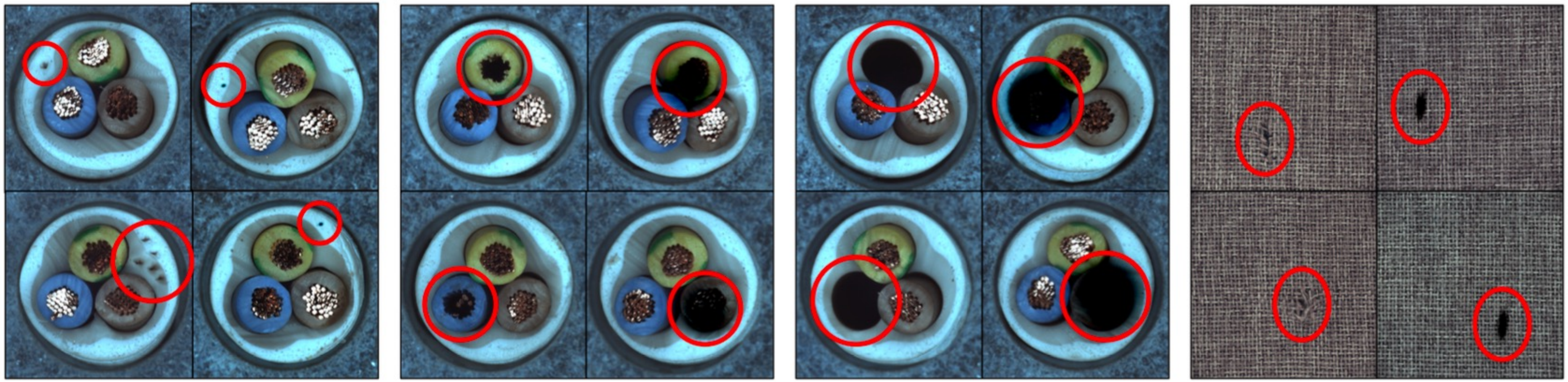
Quantitative Experiments

- ST-SSAD generally outperforms the baselines, but not in all cases.

		Main Result									Ablation Study		
Object	Anomaly Type	AE	D-SVDD	RS-CO	RD-CO	RS-CP	RD-CP	RS-CD	RD-CD	**ST-SSAD**	MMD1	MMD2	FO
Cable	Bent wire	0.515	0.432	0.556	0.560	0.703	**0.756**	0.527	0.580	0.490	0.581	0.643	0.579
Cable	Cable swap	0.639	0.295	0.483	0.625	0.618	0.683	0.574	**0.696**	0.532	0.510	0.562	0.545
Cable	Combined	0.584	0.587	0.879	0.857	0.880	**0.949**	0.901	0.879	0.925	0.939	0.962	0.882
Cable	Cut inner insulation	0.758	0.591	0.630	0.737	0.766	**0.833**	0.623	0.732	0.667	0.633	0.649	0.689
Cable	Cut outer insulation	**0.989**	0.343	0.695	0.815	0.787	0.871	0.703	0.790	0.516	0.428	0.461	0.527
Cable	Missing cable	0.920	0.466	0.953	0.961	0.755	0.801	0.935	0.945	**0.998**	0.855	0.772	0.999
Cable	Missing wire	0.433	0.494	0.781	0.655	0.501	0.546	0.708	0.620	**0.863**	0.547	0.477	0.699
Cable	Poke insulation	0.287	0.471	0.469	0.527	0.645	**0.672**	0.489	0.503	0.630	0.692	0.816	0.676
Carpet	Color	0.578	0.716	0.669	0.508	0.412	0.287	0.643	0.639	**0.938**	0.761	0.741	0.918
Carpet	Cut	0.198	0.758	0.439	0.608	0.403	0.411	0.490	0.767	**0.790**	0.353	0.401	0.595
Carpet	Hole	0.626	0.676	0.379	0.613	0.404	0.389	0.470	**0.765**	0.590	0.438	0.229	0.630
Carpet	Metal contamination	0.056	**0.739**	0.198	0.304	0.240	0.167	0.255	0.474	0.076	0.392	0.134	0.392
Carpet	Thread	0.394	**0.742**	0.494	0.585	0.469	0.517	0.508	0.679	0.483	0.492	0.541	0.642
Grid	Bent	**0.849**	0.168	0.456	0.322	0.421	0.433	0.337	0.354	0.771	0.780	0.650	0.602
Grid	Broken	0.806	0.183	0.397	0.312	0.487	0.502	0.340	0.392	**0.869**	0.845	0.887	0.884
Grid	Glue	0.704	0.143	0.634	0.568	0.674	0.732	0.681	0.578	**0.906**	0.966	0.974	0.721
Grid	Metal contamination	0.851	0.229	0.421	0.380	0.499	0.514	0.425	0.613	**0.858**	0.861	0.665	0.732
Grid	Thread	0.583	0.209	0.612	0.494	0.500	0.549	0.654	0.611	**0.973**	0.962	0.969	0.964
Tile	Crack	0.770	0.728	0.872	0.993	0.743	0.636	0.837	**0.999**	0.749	0.740	0.820	0.595
Tile	Glue strip	0.697	0.509	0.693	**0.836**	0.665	0.700	0.675	0.831	0.767	0.585	0.649	0.561
Tile	Gray stroke	0.637	0.785	0.845	0.642	0.583	0.657	0.856	0.802	**0.974**	0.653	0.706	0.973
Tile	Oil	0.414	0.690	0.708	0.745	0.464	0.576	0.683	**0.837**	0.554	0.548	0.614	0.555
Tile	Rough	0.724	0.387	0.606	**0.725**	0.631	0.661	0.568	0.657	0.690	0.700	0.549	0.605
	p-value	.0000	.0000	.0000	.0012	.0000	.0000	.0000	.0728	**Ours**	.0268	.0073	.1332

Qualitative Experiments

- ST-SSAD learns the patch size and ratio in an end-to-end way:



(a) Anomaly (poke_insulation) | Aug. (scale=0.001) | (b) Anomaly (missing_wire) | Aug. (scale=0.081) | (c) Anomaly (missing_cable) | Aug. (scale=0.177) | (d) Anomaly (cut) | Aug. (scale=0.047)

Outline

1. Introduction
2. Preliminaries: Benchmark Study
3. Approach 1: Offline Selection
4. Approach 2: End-to-end Learning
5. **Conclusion**

Conclusion

- Hyperparameter tuning is an essential problem in SSAD.
- However, the problem is largely underexplored.
  - I hope more people to get interested in the topic and participate.
- We have proposed an **offline** and an **end-to-end** method.

Prof. Leman Akoglu (CMU)

Lingxiao Zhao (CMU)

Prof. Yue Zhao (USC)

References

- [1] J. Yoo, T. Zhao, and L. Akoglu. “Data Augmentation is a Hyperparameter: Cherry-picked Self-Supervision for Unsupervised Anomaly Detection is Creating the Illusion of Success.” **Transactions on Machine Learning Research (2023)**
- [2] J. Yoo, Y. Zhao, L. Zhao, and L. Akoglu. “DSV: An Alignment Validation Loss for Self-supervised Outlier Model Selection.” **ECML PKDD 2023**
- [3] J. Yoo, L. Zhao, and L. Akoglu. “End-to-End Augmentation Hyperparameter Tuning for Self-Supervised Anomaly Detection.” **arXiv (2023)**
- [4] L. Akoglu and J. Yoo. “Self-Supervision for Tackling Unsupervised Anomaly Detection: Pitfalls and Opportunities.” **BigData 2023**

Appendix

Jaemin Yoo

School of Electrical Engineering

KAIST

Self-supervised Learning in General

- **Self-supervised learning (SSL)** is a general technique
  1. **Pre-training:**
     - Given a large set of unlabeled data
     - Create pseudo labels for training a model in a supervised way
  2. **Fine-tuning:**
     - Update the model for a downstream task with a few labels
- **Example:** Large language models (GPT, BERT, etc.)

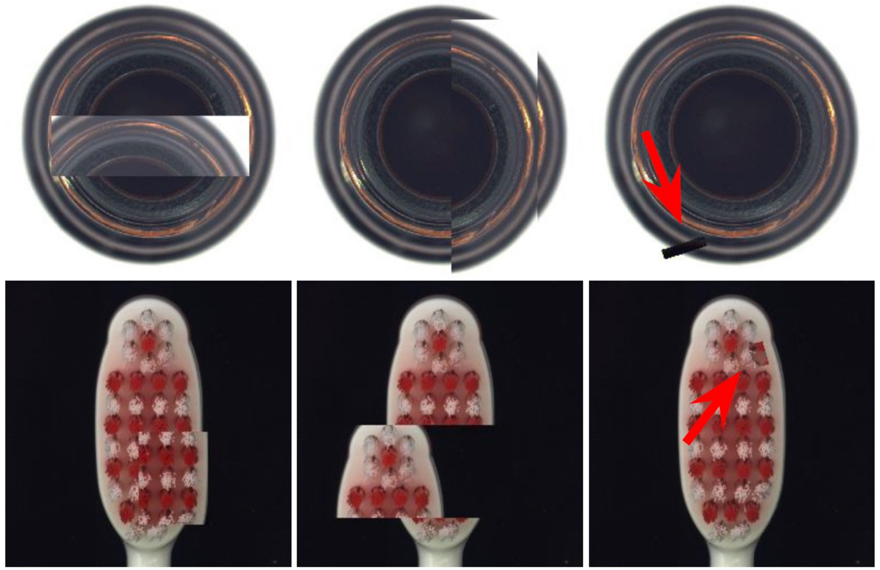
I have eaten an apple and a banana.

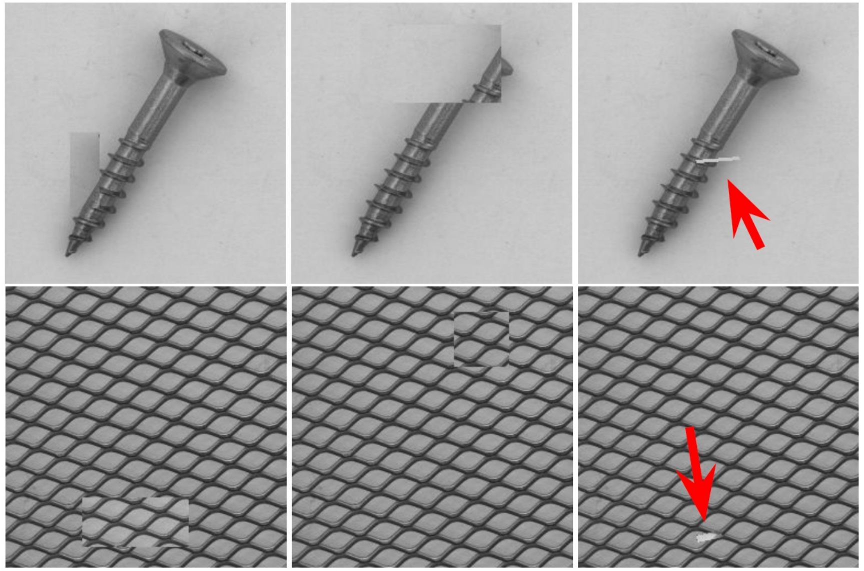
SSAD vs. SSL in Supervised Learning

- SSL is generally used **with fine-tuning**
  - SSL may not be perfectly aligned with the downstream task
- **SSL on AD** is used **without fine-tuning**
  - SSL task solely determines the performance of AD
  - SSL task should be **aligned well** with the downstream task
- **Implication:** The choice of f_{aug} is very important SSAD

SSAD: Example

- **CutPaste** (Li et al., 2021) is an example of f_{aug}
  - Cuts a random patch from an image and pastes into a different location
  - Generated images look like (local) defects in industrial object images





Li et al. "CutPaste: Self-Supervised Learning for Anomaly Detection and Localization." CVPR 2021