Deep Anomaly Detection Using Geometric Transformations

Izhak Golan, Ran El-Yaniv Technion – Israel Institute of Technology {izikgo, rani}@cs.technion.ac.il

Neural Information Processing Systems Foundation

Abstract

- Future machine learning applications (e.g., self-driving cars ,domestic robots), will inevitably encounter risks including **statistical uncertainties**.
- The well-known problem of **anomaly detection** is an example of such risk, and its resolution is of the utmost importance to mission critical machine learning applications.
- We present a novel and simple technique that **dramatically improves** all known anomaly detection methods for images.

Problem Statement

We consider the problem of anomaly detection in images:

- Let $\mathcal{X}$ be the space of all "natural" images, and let $X \subseteq \mathcal{X}$ be the set of images defined as *normal*. Images that are not in X are referred to as *anomalies* or *novelties*.
- Given a sample $S \subseteq X$, we would like to learn a scoring (ranking) function $n_S(x) : \mathcal{X} \to \mathbb{R}$, such that higher scores indicate that samples are more likely to be in X.
- Performance is measured using the area under the Receiver Operating Characteristic (ROC) curve.

Approach

- Our method is based on the observation that learning to discriminate between many types of geometric transformations applied to normal images, encourages learning of features that are useful for detecting novelties.
- Unlike most published work on anomaly detection in high-dimensional data, our method does not incorporate any kind of reconstruction of samples.

Algorithm

Algorithm 1 Deep Anomaly Detection Using Geometric Transformations

Input: S: a set of "normal" images. $\mathcal{T} = \{T_0, T_1, \ldots, T_{k-1}\}$: a set of geometric transformations. f_θ: a softmax classifier parametrized by θ.
Output: A normality scoring function $n_S(x)$.

procedure GetNormalityScore($S, \mathcal{T}, f_\theta$)
  $S_\mathcal{T} \leftarrow \{(T_j(x), j) : x \in S, T_j \in \mathcal{T}\}$
  while not converged **do**
    Train f_θ on the labeled set $S_\mathcal{T}$
  end while
  $n \leftarrow |S|$
  for $i \in \{0, \ldots, k-1\}$ **do**
    $S_i \leftarrow \{\mathbf{y}(T_i(x)) | x \in S\}$
    $\bar{\boldsymbol{s}} \leftarrow \frac{1}{n} \sum_{\boldsymbol{s} \in S_i} \boldsymbol{s}$
    $\bar{\boldsymbol{l}} \leftarrow \frac{1}{n} \sum_{\boldsymbol{s} \in S_i} \log \boldsymbol{s}$
    $\tilde{\boldsymbol{\alpha}}_i \leftarrow \bar{\boldsymbol{s}} \frac{(k-1)(-\Psi(1))}{\bar{\boldsymbol{s}} \cdot \log \bar{\boldsymbol{s}} - \bar{\boldsymbol{s}} \cdot \bar{\boldsymbol{l}}}$ ▷ Initialization from [3]
    while not converged **do**
      $\tilde{\boldsymbol{\alpha}}_i \leftarrow \Psi^{-1}\left(\Psi\left(\sum_j [\boldsymbol{\alpha}_i]_j\right) + \bar{\boldsymbol{l}}\right)$ ▷ Fixed point method from [5]
    end while
  end for
  return $n_S(x) \triangleq \sum_{i=0}^{k-1} (\tilde{\boldsymbol{\alpha}}_i - 1) \cdot \log \mathbf{y}(T_i(x))$
end procedure

* The function $\Psi(\cdot)$ is the Digamma function, and its inverse is calculated numerically.

References

[1] L. Deecke, R. Vandermeulen, L. Ruff, S. Mandt, and M. Kloft. Anomaly detection with generative adversarial networks, 2018. URL https://openreview.net/forum?id=S1EfylZ0Z.

[2] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1VGkIxRZ.

[3] T. Minka. Estimating a dirichlet distribution, 2000.

[4] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. Vandermeulen, A. Binder, E. Müller, and M. Kloft. Deep one-class classification. In *International Conference on Machine Learning*, pages 4390–4399, 2018.

[5] N. Wicker, J. Muller, R. K. R. Kalathur, and O. Poch. A maximum likelihood approximation method for dirichlet's parameter estimation. *Computational statistics & data analysis*, 52(3):1315–1322, 2008.

[6] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang. Deep structured energy based models for anomaly detection. In *Proceedings of the 33rd International Conference on Machine Learning - Volume 48*, ICML'16, pages 1100–1109. JMLR.org, 2016.

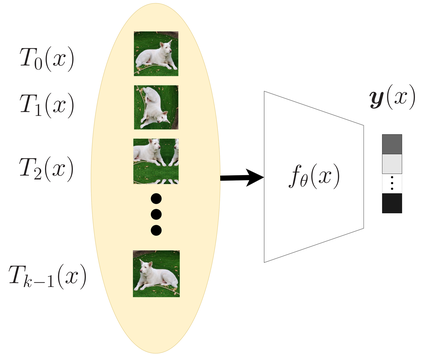
[7] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018.

Method – Training

Step I: Train a classifier on a self-labeled dataset

- Let $\mathcal{T} = \{T_0, T_1, \ldots, T_{k-1}\}$ be a set of geometric transformations, where for each $1 \le i \le k-1, T_i : \mathcal{X} \to \mathcal{X}$, and $T_0(x) = x$ is the identity transformation.
- Train a multi-class deep classifier (f_θ) with a softmax output on the self-labeled set $S_\mathcal{T}$ defined as:

$$S_\mathcal{T} \triangleq \{(T_j(x), j) : x \in S, T_j \in \mathcal{T}\}.$$



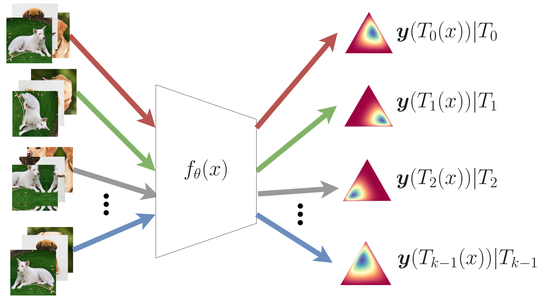
Step II: Dirichlet Normality Score

- We approximate each conditional distribution to be $\mathbf{y}(T_i(x))|T_i \sim \mathrm{Dir}(\boldsymbol{\alpha}_i)$, where $\boldsymbol{\alpha}_i \in \mathbb{R}_+^k, x \sim p_X(x)$, $i \sim \mathrm{Uni}(0, k-1)$, and $p_X(x)$ is the (unknown) real data probability distribution of "normal" samples.
- Calculate MLE parameters for the Dirichlet distributions (denoted $\tilde{\boldsymbol{\alpha}}_i$).
- Normality score of an image x is the combined log-likelihood under all conditional distributions:

$$n_S(x) = \sum_{i=0}^{k-1} \left[\log \Gamma\Big(\sum_{j=0}^{k-1} [\tilde{\boldsymbol{\alpha}}_i]_j\Big) - \sum_{j=0}^{k-1} \log \Gamma([\tilde{\boldsymbol{\alpha}}_i]_j) + \sum_{j=0}^{k-1} ([\tilde{\boldsymbol{\alpha}}_i]_j - 1) \log \mathbf{y}(T_i(x))_j \right].$$

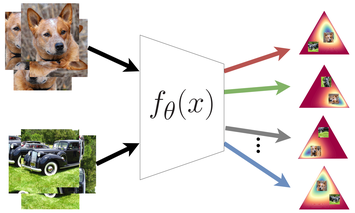
- All $\tilde{\boldsymbol{\alpha}}_i$ are constant w.r.t x, so we use an equivalent, simplified, score:

$$n_S(x) = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} ([\tilde{\boldsymbol{\alpha}}_i]_j - 1) \log \mathbf{y}(T_i(x))_j = \sum_{i=0}^{k-1} (\tilde{\boldsymbol{\alpha}}_i - 1) \cdot \log \mathbf{y}(T_i(x)). \quad (1)$$



Method – Inference

- When testing an unseen image, pass each of its $|\mathcal{T}|$ transformed versions through the classifier f_θ.
- Using estimated $\tilde{\boldsymbol{\alpha}}_i$ parameters, compute the normality score for each image by Equation 1.



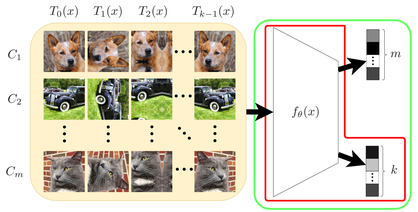
Experimental Results

Average area under the ROC curve in % with SEM (over 5 runs) of anomaly detection methods. For all datasets, each model was trained on the single class, and tested against all other classes. OC-SVM hyperparameters in RAW and CAE variants were optimized with hindsight knowledge.

Dataset	c_i	OC-SVM RAW	OC-SVM CAE	OC-SVM E2E [4]	DAGMM [7]	DSEBM [6]	AD-GAN [1]	OURS
CIFAR-10 (32x32x3)	0	70.6	**74.9**	61.7±1.3	41.4±2.3	56.0±6.9	64.9	74.7±0.4
	1	51.3	51.7	65.9±0.7	57.1±2.0	48.3±1.8	39.0	**95.7±0.0**
	2	69.1	68.9	50.8±0.3	53.8±4.0	61.9±0.1	65.2	**78.1±0.4**
	3	52.4	52.8	59.1±0.4	51.2±0.8	50.1±0.4	48.1	**72.4±0.5**
	4	77.3	76.7	60.9±0.3	52.2±7.3	73.3±0.2	73.5	**87.8±0.2**
	5	51.2	52.9	65.7±0.8	49.3±3.6	60.5±0.3	47.6	**87.8±0.1**
	6	74.1	70.9	67.7±0.8	64.9±1.7	68.4±0.3	62.3	**83.4±0.5**
	7	52.6	53.1	67.3±0.3	55.3±0.8	53.3±0.7	48.7	**95.5±0.1**
	8	70.9	71.0	75.9±0.4	51.9±2.4	73.9±0.3	66.0	**93.3±0.0**
	9	50.6	50.6	73.1±0.4	54.2±5.8	63.6±3.1	37.8	**91.3±0.1**
	Average	62.0	62.4	64.8	53.1	60.9	55.3	**86.0**
CIFAR-100 (32x32x3)	0	68.0	68.4	-	43.4±3.9	64.0±0.2	63.1	**74.7±0.4**
	1	63.1	63.6	-	49.5±2.7	47.9±0.1	54.9	**68.5±0.2**
	2	50.4	52.0	-	66.1±1.7	53.7±4.1	41.3	**74.0±0.5**
	3	62.7	64.7	-	52.6±1.0	48.4±0.5	50.0	**81.0±0.8**
	4	59.7	58.2	-	56.9±3.0	59.7±6.3	40.6	**78.4±0.5**
	5	53.5	54.9	-	52.4±2.2	46.6±1.6	42.8	**59.1±1.0**
	6	55.9	57.2	-	55.0±1.1	51.7±0.8	51.1	**81.8±0.2**
	7	64.4	62.9	-	52.8±3.7	54.8±1.6	55.4	**65.0±0.1**
	8	66.7	65.6	-	53.2±4.8	66.7±0.2	59.2	**85.5±0.4**
	9	70.1	74.1	-	42.5±2.5	71.2±1.2	62.7	**90.6±0.1**
	10	83.0	84.1	-	52.7±3.9	78.3±1.1	79.8	**87.6±0.2**
	11	59.7	58.0	-	46.4±2.4	62.7±0.7	53.7	**83.9±0.6**
	12	68.7	68.5	-	42.7±3.1	66.8±0.0	58.9	**83.2±0.3**
	13	**65.0**	64.6	-	45.4±0.7	52.6±0.1	57.4	58.0±0.4
	14	50.7	51.2	-	57.2±1.3	44.0±0.6	39.4	**92.1±0.2**
	15	63.5	62.8	-	48.8±1.5	56.8±0.1	55.6	**68.3±0.1**
	16	68.3	66.6	-	54.4±3.1	63.1±0.1	63.3	**73.5±0.2**
	17	71.7	73.7	-	36.4±2.3	73.0±1.0	66.7	**93.8±0.1**
	18	50.2	52.8	-	52.4±1.4	57.7±1.6	44.3	**90.7±0.1**
	19	57.5	58.4	-	50.3±1.0	55.5±0.7	53.0	**85.0±0.2**
	Average	62.6	63.1	-	50.5	58.8	54.7	**78.7**
Fashion-MNIST (32x32x1)	0	98.2	97.7	-	42.1±9.1	91.6±1.2	89.9	**99.4±0.0**
	1	90.3	89.9	-	55.1±3.5	71.8±0.5	81.9	**97.6±0.1**
	2	90.7	**91.4**	-	50.4±7.3	88.3±0.2	87.6	91.1±0.2
	3	**94.2**	90.7	-	57.0±6.7	87.3±3.6	91.2	89.9±0.4
	4	89.4	89.1	-	26.9±5.4	85.2±0.9	86.5	**92.1±0.0**
	5	91.8	88.5	-	70.5±9.7	87.1±0.0	89.6	**93.4±0.9**
	6	**83.4**	81.7	-	48.3±5.0	73.4±4.1	74.3	83.3±0.1
	7	98.8	98.7	-	83.5±11.4	98.1±0.0	97.2	**98.9±0.1**
	8	**91.9**	90.6	-	49.9±7.2	86.0±3.2	89.0	90.8±0.1
	9	99.0	98.6	-	34.0±3.0	97.1±0.3	97.1	**99.2±0.0**
	Average	92.8	91.7	-	51.8	86.6	88.4	**93.5**
CatsVsDogs (64x64x3)	0	50.4	55.2	-	43.4±0.5	47.1±1.7	50.7	**88.3±0.3**
	1	53.0	49.9	-	52.0±1.9	56.1±1.2	48.1	**89.2±0.3**
	Average	51.7	52.5	-	47.7	51.6	49.4	**88.8**

Identifying Out-of-distribution Samples in Labeled datasets

- Although not the main focus of this work, our method can easily be extended to multi-class datasets.
- We created a two-headed classification model with two separate softmax output layers. One for categories (e.g., dog, car, cat, etc.), and another for classifying transformations (our method).
- During training, both softmax outputs are trained. At test time, we only utilize the transformations softmax layer output.



- When training on the CIFAR-10 dataset, and taking the tiny-imagenet dataset to be anomalies as done in [2] (ODIN method), we improved ODIN's results on all three reported metrics: AUROC, AUPR-In, and AUPR-Out.

Method	AUROC	AUPR-In	AUPR-Out
ODIN	92.1	89.0	93.6
Ours	**95.7**	**96.1**	**95.4**

- In contrast to our method, many multi-class out-of-distribution detection approaches are *inapplicable* in the pure single class setting, where there are no class labels.

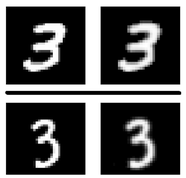
Intuition Experiments

- The effectiveness of the chosen transformations set is affected by the classifier's ability to predict which transformation was applied on a given transformed image.
- In addition, the harder it is for the trained classifier to correctly predict the transformations that were applied on anomalies, the better our approach performs.
- Examples using MNIST digits:

Normal Class	Anomaly Class	$\mathcal{T}$	AUROC
8	3	(identity, horizontal flip)	64.6
3	8	(identity, horizontal flip)	95.7
8	3	(identity, translation)	91.9

- A model trained on a normal sample captures important features of the normal class. Example - a model trained for normal images ('3') is used to "optimize" input images ('0' and '3'):

Left: original '0's. **Right**: the '0's optimized to a normality score learned on '3's.



Left: original '3's. **Right**: the '3's optimized to a normality score learned on '3's.

Future Research

- Develop a theory that grounds the use of geometric transformations.
- Can we explicitly optimize the choice of the set of transformations, perhaps using a small sample of anomalies?
- Can we improve multi-class classification performance using the proposed method?