i-Tap
Outlier Detection

홍익대학교
노승문
2020. 12. 12.

Introduction

- Goal
  - Outlier/Anomaly detection
  - Robust algorithm

- Problem Setting
  - Definitions of outliers

- Strategies
  - Various scoring rules

Outlier Detection and Robust PCA Using a Convex Measure of Innovation

Mostafa Rahmani and Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th St. Bellevue, WA 98004, USA
{mostafarahmani,liping11}@baidu.com

NIPS 2019

Abstract

This paper presents a provable and strong algorithm, termed Innovation Search (iSearch), to robust Principal Component Analysis (PCA) and outlier detection. An outlier by definition is a data point which does not participate in forming a low dimensional structure with a large number of data points in the data. In other words, an outlier carries some innovation with respect to most of the other data points. iSearch ranks the data points based on their values of innovation. A convex optimization problem is proposed whose optimal value is used as our measure of innovation. We derive analytical performance guarantees for the proposed robust PCA method under different models for the distribution of the outliers including randomly distributed outliers, clustered outliers, and linearly dependent outliers. Moreover, it is shown that iSearch provably recovers the span of the inliers when the inliers lie in a union of subspaces. In the challenging scenarios in which the outliers are close to each other or they are close to the span of the inliers, iSearch is shown to outperform most of the existing methods.

Robust PCA: two data corruption models

the inliers lie in a low dimensional subspace. In the literature of robust PCA, two main models for the data corruption are considered: the element-wise model and the column-wise model. These two models are corresponding to two different robust PCA problems. In the element-wise model, it is assumed that a small subset of the elements of the data matrix are corrupted and the support of the corrupted elements is random. This problem is known as the low rank plus sparse matrix decomposition problem [1,3,4,23,24]. In the column-wise model, a subset of the columns of the data are affected by the data corruption [5,7,8,11,17,20,25,26,36–39]. Section 2 provides a review of the robust (to column-wise corruption) PCA methods. This paper focuses on the column-wise model, i.e., we assume that the given data follows Data Model 1.

Data Model 1

Data Model 1. *The data matrix* $\mathbf{D} \in \mathbb{R}^{M_1 \times M_2}$ *can be expressed as* $\mathbf{D} = [\mathbf{B} \;\; (\mathbf{A} + \mathbf{N})]\,\mathbf{T}$ *, where* $\mathbf{A} \in \mathbb{R}^{m \times n_i}$, $\mathbf{B} \in \mathbb{R}^{m \times n_o}$, $\mathbf{T}$ *is an arbitrary permutation matrix, and* $[\mathbf{B} \;\; (\mathbf{A} + \mathbf{N})]$ *represents the concatenation of* $\mathbf{B}$ *and* $(\mathbf{A} + \mathbf{N})$*. The columns of* $\mathbf{A}$ *lie in an* r*-dimensional subspace* $\mathcal{U}$*. The columns of* $\mathbf{B}$ *do not lie entirely in* $\mathcal{U}$*, i.e., the* n_i *columns of* $\mathbf{A}$ *are the inliers and the* n_o *columns of* $\mathbf{B}$ *are the outliers. The matrix* $\mathbf{N}$ *represents additive noise. The orthonormal matrix* $\mathbf{U} \in \mathbb{R}^{M_1 \times r}$ *is a basis for* $\mathcal{U}$*. Evidently,* $M_2 = n_i + n_o$.

Summary

Summary of Contributions: The main contributions can be summarized as follows.

- The proposed approach introduces a new idea to the robust PCA problem. iSearch uses a convex optimization problem to measure the Innovation of the data points. It is shown that iSearch mostly outperforms the exiting methods in handling close outliers and noisy data.
- To the best of our knowledge, the proposed approach and the CoP method presented in [27] are the only robust PCA methods which are supported with analytical performance guarantees under different models for the distributions of the outliers including the randomly distributed outliers, the clustered outliers, and the linearly dependent outliers.
- In addition to considering different models for the distribution of the outliers, we provide analytical performance guarantees under different models for the distributions of the inliers too. The presumed models include the union of subspaces and the uniformly at random distribution on $\mathcal{U} \cap \mathbb{S}^{M_1-1}$ where $\mathbb{S}^{M_1-1}$ denotes the unit ℓ_2-norm sphere in $\mathbb{R}^{M_1}$.

Comparison 1

Connection and Contrast to Coherence Pursuit: In [27], Coherence Pursuit (CoP) was proposed as a provable robust PCA method. CoP computes the Coherence Values for all the data points to rank the data points. The Coherence value corresponding to data column $\mathbf{d}$ is a measure of resemblance between $\mathbf{d}$ and the rest of the data columns. CoP uses the inner product between $\mathbf{d}$ and the rest of the data points to measure the resemblance between $\mathbf{d}$ and the rest of data. In sharp contrast, iSearch finds an optimal direction corresponding to each data column. The optimal direction corresponding to data column $\mathbf{d}$ is used to measure the innovation of $\mathbf{d}$ with respect to the rest of the data columns. We show through theoretical studies and numerical experiments that finding the optimal directions makes iSearch significantly stronger than CoP in detecting outliers which carry weak innovation.

Comparison 2

Connection and Contrast to Innovation Pursuit: In [28, 29], Innovation Pursuit was proposed as a new subspace clustering method. The optimization problem proposed in [28] finds a direction in the span of the data such that it is orthogonal to the maximum number of data points. We present a new discovery about the applications of Innovation Pursuit. It is shown that the idea of innovation search can be used to design a strong outlier detection algorithm. iSearch uses an optimization problem similar to the linear optimization problem used in [28] to measure the innovation of the data points.

Innovation Value

Suppose $\mathbf{d}$ is a column of $\mathbf{D}$, define $\mathbf{c}^*$ as the optimal point of

$$\min_{\mathbf{c}} \|\mathbf{c}^T\mathbf{D}\|_1 \quad \text{subject to} \quad \mathbf{c}^T\mathbf{d} = 1\,, \tag{2}$$

and define the Innovation Value corresponding to $\mathbf{d}$ as $1/\|\mathbf{D}^T\mathbf{c}^*\|_1$. The main idea of iSearch is that $\mathbf{c}^*$ has two completely different behaviours with respect to $\mathcal{U}$ (when $\mathbf{d}$ is an outlier and when $\mathbf{d}$ is an inlier). Suppose $\mathbf{d}$ is an outlier. The optimization problem (2) searches for a direction whose projection on $\mathbf{d}$ is non-zero and it has the minimum projection on the rest of the data points. As $\mathbf{d}$ is an outlier, $\mathbf{d}$ has a non-zero projection on $\mathcal{U}^\perp$. In addition, as n_i is large, (2) searches for a direction in the ambient whose projection on $\mathcal{U}$ is as weak as possible. Thus, $\mathbf{c}^*$ lies in $\mathcal{U}^\perp$ or it is close to $\mathcal{U}^\perp$.

Data reduction normalization

Data dimension

Reduced dimension

# of data

Algorithm 1 Subspace Recovery Using iSearch

1. Data Preprocessing. The input is data matrix $\mathbf{D} \in \mathbb{R}^{M_1 \times M_2}$.
1.1 Define $\mathbf{Q} \in \mathbb{R}^{M_1 \times r_d}$ as the matrix of first r_d left singular vectors of $\mathbf{D}$ where r_d is the number of non-zero singular values. Set $\mathbf{D} = \mathbf{Q}^T\mathbf{D}$. If dimensionality reduction is not required, skip this step.
1.2 Normalize the ℓ_2-norm of the columns of $\mathbf{D}$, i.e., set $\mathbf{d}_i$ equal to $\mathbf{d}_i/\|\mathbf{d}_i\|_2$ for all $1 \leq i \leq M_2$.

2. Direction Search. Define $\mathbf{C}^* \in \mathbb{R}^{r_d \times M_2}$ such that $\mathbf{c}_i^* \in \mathbb{R}^{r_d \times 1}$ is the optimal point of

$$\min_{\mathbf{c}} \|\mathbf{c}^T\mathbf{D}\|_1 \quad \text{subject to} \quad \mathbf{c}^T\mathbf{d}_i = 1$$

or define $\mathbf{C}^* \in \mathbb{R}^{r_d \times M_2}$ as the optimal point of

$$\min_{\mathbf{C}} \|(\mathbf{C}^T\mathbf{D})^T\|_1 \quad \text{subject to} \quad \text{diag}(\mathbf{C}^T\mathbf{D}) = \mathbf{1}\,. \tag{1}$$

Score of i-th data point

3. Computing the Innovation Values. Define vector $\mathbf{x} \in \mathbb{R}^{M_2 \times 1}$ such that $\mathbf{x}(i) = 1/\|\mathbf{D}^T\mathbf{c}_i^*\|_1$.

4. Building Basis. Construct matrix $\mathbf{Y}$ from the columns of $\mathbf{D}$ corresponding to the smallest elements of $\mathbf{x}$ such that they span an r-dimensional subspace.

Output: The column-space of $\mathbf{Y}$ is the identified subspace.

Assumption

Assumption 1. *The columns of* $\mathbf{A}$ *are drawn uniformly at random from* $\mathcal{U} \cap \mathbb{S}^{M_1-1}$. *The columns of* $\mathbf{B}$ *are drawn uniformly at random from* $\mathbb{S}^{M_1-1}$. *To simplify the exposition and notation, it is assumed without loss of generality that* $\mathbf{T}$ *in Data Model 1 is the identity matrix, i.e,* $\mathbf{D} = [\mathbf{B} \ \ \mathbf{A}]$.

We use a synthetic numerical example to explain the idea behind the proposed approach. Suppose $\mathbf{D} \in \mathbb{R}^{20\times 250}$, $n_i = 200$, $n_o = 50$, and $r = 3$. Assume that $\mathbf{D}$ follows Assumption 1.

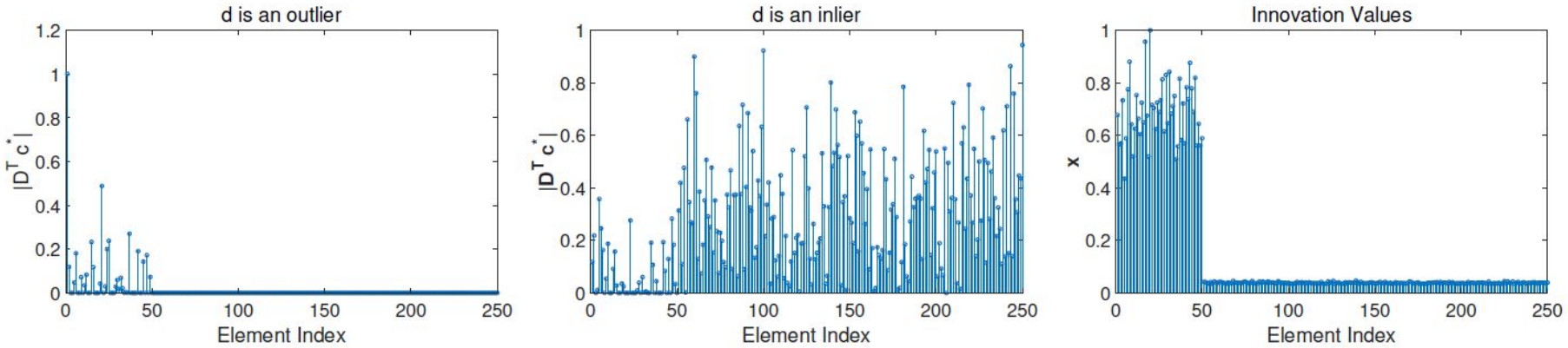


Figure 1: The first 50 columns are outliers. The left panel shows vector $\mathbf{D}^T\mathbf{c}^*$ when $\mathbf{d}$ is an outlier. The middle panel depicts $\mathbf{D}^T\mathbf{c}^*$ when $\mathbf{d}$ is an inlier. The right panel shows the Innovation Values corresponding to all the data points (vector $\mathbf{x}$ was defined in Algorithm 1).

3.2 Building the Basis Matrix

The data points corresponding to the least Innovation Values are used to construct the basis matrix $\mathbf{Y}$. If the data follows Assumption 1, the r data points corresponding to the r smallest Innovation Values span $\mathcal{U}$ with overwhelming probability [35]. In practise, the algorithm should continue adding new columns to $\mathbf{Y}$ until the columns of $\mathbf{Y}$ spans an r-dimensional subspace. This approach requires to check the singular values of $\mathbf{Y}$ several times. We propose two techniques to avoid this extra steps. The first approach is based on the side information that we mostly have about the data. In many applications, we can have an upper-bound on n_o because outliers are mostly associated with rare events. If we know that the number of outliers is less than y percent of the data, matrix $\mathbf{Y}$ can be constructed using $(1-y)$ percent of the data columns which are corresponding to the least Innovation Values. The second approach is the adaptive column sampling method proposed in [27]. The adaptive column sampling method avoids sampling redundant columns.

Want to show

Minimum innovation values among all outliers

$$\max\left(\{1/\|\mathbf{D}^T\mathbf{c}_i^*\|_1\}_{i=n_o+1}^{M_2}\right) < \min\left(\{1/\|\mathbf{D}^T\mathbf{c}_j^*\|_1\}_{j=1}^{n_o}\right). \quad (3)$$

Maximum innovation values among all inliers

Definition 1. *Define* $\mathbf{c}_j^* = \underset{\mathbf{d}_j^T\mathbf{c}=1}{\arg\min} \|\mathbf{c}^T\mathbf{D}\|_1$. *In addition, define* $\chi = \max\left(\{\|\mathbf{c}_j^*\|_2\}_{i=1}^{n_o}\right)$, *and* $n_z^{'} = \max\left(\{|\mathcal{I}_0^i|\}_{i=1}^{n_o}\right)$ *where* $\mathcal{I}_0^i = \{i \in [n_o] : \mathbf{c}_i^{*T}\mathbf{b}_i = 0\}$ *and* $\mathbf{b}_i$ *is the* i^{th} *column of* $\mathbf{B}$. *The value* $|\mathcal{I}_0^i|$ *is the number of outliers which are orthogonal to* $\mathbf{c}_i^*$.

Case 1: unstructured outliers

Theorem 1. *Suppose* $\mathbf{D}$ *follows Assumption 1 and define* $\mathcal{A} = \sqrt{\frac{1}{2\pi}\frac{n_i}{\sqrt{r}}} - \sqrt{n_i} - \sqrt{\frac{n_i \log\frac{1}{\delta}}{2r-2}}$. *If*

$$\mathcal{A} > \Bigg[\frac{n_o}{M_1} + 2\sqrt{\frac{n_o}{M_1}} + \sqrt{\frac{2n_o \log 1/\delta}{(M_1-1)M_1}} + \sqrt{\frac{n_o c_\delta'' \log n_o/\delta}{M_1^2}} + n_z' \sqrt{\frac{c_\delta''}{M_1^2}} + \sqrt{\left(\frac{n_o}{M_1^2} + \frac{\eta_\delta}{M_1}\right)\log n_o/\delta}\Bigg]\sqrt{\frac{4M_1 c_\delta}{M_1 - c_\delta r}} \quad \textit{and} \tag{4}$$

$$\mathcal{A} > \max\left(\chi\frac{n_o}{\sqrt{M_1}} + 2\sqrt{n_o}(1+\sqrt{\chi}) + 2\sqrt{\frac{2\chi n_o \log\frac{1}{\delta}}{M-1}}, 2n_z'\sqrt{\frac{c_\delta r}{M_1}} + 2\sqrt{\frac{n_o c_\delta r \log n_o/\delta}{M_1}}\right),$$

then (3) holds and $\mathcal{U}$ *is recovered exactly with probability at least* $1 - 7\delta$ *where* $\sqrt{c_\delta} = 3\max\left(1, \sqrt{\frac{8M_1\pi}{(M_1-1)r}}, \sqrt{\frac{8M_1 \log n_0/\delta}{(M_1-1)r}}\right)$, $\sqrt{c_\delta''} = 3\max\left(1, \sqrt{\frac{8M_1\pi}{M_1-1}}, \sqrt{\frac{16M_1 \log n_0/\delta}{M_1-1}}\right)$, *and* $\eta_\delta = \max\left(\frac{4}{3}\log\frac{2M_1}{\delta}, \sqrt{4\frac{n_o}{M_1}\log\frac{2M_1}{\delta}}\right)$.

Case 2: clustered outliers

- Structured outliers can form a low dimensional structure

- Important rare event
  - Web attack
  - Malignant tissues

Assumption 2. *A column of* $\mathbf{B}$ *is formed as* $\mathbf{b}_i = \frac{1}{\sqrt{1+\eta^2}}(\mathbf{q} + \eta\mathbf{v}_i)$. *The unit* ℓ_2*-norm vector* $\mathbf{q}$ *does not lie in* $\mathcal{U}$, $\{\mathbf{v}_i\}_{i=1}^{n_o}$ *are drawn uniformly at random from* $\mathbb{S}^{M_1-1}$, *and* η *is a positive number.*

Case 2: clustered outliers

Theorem 2. *Suppose the distribution of the inliers/outliers follows Assumption-1/Assumption-2. Assume that $\mathcal{Q}$ is equal to the column-space of $[\mathbf{U}\ \ \mathbf{q}]$. Define $\mathbf{q}^{\perp} = \frac{(\mathbf{I}-\mathbf{U}\mathbf{U}^T)\mathbf{q}}{\|(\mathbf{I}-\mathbf{U}\mathbf{U}^T)\mathbf{q}\|_2}$, define $\beta = \max\left(\{1/|\mathbf{d}_i^T\mathbf{q}^{\perp}| \ : \mathbf{d}_i \in \mathbf{B}\}\right)$, define $\mathbf{c}_i^*$ as the optimal point of (5) with $\mathbf{d} = \mathbf{d}_i$, and assume that $\eta < |\mathbf{q}^T\mathbf{q}^{\perp}|$. In addition, define $\mathcal{A} = \frac{\sqrt{1+\eta^2}}{2\beta}\left(\sqrt{\frac{2}{\pi}}\frac{n_i}{\sqrt{r}} - 2\sqrt{n_i} - \sqrt{\frac{2n_i \log\frac{1}{\delta}}{r-1}}\right)$. If*

outliers

$$\mathcal{A} > n_o\|\mathbf{U}^T\mathbf{q}\|_2 + \eta\sqrt{\frac{n_o r c_\delta \log n_o/\delta}{M_1}}\,, \tag{6}$$

$$\mathcal{A} > n_o|\mathbf{q}^T\mathbf{q}^{\perp}| + n_o\eta\sqrt{\frac{c''_\delta \log n_o/\delta}{M_1}}\,,$$

then (3) holds and $\mathcal{U}$ is recovered exactly with probability at least $1-5\delta$.

Case 3: linearly dependent outliers

Assumption 3. *Define subspace $\mathcal{U}_o$ with dimension r_o such that $\mathcal{U}_o \nsubseteq \mathcal{U}$ and $\mathcal{U} \nsubseteq \mathcal{U}_o$. The outliers are randomly distributed on $\mathbb{S}^{M_1-1} \cap \mathcal{U}_o$. The orthonormal matrix $\mathbf{U}_o \in \mathbb{R}^{M_1 \times r_o}$ is a basis for $\mathcal{U}_o$.*

Theorem 3. *Suppose the distribution of the inliers/outliers follows Assumption-1/Assumption-3. Define* $\mathcal{A} = \sqrt{\frac{2}{\pi}}\frac{n_i}{\sqrt{r}} - 2\sqrt{n_i} - \sqrt{\frac{2n_i \log\frac{1}{\delta}}{r-1}}$. *If*

$$\mathcal{A} > 2n_z^{'}\|\mathbf{U}^T\mathbf{U}_o\| + 2\|\mathbf{U}^T\mathbf{U}_o\|\sqrt{n_o \log n_o/\delta}\,,$$

$$\mathcal{A} > \frac{2\|\mathbf{U}^T\mathbf{U}_o\|}{\xi}\left(\frac{n_o}{\sqrt{r_o}} + 2\sqrt{n_o} + \sqrt{\frac{2n_o\log\frac{1}{\delta}}{r_o-1}} + 2\sqrt{\left(\frac{n_o}{r_o}+\eta_\delta^{'}\right)\log\frac{n_o}{\delta}} + n_z^{'}\right), \tag{7}$$

$$\mathcal{A} > \left(\frac{\chi n_o}{\sqrt{r_o}} + 2\sqrt{\chi n_o} + \sqrt{\chi\frac{2n_o\log\frac{1}{\delta}}{r_o-1}}\right)\|\mathbf{U}_o^T\mathbf{U}^\perp\|\,,$$

then (3) holds and $\mathcal{U}$ is recovered exactly with probability at least $1-5\delta$ where $\eta_\delta^{'} = \max\left(\frac{4}{3}\log 2(r_o)/\delta\,,\ \sqrt{4\frac{n_o}{r_o}\log\frac{2r_d}{\delta}}\right)$ *and* $\xi = \frac{\min\left(\left\{\|\mathbf{b}_j^T\mathbf{U}^\perp\|_2\right\}_{j=1}^{n_o}\right)}{\|\mathbf{U}_o^T\mathbf{U}^\perp\|}$.

The data follows Assumption 1 with $r = 4$ and $M_1 = 100$. The left plot of Figure 2 shows the phase transition of iSearch versus n_i/r and n_o/M_1. White indicates correct subspace recovery and black designates incorrect recovery. Theorem 1 indicated that if n_i/r is sufficiently large, iSearch yields exact recovery even if n_o is larger than n_i. This experiment confirms the theoretical result. According to Figure 2, even when $n_o = 3000$, 40 inliers are enough to guarantee exact subspace recovery.

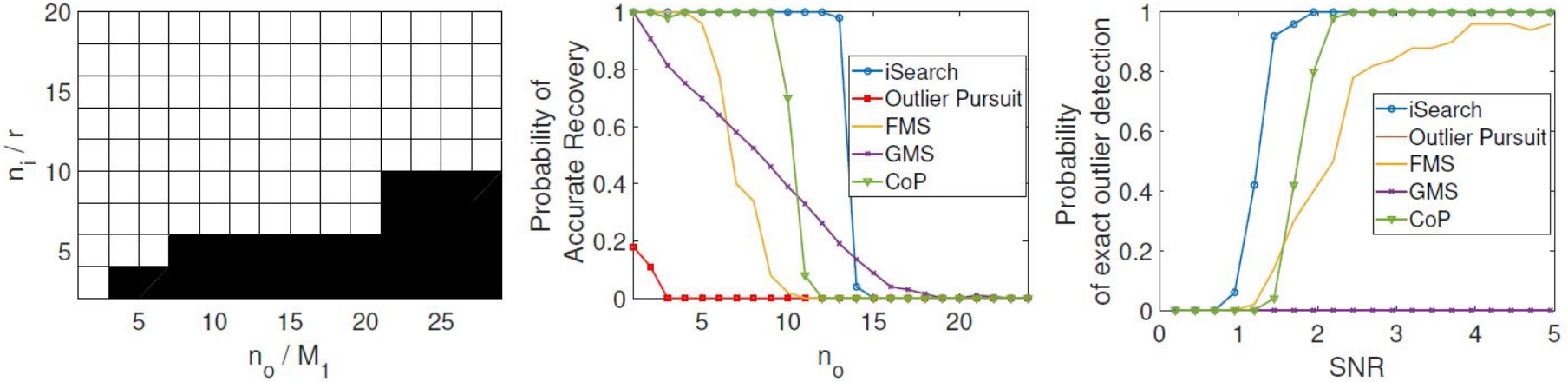


Figure 2: **Left panel:** The phase transition of iSearch in presence of the unstructured outliers versus n_i/r and n_o/M_1 ($M_1 = 100$ and $r = 4$). **Middle panel:** The probability of accurate subspace recovery versus the number of structured outliers ($n_i = 100$, $\eta = 0.1$, $M_1 = 100$, and $r = 10$). **Right panel:** The probability of exact outlier detection versus SNR. The data contains 10 structured outliers and 300 unstructured outliers ($n_i = 100$, $n_o = 310$, $r = 5$, and $M_1 = 100$).

Figure 3: Some of the frames of the Waving Tree video file. The highlighted frames are detected as outliers by R1-PCA.

Questions?

Likelihood Ratios for Out-of-Distribution Detection

Jie Ren[*,†]
Google Research
jjren@google.com

Peter J. Liu [‡]
Google Research
peterjliu@google.com

Emily Fertig[†]
Google Research
emilyaf@google.com

Jasper Snoek
Google Research
jsnoek@google.com

Ryan Poplin
Google Research
rpoplin@google.com

Mark A. DePristo
Google Research
mdepristo@google.com

Joshua V. Dillon [‡]
Google Research
jvdillon@google.com

Balaji Lakshminarayanan[*,‡]
DeepMind
balajiln@google.com

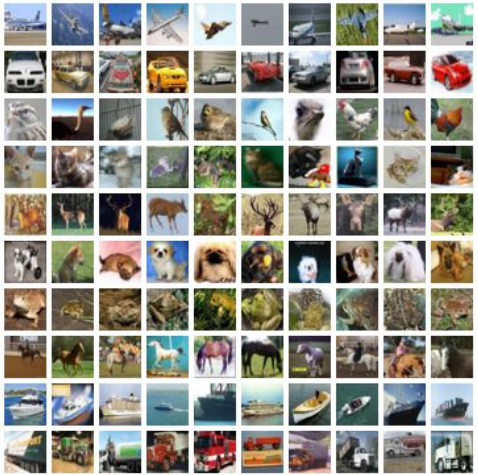
NIPS 2019

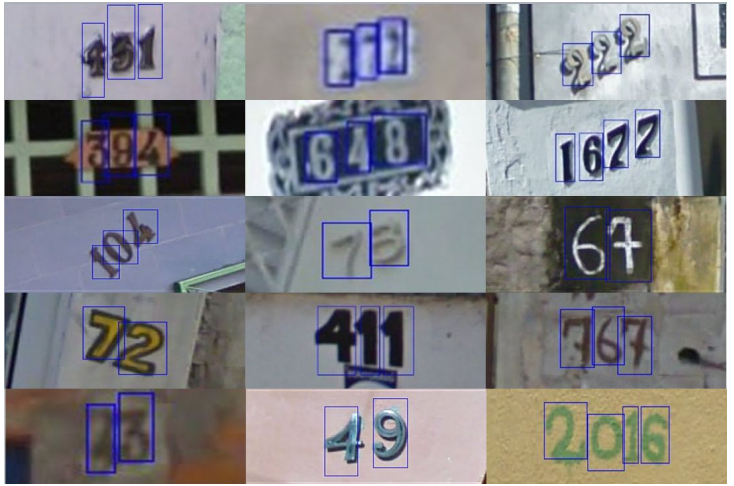
Abstract

Discriminative neural networks offer little or no performance guarantees when deployed on data not generated by the same process as the training distribution. On such out-of-distribution (OOD) inputs, the prediction may not only be erroneous, but confidently so, limiting the safe deployment of classifiers in real-world applications. One such challenging application is bacteria identification based on genomic sequences, which holds the promise of early detection of diseases, but requires a model that can output low confidence predictions on OOD genomic sequences from new bacteria that were not present in the training data. We introduce a genomics dataset for OOD detection that allows other researchers to benchmark progress on this important problem. We investigate deep generative model based approaches for OOD detection and observe that the likelihood score is heavily affected by population level background statistics. We propose a likelihood ratio method for deep generative models which effectively corrects for these confounding background statistics. We benchmark the OOD detection performance of the proposed method against existing approaches on the genomics dataset and show that our method achieves state-of-the-art performance. We demonstrate the generality of the proposed method by showing that it significantly improves OOD detection when applied to deep generative models of images.

Unrealistic Dataset

AI safety (Amodei et al., 2016). The majority of recent work on OOD detection for neural networks is evaluated on image datasets where the neural network is trained on one benchmark dataset (e.g. CIFAR-10) and tested on another (e.g. SVHN). While these benchmarks are important, there is a need for more realistic datasets which reflect the challenges of dealing with OOD inputs in practical applications.





Failure Mode

A popular and intuitive strategy for detecting OOD inputs is to train a generative model (or a hybrid model cf. Nalisnick et al. (2019)) on training data and use that to detect OOD inputs at test time (Bishop, 1994). However, Nalisnick et al. (2018) and Choi et al. (2018) recently showed that deep generative models trained on image datasets can assign higher likelihood to OOD inputs. We report a similar failure mode for likelihood based OOD detection using deep generative models of genomic sequences. We investigate this phenomenon and find that the likelihood can be confounded by general population level background statistics. We propose a likelihood ratio method which uses a background model to correct for the background statistics and enhances the in-distribution specific features for OOD detection. While our investigation was motivated by the genomics problem, we found our methodology to be more general and it shows positive results on image datasets as well. In

Summary

- We create a realistic benchmark for OOD detection, that is motivated by challenges faced in applying deep learning models on genomics data. The sequential nature of genetic sequences provides a new modality and hopefully encourages the OOD research community to contribute to "machine learning that matters" (Wagstaff, 2012).
- We show that likelihood from deep generative models can be confounded by background statistics.
- We propose a likelihood ratio method for OOD detection, which significantly outperforms the raw likelihood on OOD detection for deep generative models on image datasets.
- We evaluate existing OOD methods on the proposed genomics benchmark and demonstrate that our method achieves state-of-the-art (SOTA) performance on this challenging problem.

Setting

Suppose we have an in-distribution dataset $\mathcal{D}$ of $(\boldsymbol{x}, y)$ pairs sampled from the distribution $p^*(\boldsymbol{x}, y)$ where $\boldsymbol{x}$ is the extracted feature vector or raw input and $y \in \mathcal{Y} := \{1, \ldots, k, \ldots, K\}$ is the label assigning membership to one of K in-distribution classes. For simplicity, we assume inputs to be discrete, i.e. $x_d \in \{A, C, G, T\}$ for genomic sequences and $x_d \in \{0, \ldots, 255\}$ for images. In general, OOD inputs are samples $(\boldsymbol{x}, y)$ generated from an underlying distribution other than $p^*(\boldsymbol{x}, y)$. In this paper, we consider an input $(\boldsymbol{x}, y)$ to be OOD if $y \notin \mathcal{Y}$: that is, the class y does not belong to one of the K in-distribution classes. Our goal is to accurately detect if an input $\boldsymbol{x}$ is OOD or not.

Failure Mode 1

to the input data, and then evaluate the likelihood of new inputs under that model. However, recent work has highlighted significant issues with this approach for OOD detection on images, showing that deep generative models such as Glow (Kingma & Dhariwal, 2018) and PixelCNN (Oord et al., 2016; Salimans et al., 2017) sometimes assign higher likelihoods to OOD than in-distribution inputs. For example, Nalisnick et al. (2018) and Choi et al. (2018) show that Glow models trained on the CIFAR-10 image dataset assign higher likelihood to OOD inputs from the SVHN dataset than they do to in-distribution CIFAR-10 inputs; Nalisnick et al. (2018), Shafaei et al. (2018) and Hendrycks et al. (2018) show failure modes of PixelCNN and PixelCNN++ for OOD detection.

Failure Mode 2

Failure of density estimation for OOD detection We investigate whether density estimation-based methods work well for OOD detection in genomics. As a motivating observation, we train a deep generative model, more precisely LSTM (Hochreiter & Schmidhuber, 1997), on in-distribution genomic sequences (composed by {A, C, G, T}), and plot the log-likelihoods of both in-distribution and OOD inputs (See Section 5.2 for the dataset and the full experimental details). Figure 1a shows that the histogram of the log-likelihood for OOD sequences largely overlaps with that of in-distribution sequences with AUROC of 0.626, making it unsuitable for OOD detection. Our observations show a failure mode of deep generative models for OOD detection on genomic sequences and are complementary to earlier work which showed similar results for deep generative models on images (Nalisnick et al., 2018; Choi et al., 2018).

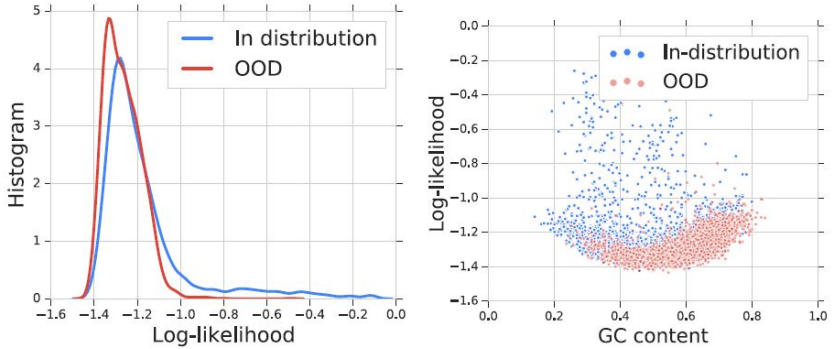


Figure 1: (a) Log-likelihood hardly separates in-distribution and OOD inputs with AUROC of 0.626. (b) The log-likelihood is heavily affected by the GC-content of a sequence.

Failure Mode 2 - cont.

When investigating this failure mode, we discovered that the log-likelihood under the model is heavily affected by a sequence's *GC-content*, see Figure 1b. GC-content is defined as the percentage of bases that are either G or C, and is used widely in genomic studies as a basic statistic for describing overall genomic composition (Sueoka, 1962), and studies have shown that bacteria have an astonishing diversity of genomic GC-content, from 16.5% to 75% (Hildebrand et al., 2010). Bacteria from similar groups tend to have similar GC-content at the population level, but they also have characteristic biological patterns that can distinguish them well from each other. The confounding effect of GC-content in Figure 1b makes the likelihood less reliable as a score for OOD detection, because an OOD input may result in a higher likelihood than an in-distribution input, because it has high GC-content (cf. the bottom right part of Figure 1b) and not necessarily because it contains characteristic patterns specific to the in-distribution bacterial classes.

Main Idea

High level idea Assume that an input $\boldsymbol{x}$ is composed of two components, (1) a *background* component characterized by population level background statistics, and (2) a *semantic* component characterized by patterns specific to the in-distribution data. For example, images can be modeled as backgrounds plus objects; text can be considered as a combination of high frequency stop words plus semantic words (Luhn, 1960); genomes can be modeled as background sequences plus motifs (Bailey & Elkan, 1995; Reinert et al., 2009). More formally, for a D-dimensional input $\boldsymbol{x} = x_1, \ldots, x_D$, we assume that there exists an unobserved variable $\boldsymbol{z} = z_1, \ldots, z_D$, where $z_d \in \{B, S\}$ indicates if the dth dimension of the input x_d is generated from the *B*ackground model or the *S*emantic model.

Grouping the semantic and background parts, the input can be factored as $\boldsymbol{x} = \{\boldsymbol{x}_B, \boldsymbol{x}_S\}$ where $\boldsymbol{x}_B = \{x_d \mid z_d = B, d = 1, \ldots, D\}$. For simplicity, assume that the background and semantic components are generated independently. The likelihood can be then decomposed as follows,

$$p(\boldsymbol{x}) = p(\boldsymbol{x}_B)p(\boldsymbol{x}_S). \tag{1}$$

Log-likelihood Ratio

Assume that $p_{\boldsymbol{\theta}}(\cdot)$ is a model trained using in-distribution data, and $p_{\boldsymbol{\theta}_0}(\cdot)$ is a background model that captures general background statistics. We propose a likelihood ratio statistic that is defined as

$$\mathsf{LLR}(\boldsymbol{x}) = \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x})}{p_{\boldsymbol{\theta}_0}(\boldsymbol{x})} = \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_B)\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_S)}{p_{\boldsymbol{\theta}_0}(\boldsymbol{x}_B)\, p_{\boldsymbol{\theta}_0}(\boldsymbol{x}_S)}, \tag{2}$$

where we use the factorization from Equation 1. Assume that (i) both models capture the background information equally well, that is $p_{\boldsymbol{\theta}}(\boldsymbol{x}_B) \approx p_{\boldsymbol{\theta}_0}(\boldsymbol{x}_B)$ and (ii) $p_{\boldsymbol{\theta}}(\boldsymbol{x}_S)$ is more peaky than $p_{\boldsymbol{\theta}_0}(\boldsymbol{x}_S)$ as the former is trained on data containing semantic information, while the latter model $\boldsymbol{\theta}_0$ is trained using data with noise perturbations. Then, the likelihood ratio can be approximated as

$$\mathsf{LLR}(\boldsymbol{x}) \approx \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_S) - \log p_{\boldsymbol{\theta}_0}(\boldsymbol{x}_S). \tag{3}$$

Learn the background

background but different semantic component). In practice, we only observe $\boldsymbol{x}$, and it is not always easy to split an input into background and semantic parts $\{\boldsymbol{x}_B, \boldsymbol{x}_S\}$. As a practical alternative, we propose training a background model by perturbing inputs. Adding the right amount of perturbations to inputs can corrupt the semantic structure in the data, and hence the model trained on perturbed inputs captures only the population level background statistics.

Training the Background Model In practice, we add perturbations to the input data by randomly selecting positions in $x_1 \ldots x_D$ following an independent and identical Bernoulli distribution with rate μ and substituting the original character with one of the other characters with equal probability. The procedure is inspired by genetic mutations. See Algorithm 1 in Appendix A for the pseudocode

LLR for AR

Likelihood ratio for auto-regressive models Auto-regressive models are one of the popular choices for generating images (Oord et al., 2016; Van den Oord et al., 2016; Salimans et al., 2017) and sequence data such as genomics (Zou et al., 2018; Killoran et al., 2017) and drug molecules (Olivecrona et al., 2017; Gupta et al., 2018), and text (Jozefowicz et al., 2016). In auto-regressive models, the log-likelihood of an input can be expressed as $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{d=1}^{D} \log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d})$, where $\boldsymbol{x}_{<d} = x_1 \dots x_{d-1}$. Decomposing the log-likelihood into background and semantic parts, we have

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{d:x_d \in \boldsymbol{x}_B} \log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d}) + \sum_{d:x_d \in \boldsymbol{x}_S} \log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d}). \tag{4}$$

We can use a similar auto-regressive decomposition for the background model $p_{\boldsymbol{\theta}_0}(\boldsymbol{x})$ as well. Assuming that both the models capture the background information equally well, $\sum_{d:x_d \in \boldsymbol{x}_B} \log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d}) \approx \sum_{d:x_d \in \boldsymbol{x}_B} \log p_{\boldsymbol{\theta}_0}(x_d|\boldsymbol{x}_{<d})$, the likelihood ratio is approximated as

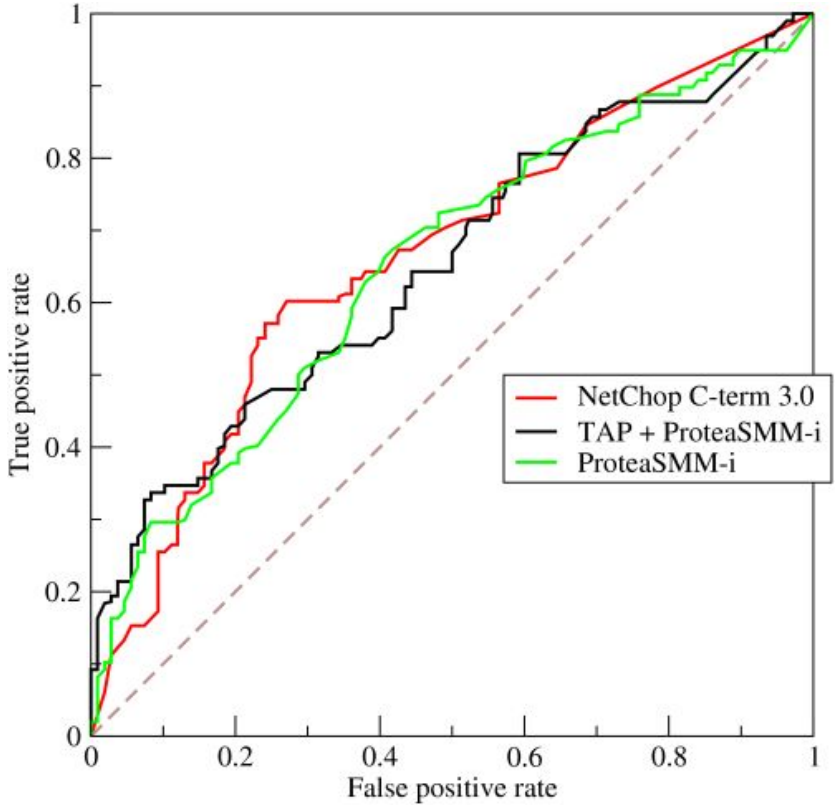
$$\mathsf{LLR}(\boldsymbol{x}) \approx \sum_{d:x_d \in \boldsymbol{x}_S} \log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d}) - \sum_{d:x_d \in \boldsymbol{x}_S} \log p_{\boldsymbol{\theta}_0}(x_d|\boldsymbol{x}_{<d}) = \sum_{d:x_d \in \boldsymbol{x}_S} \log \frac{p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d})}{p_{\boldsymbol{\theta}_0}(x_d|\boldsymbol{x}_{<d})}. \tag{5}$$

Comparison

Baseline methods for comparison We compare our approach to several existing methods.

1. The maximum class probability, $p(\hat{y}|\boldsymbol{x}) = \max_k p(y = k|\boldsymbol{x})$. OOD inputs tend to have lower scores than in-distribution data (Hendrycks & Gimpel, 2016).
2. The entropy of the predicted class distribution, $-\sum_k p(y = k|\boldsymbol{x}) \log p(y = k|\boldsymbol{x})$. High entropy of the predicted class distribution, and therefore a high predictive uncertainty, which suggests that the input may be OOD.
3. The ODIN method proposed by Liang et al. (2017). ODIN uses temperature scaling (Guo et al., 2017), adds small perturbations to the input, and applies a threshold to the resulting predicted class to distinguish in- and out-of- distribution inputs. This method was designed for continuous inputs and cannot be directly applied to discrete genomic sequences. We propose instead to add perturbations to the input of the last layer that is closest to the output of the neural network.
4. The Mahalanobis distance of the input to the nearest class-conditional Gaussian distribution estimated from the in-distribution data. Lee et al. (2018) fit class-conditional Gaussian distributions to the activations from the last layer of the neural network.
5. The classifier-based ensemble method that uses the average of the predictions from multiple independently trained models with random initialization of network parameters and random shuffling of training inputs (Lakshminarayanan et al., 2017).
6. The log-odds of a binary classifier trained to distinguish between in-distribution inputs from all classes as one class and randomly perturbed in-distribution inputs as the other.
7. The maximum class probability over K in-distribution classes of a $(K + 1)$-class classifier where the additional class is perturbed in-distribution.
8. The maximum class probability of a K-class classifier for in-distribution classes but the predicted class distribution is explicitly trained to output uniform distribution on perturbed in-distribution inputs. This is similar to using simulated OOD inputs from GAN (Lee et al., 2017) or using auxiliary datasets of outliers (Hendrycks et al., 2018) for calibration purpose.
9. The generative model-based ensemble method that measures $\mathbb{E}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})] - \mathrm{Var}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]$ of multiple likelihood estimations from independently trained model with random initialization and random shuffling of the inputs. (Choi et al., 2018).

AUROC: Area under Receiver Operating Characteristic Curve



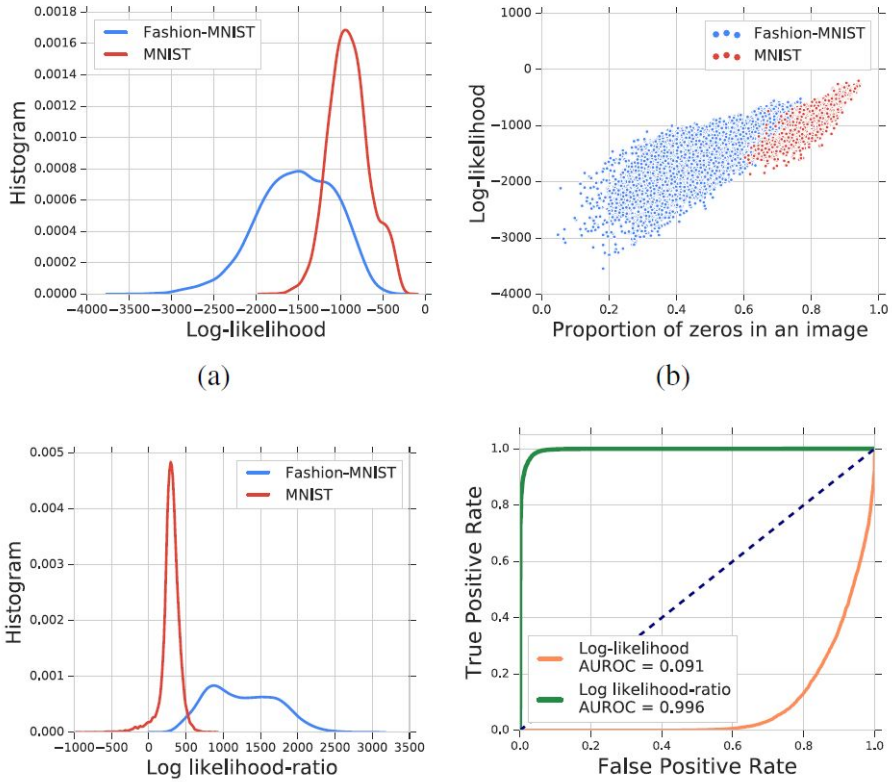
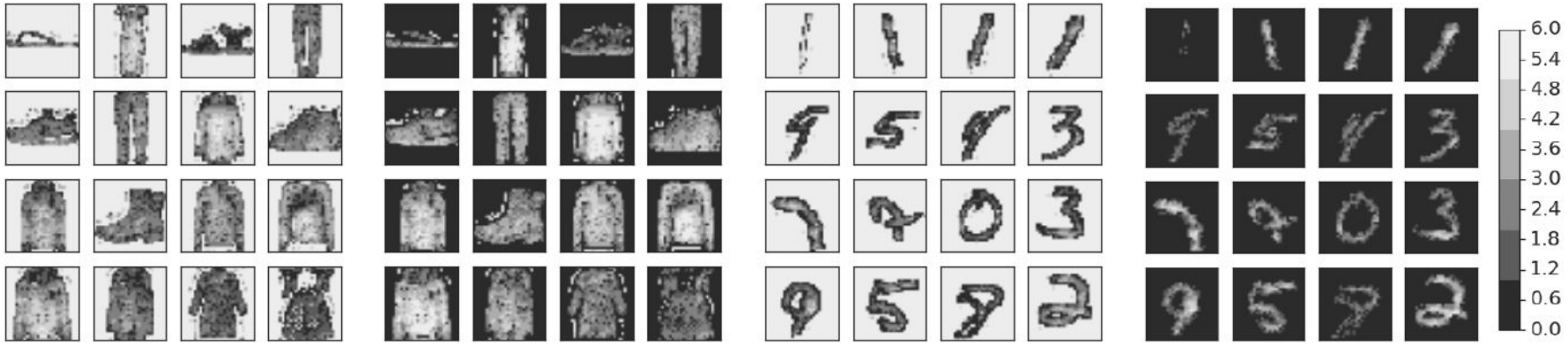


Figure 2: (a) Log-likelihood of MNIST images (OOD) is higher than that of Fashion-MNIST images (in-distribution). (b) Log-likelihood is highly correlated with the background (proportion of zeros in an image). (c) Log-likelihood ratio is higher for Fashion-MNIST (in-dist) than MNIST (OOD). (d) Likelihood ratio significantly improves the AUROC of OOD detection from 0.091 to 0.996.



(a) Likelihood (b) Likelihood-Ratio (c) Likelihood (d) Likelihood-Ratio

Figure 3: The log-likelihood of each pixel in an image $\log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d})$, and the log likelihood-ratio of each pixel $\log p_{\boldsymbol{\theta}}(x_d|\boldsymbol{x}_{<d}) - \log p_{\boldsymbol{\theta}_0}(x_d|\boldsymbol{x}_{<d}), d = 1 \ldots, 784.$, for 16 Fashion-MNIST images (a, b) and MNIST images (c, d). Lighter gray color indicates larger value (see colorbar). Note that the range of log-likelihood (negative value) is different from that of log likelihood-ratio (mostly positive value). For the ease of visualization, we unify the colorbar by adding a constant to the log-likelihood score. The images are randomly sampled from the test dataset and sorted by their likelihood $p_\theta(\mathbf{x})$. Looking at which pixels contribute the most to each quantity, we observe that the likelihood value is dominated by the "background" pixels on both Fashion-MNIST and MNIST, whereas likelihood ratio focuses on the "semantic" pixels.

5.2 OOD detection for genomic sequences

Dataset for detecting OOD genomic sequences We design a new dataset for evaluating OOD methods. As bacterial classes are discovered gradually over time, in- and out-of-distribution data can be naturally separated by the time of discovery. Classes discovered before a cutoff time can be regarded as in-distribution classes, and those discovered afterward, which were unidentified at the cutoff time, can be regarded as OOD. We choose two cutoff years, 2011 and 2016, to define the training, validation, and test splits (Figure 4). Our dataset contains of 10 in-distribution classes, 60 OOD classes for validation, and 60 OOD classes for testing. Note that the validation OOD dataset is only used for hyperparameter tuning, and the validation OOD classes are disjoint from the test OOD

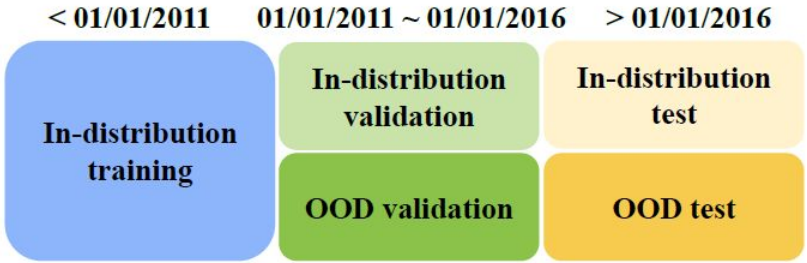


Table 1: AUROC↑, AUPRC↑, and FPR80↓ for detecting OOD inputs using likelihood and likelihood-ratio method and other baselines on (a) Fashion-MNIST vs. MNIST datasets and (b) genomic dataset. The up and down arrows on the metric names indicate whether greater or smaller is better. μ in the parentheses indicates the background model is tuned only using noise perturbed input, and (μ and λ) indicates the background model is tuned by both perturbation and L_2 regularization. Numbers in front and inside of the brackets are mean and standard error respectively based on 10 independent runs with random initialization of network parameters and random shuffling of training inputs. For ensemble models, the mean and standard error are estimated based on 10 bootstrap samples from 30 independent runs, which can be underestimations of the true standard errors.

	AUROC↑	AUPRC↑	FPR80↓
Likelihood	0.089 (0.002)	0.320 (0.000)	1.000 (0.001)
Likelihood Ratio (ours, μ)	0.973 (0.031)	0.951 (0.063)	0.005 (0.008)
Likelihood Ratio (ours, μ, λ)	**0.994 (0.001)**	**0.993 (0.002)**	**0.001 (0.000)**
$p(\hat{y}\|\boldsymbol{x})$	0.734 (0.028)	0.702 (0.026)	0.506 (0.046)
Entropy of $p(y\|\boldsymbol{x})$	0.746 (0.027)	0.726 (0.026)	0.448 (0.049)
ODIN	0.752 (0.069)	0.763 (0.062)	0.432 (0.116)
Mahalanobis distance	0.942 (0.017)	0.928 (0.021)	0.088 (0.028)
Ensemble, 5 classifiers	0.839 (0.010)	0.833 (0.009)	0.275 (0.019)
Ensemble, 10 classifiers	0.851 (0.007)	0.844 (0.006)	0.241 (0.014)
Ensemble, 20 classifiers	0.857 (0.005)	0.849 (0.004)	0.240 (0.011)
Binary classifier	0.455 (0.105)	0.505 (0.064)	0.886 (0.126)
$p(\hat{y}\|\boldsymbol{x})$ with noise class	0.877 (0.050)	0.871 (0.054)	0.195 (0.101)
$p(\hat{y}\|\boldsymbol{x})$ with calibrations	0.904 (0.023)	0.895 (0.023)	0.139 (0.044)
WAIC, 5 models	0.221 (0.013)	0.401 (0.008)	0.911 (0.008)

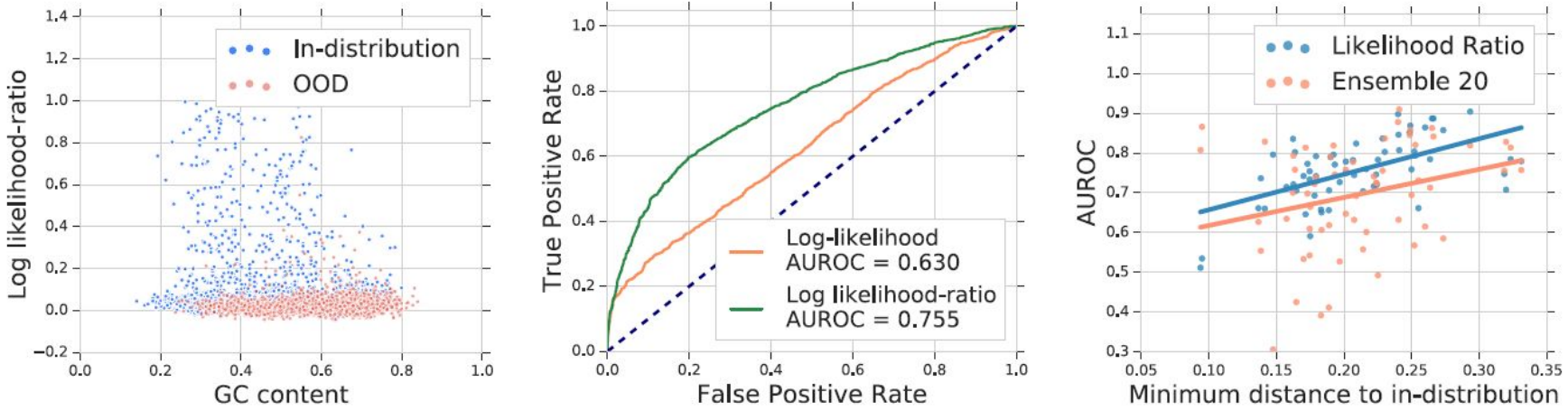
(a)

	AUROC↑	AUPRC↑	FPR80↓
Likelihood	0.626 (0.001)	0.613 (0.001)	0.661 (0.002)
Likelihood Ratio (ours, μ)	0.732 (0.015)	0.685 (0.017)	0.534 (0.031)
Likelihood Ratio (ours, μ, λ)	**0.755 (0.005)**	**0.719 (0.006)**	**0.474 (0.011)**
$p(\hat{y}\|\boldsymbol{x})$	0.634 (0.003)	0.599 (0.003)	0.669 (0.007)
Entropy of $p(y\|\boldsymbol{x})$	0.634 (0.003)	0.599 (0.003)	0.617 (0.007)
Adjusted ODIN	0.697 (0.010)	0.671 (0.012)	0.550 (0.021)
Mahalanobis distance	0.525 (0.010)	0.503 (0.007)	0.747 (0.014)
Ensemble, 5 classifiers	0.682 (0.002)	0.647 (0.002)	0.589 (0.004)
Ensemble, 10 classifiers	0.690 (0.001)	0.655 (0.002)	0.574 (0.004)
Ensemble, 20 classifiers	0.695 (0.001)	0.659 (0.001)	0.570 (0.004)
Binary classifier	0.635 (0.016)	0.634 (0.015)	0.619 (0.025)
$p(\hat{y}\|\boldsymbol{x})$ with noise class	0.652 (0.004)	0.627 (0.005)	0.643 (0.008)
$p(\hat{y}\|\boldsymbol{x})$ with calibration	0.669 (0.005)	0.635 (0.004)	0.627 (0.006)
WAIC, 5 models	0.628 (0.001)	0.616 (0.001)	0.657 (0.002)

(b)

Table 2: CIFAR-10 vs SVHN results: AUROC↑, AUPRC↑, FPR80↓ for detecting OOD inputs using likelihood and our likelihood-ratio method.

	AUROC↑	AUPRC↑	FPR80↓
Likelihood	0.095 (0.003)	0.320 (0.001)	1.000 (0.000)
Likelihood Ratio (ours, μ)	0.931 (0.032)	0.888 (0.049)	0.062 (0.073)
Likelihood Ratio (ours, μ, λ)	0.930 (0.042)	0.881 (0.064)	0.066 (0.123)



(a) (b) (c)

Figure 5: (a) The likelihood-ratio score is roughly independent of the GC-content which makes it less susceptible to background statistics and better suited for OOD detection. (b) ROCs and AUROCs for OOD detection using likelihood and likelihood-ratio. (c) Correlation between the AUROC of OOD detection and distance to in-distribution classes using Likelihood Ratio and the Ensemble method.

Questions?

Effective End-to-end Unsupervised Outlier Detection via Inlier Priority of Discriminative Network

Siqi Wang[1]*, **Yijie Zeng**[2]*, **Xinwang Liu**[1], **En Zhu**[1], **Jianping Yin**[3], **Chuanfu Xu**[1], **Marius Kloft**[4]
[1]National University of Defense Technology, [2]Nanyang Technological University
[3]Dongguan University of Technology, [4]Technische Universität Kaiserslautern
wangsiqi10c@nudt.edu.cn, yzeng004@e.ntu.edu.sg, {xinwangliu, enzhu}@nudt.edu.cn
jpyin@dgut.edu.cn, xuchuanfu@nudt.edu.cn, kloft@cs.uni-kl.de

NIPS 2019

Abstract

Despite the wide success of deep neural networks (DNN), little progress has been made on end-to-end unsupervised outlier detection (UOD) from high dimensional data like raw images. In this paper, we propose a framework named E^3*Outlier*, which can perform UOD in a both ***effective*** and ***end-to-end*** manner: First, instead of the commonly-used autoencoders in previous end-to-end UOD methods, E^3*Outlier* for the first time leverages a discriminative DNN for better representation learning, by using *surrogate supervision* to create multiple pseudo classes from original unlabelled data. Next, unlike classic UOD that utilizes data characteristics like density or proximity, we exploit a novel property named *inlier priority* to enable end-to-end UOD by discriminative DNN. We demonstrate theoretically and empirically that the intrinsic class imbalance of inliers/outliers will make the network prioritize minimizing inliers' loss when inliers/outliers are indiscriminately fed into the network for training, which enables us to differentiate outliers directly from DNN's outputs. Finally, based on inlier priority, we propose the negative entropy based score as a simple and effective outlierness measure. Extensive evaluations show that E^3*Outlier* significantly advances UOD performance by up to 30% AUROC against state-of-the-art counterparts, especially on relatively difficult benchmarks.

Outlier Detection (OD)

[7]: **1)** *Supervised* OD (SOD) deals with the case where a training set is provided with both labelled inliers/outliers, but it suffers from expensive data labelling and the rarity of outliers in practice [6]. **2)** *Semi-supervised* OD (SSOD) only requires pure single-class training data that are labelled as "inlier" or "normal", and no outlier is involved during training. **3)** *Unsupervised* OD (UOD) handles completely unlabelled data mixed with outliers, and no data label is provided for training at all.

In this paper we will limit our discussion to **UOD**, as most data are unlabelled in practice and UOD is

Contributions

Contributions. This paper proposes an **e**ffective and **e**nd-to-**e**nd UOD framework named E^3*Outlier*. Specifically, our contributions can be summarized below: **1)** To liberate DNN based UOD from AE/CAE's ineffective representation learning, E^3*Outlier* for the first time enables us to adopt powerful discriminative DNN architectures like ResNet [19] for representation learning in UOD. This is realized by *surrogate supervision*, which creates multiple pseudo classes by imposing various simple operations on original unlabelled data. **2)** E^3*Outlier* discovers outliers based on a novel property of discriminative network named *inlier priority*, which evidently differs from previous methods that utilize certain data characteristics (e.g. density, proximity, distance) to perform UOD. Through both theory and experiments, we demonstrate that inlier priority will encourage the network to prioritize the reduction of inliers' loss during network training. On the foundation of inlier priority, E^3*Outlier* is able to achieve end-to-end UOD by directly inspecting the DNN's outputs, which reflect each datum's priority level. In this way, it avoids the possible suboptimal performance yielded by feeding the DNN's learned representations into a decoupled UOD method [20]. **3)** Based on inlier priority, we explore several strategies and propose a simple and effective negative entropy based score to measure outlierness. Extensive experiments report a remarkable improvement by E^3*Outlier* against state-of-the-art methods, particularly on relatively difficult benchmarks for unsupervised tasks.

DNN based Outlier Detection. DNN's recent success naturally inspires DNN based OD [20]. For SOD, discriminative DNN can be directly applied, while the main issue is the class imbalance of inliers/outliers [20], which is explored by [43, 44, 45, 46]. For SSOD, the case is more difficult as only labelled inliers are provided. DNN solutions for SSOD fall into three types: Mainstream DNN based SSOD methods handle high dimensional data by label-free generative models, i.e. AE/CAE [47, 48, 49, 50] and generative adversarial network (GAN) [51, 52, 53]. The second type extends classic SSOD methods into their deep counterparts, such as deep support vector data description [54] and deep one-class SVM [55]. The last type turns SSOD into SOD by certain means like introducing reference datasets [56], intra-class splitting [57], geometric transformations [58] or synthetic outlier generation [59]. As to UOD, the absence of both inlier and outlier label poses great challenges to combining UOD with DNN, which results in much less progress than SOD and SSOD. In addition to the naive solution that feeds DNN's learned representations into a separated UOD method [20], to our best knowledge only the following works have explored DNN based UOD: Zhou et al. [17] propose a decoupled solution that combines a deep AE with Robust PCA, which decomposes the inputs into a low-rank part from inliers and a sparse part from outliers; For end-to-end UOD, Xia et al. [16] use deep AE directly and propose a variant that estimates inliers by seeking a threshold that maximizes the inter-class variance of AE's reconstruction loss. A loss function is designed to encourage the separation of estimated inliers/outliers; Zong et al. [18] jointly optimize a deep AE and an estimation network to perform simultaneous representation learning and density estimation for end-to-end UOD.

Surrogate Supervision

Surrogate Supervision. Recent studies propose surrogate supervision to improve DNN pre-training for downstream high-level tasks like image classification and object detection. It imposes certain operations on unlabelled data to create corresponding pseudo classes and provide supervision signal, such as rotation [60], image patch permutation [61], clustering [62], etc. Surrogate supervision is also called self-supervision (see [63] for a comprehensive survey), but we use surrogate supervision to better distinguish it from AE/CAE, which are also viewed as "self-supervised" in some context. To our best knowledge, our work is the first to connect surrogate supervision with end-to-end UOD.

Goal

Problem Formulation of UOD. Considering a data space $\mathcal{X}$ (in this context the space of images), an unlabelled data collection $X \subseteq \mathcal{X}$ consists of an inlier set X_{in} and an outlier set X_{out}, which originate from fundamentally different underlying distributions [1]. Our goal is to obtain an end-to-end UOD method $S(\cdot)$ that in the ideal case outputs $S(\mathbf{x}) = 1$ for inlier $\mathbf{x} \in X_{in}$ and $S(\mathbf{x}) = 0$ for outlier $\mathbf{x} \in X_{out}$. In practice, a smaller $S(\mathbf{x})$ indicates a higher likelihood of $\mathbf{x}$ to be an outlier.

Why not AutoEncoder?

Why NOT AE/CAE? We note that existing DNN based UOD methods rely on AE/CAE [16, 17, 18]. However, it is hard for them to handle relatively complex datasets like CIFAR10 and SVHN: As our UOD experiments[2] show in Fig. 1(b), even a sophisticated deep CAE with isolation forest [40] only performs slightly better than random guessing (50% AUROC). Similar results are reported in other AE/CAE based unsupervised tasks like deep clustering [64, 65]. This is because AE/CAE typically adopt mean square error (MSE) as loss function, which forces AE/CAE to focus on reducing low-level pixel-wise error that is not sensitive to human perception, rather than learning high-level semantic features [66, 67]. Therefore, AE/CAE based representation learning is often ineffective.

Surrogate Supervision

Surrogate Supervision. Discriminative DNNs like ResNet [19] and Wide ResNet (WRN) [68] have proved to be highly effective in learning high-level semantic features, but they have not been explored in UOD due to the lack of supervision. To remedy the absence of data labels and substitute AE/CAE, we propose a *surrogate supervision based discriminative network* (SSD) for more effective representation learning in UOD. Specifically, we first define an operation set with K operations $\mathcal{O} = \{O(\cdot|y)\}_{y=1}^{K}$, where y represents the pseudo label associated with the operation $O(\cdot|y)$. Applying an operation $O(\cdot|y)$ to $\mathbf{x}$ can generate a new datum $\mathbf{x}^{(y)} = O(\mathbf{x}|y)$, and all data generated by the operation $O(\cdot|y)$ belong to the pseudo class with pseudo label y. Next, given a datum $\mathbf{x}^{(y')}$, a discriminative DNN with a K-node softmax layer is trained to classify the type of applied operation, i.e. the DNN is supposed to classify $\mathbf{x}^{(y')}$ into the y'-th pseudo class. With $P^{(y)}(\cdot)$ and $\boldsymbol{\theta}$ denoting the probability output by the y-th node of softmax layer and DNN's learnable parameters respectively, DNN's output probability vector for K operations is $P(\mathbf{x}^{(y')}|\boldsymbol{\theta}) = [P^{(y)}(\mathbf{x}^{(y')}|\boldsymbol{\theta})]_{y=1}^{K}$.

Loss Function

To train such a DNN with an unlabelled data collection $X = \{\mathbf{x}_i\}_{i=1}^{N}$, the objective function is:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{SS}(\mathbf{x}_i|\boldsymbol{\theta}) \tag{1}$$

where $\mathcal{L}_{SS}(\mathbf{x}_i|\boldsymbol{\theta})$ is the loss incurred by $\mathbf{x}_i$ under surrogate supervision. When the commonly-used cross entropy loss is used to classify pseudo classes of surrogate supervision, it can be written as:

$$\mathcal{L}_{SS}(\mathbf{x}_i|\boldsymbol{\theta}) = -\frac{1}{K} \sum_{y=1}^{K} \log(P^{(y)}(\mathbf{x}_i^{(y)}|\boldsymbol{\theta})) = -\frac{1}{K} \sum_{y=1}^{K} \log(P^{(y)}(O(\mathbf{x}_i|y)|\boldsymbol{\theta})). \tag{2}$$

Operations

As to the operation set $\mathcal{O}$, each operation $O(\cdot|y) \in \mathcal{O}$ is defined as a combination of one or more basic transformations from the following transformation sets: **1)** *Rotation*: This set's transformations clock-wisely rotate images by a certain degree. **2)** *Flip*: This set's transformations refer to flipping the image or not. **3)** *Shifting*: This set's transformations shift the image by some pixels along x-axis or y-axis. **4)** *Patch re-arranging*: This set's transformations partition the image into several equally-sized patches and re-organize them into a new image by a certain permutation. Based on them, we construct

In short

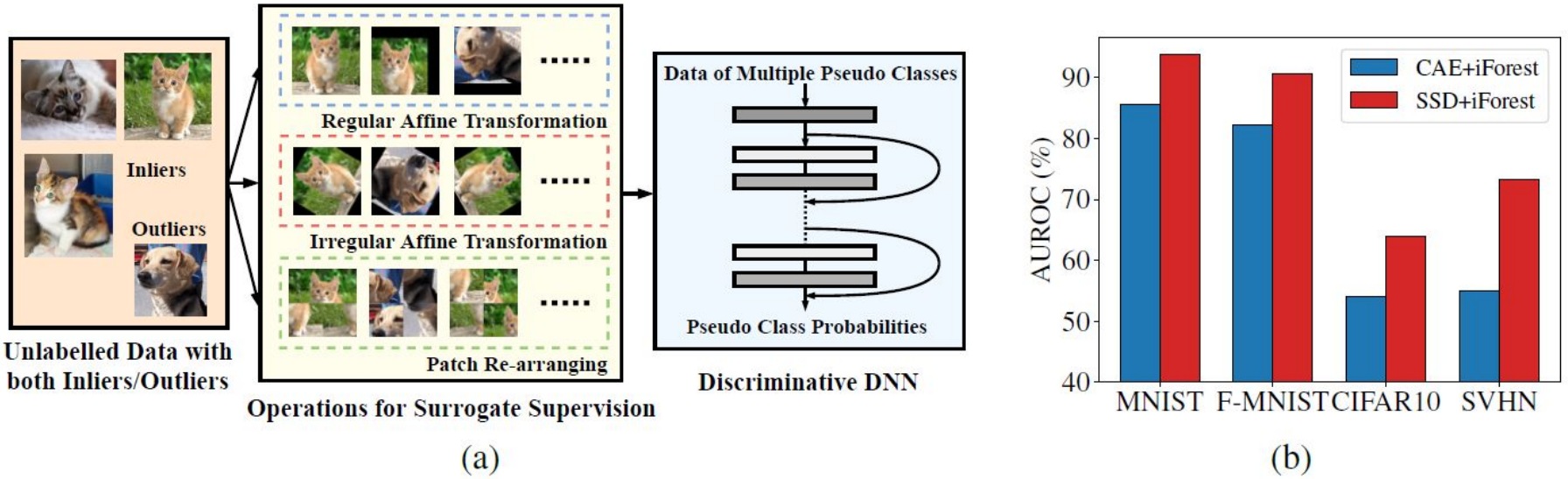


Figure 1: Surrogate supervision workflow (left) and the comparison of learned representations (right).

Utilize Imbalance

3.2 Inlier Priority: The Foundation of End-to-end UOD

Motivation. The above simple solution feeds SSD's learned representations into a decoupled UOD method, which may yield suboptimal performance because SSD and the UOD method are trained separately [18, 20]. Our goal is to achieve end-to-end UOD without using a decoupled UOD method. Recall that outliers are essentially rare patterns in a data collection [7], which implies an intrinsic *class imbalance* between inliers/outliers. Class imbalance is unfavorable in machine learning as it leads to the bias towards majority class during training [70, 71]. However, we argue that class imbalance can be favorably exploited in UOD as it gives rise to "*inlier priority*": ***Despite that inliers/outliers are indiscriminately fed into SSD for training, SSD will prioritize the minimization of inliers' loss***. This intuition naturally inspires an end-to-end UOD solution by measuring how well the SSD's output of a datum matches its target pseudo label, which directly indicates its priority level in training and the likelihood to be an inlier. We demonstrate the inlier priority in terms of two aspects below:

SSD: Surrogate Supervision based Discriminative Network

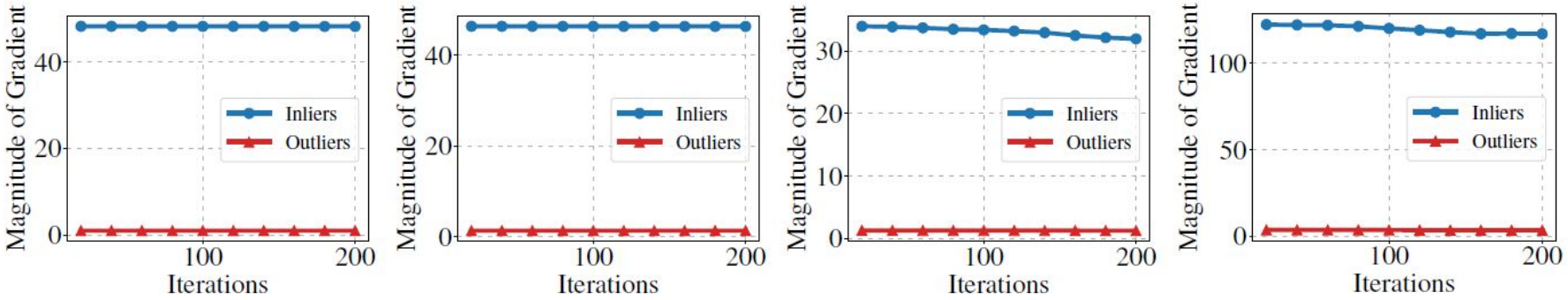
Key Idea 1

Priority by Gradient Magnitude. *Our first point is that inliers will produce gradient with stronger magnitude to update the SSD network than outliers*. To demonstrate this point, we consider an SSD

in back-propagation based optimizer like Stochastic Gradient Descent (SGD) [72]. Given unlabelled data with N_{in} inliers and N_{out} outliers, it is easy to know that $X^{(c)}$ also contains N_{in} transformed inliers and N_{out} transformed outliers. Here we are interested in the magnitude of transformed inliers and outliers' aggregated gradient to update $\mathbf{w}_c$, i.e. $||\nabla^{(in)}_{\mathbf{w}_c}\mathcal{L}||$ and $||\nabla^{(out)}_{\mathbf{w}_c}\mathcal{L}||$, which directly reflect inliers/outliers' strength to affect the training of SSD. Since SSD is randomly initialized, we need to compute the expectation of gradient magnitude. As shown in Sec. 2 of supplementary material, for a simplified SSD network with a single hidden-layer and sigmoid activation, we can quantitatively derive the following approximation on inliers and outliers' gradient magnitude:

$$\frac{E(||\nabla^{(in)}_{\mathbf{w}_c}\mathcal{L}||^2)}{E(||\nabla^{(out)}_{\mathbf{w}_c}\mathcal{L}||^2)} \approx \frac{N_{in}^2}{N_{out}^2} \tag{3}$$

where $E(\cdot)$ denotes the probability expectation. As the class imbalance between inliers and outliers leads to $N_{in} \gg N_{out}$, we naturally yield $E(||\nabla^{(in)}_{\mathbf{w}_c}\mathcal{L}||) \gg E(||\nabla^{(out)}_{\mathbf{w}_c}\mathcal{L}||)$. Therefore, it serves as a theoretical indication that *the gradient magnitude induced by inliers will be significantly larger than outliers for an untrained SSD network*. Since it is particularly difficult to directly analyze more complex network architectures such as Wide ResNet [68], we empirically examine inliers and outliers' gradient magnitude during training by experiments (see Fig. 2), and the observations on different benchmarks are consistent with the above analysis on the simplified case: The magnitude of inliers' aggregated gradient has constantly been larger than outliers during the process of SSD training.



(a) MNIST ("3") (b) F-MNIST ("bag") (c) CIFAR10 ("horse") (d) SVHN ("3")

Figure 2: Inliers and outliers' gradient magnitude on example cases of benchmark datasets during SSD training. The class used as inliers is in brackets.

Key Idea 2

Priority by Network Updating Direction. *Our second point is that the network updating direction of SSD will bias towards the direction that prioritizes reducing inliers' loss during the SSD training*. Since training is dynamic and a theoretical analysis is intractable, we demonstrate this point using an empirical verification by computing inliers/outliers' average "*de facto* update": As illustrated by Fig. 3(a), consider a datum $\mathbf{x}_i$ from a batch of data X, and its negative gradient $-\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x}_i)$ is the fastest network updating direction to reduce $\mathbf{x}_i$'s loss. However, the network weights $\boldsymbol{\theta}$ are actually updated by the negative gradient of the entire batch X, $-\nabla_{\boldsymbol{\theta}}\mathcal{L}(X) = -\frac{1}{N}\sum_i \nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x}_i)$. It is actually different from the best updating direction for each individual datum. Thus, the *de facto* update d_i for $\mathbf{x}_i$ refers to the actual gradient magnitude that $\mathbf{x}_i$ obtains along its best direction for loss reduction from the network update direction $-\nabla_{\boldsymbol{\theta}}\mathcal{L}(X)$, which can be computed by projecting $-\nabla_{\boldsymbol{\theta}}\mathcal{L}(X)$ onto the direction of $-\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x}_i)$: $d_i = -\nabla_{\boldsymbol{\theta}}\mathcal{L}(X) \cdot \frac{-\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x}_i)}{||-\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x}_i)||}$. In this way, d_i reflects how much effort the network will devote to reduce $\mathbf{x}_i$'s loss, and it is a direct indicator of data's priority during network training. We calculate the average *de facto* update of inliers/outliers w.r.t the weights between SSD's penultimate and softmax layer and visualize some examples in Fig. 3(b)-3(d): Although the average *de facto* update of inliers/outliers is very close at the beginning, the average *de*

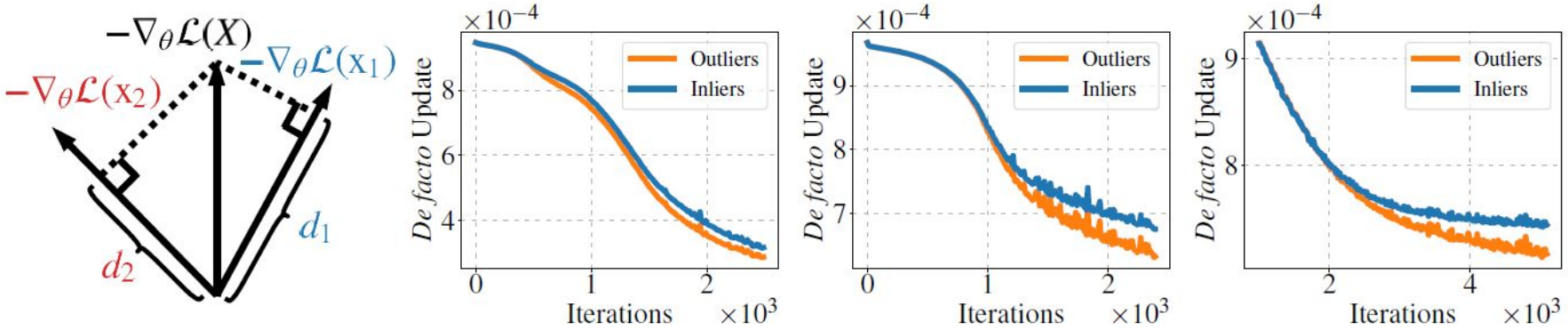


Figure 3: An illustration of *de facto* update and some example cases of the average *de facto* update for inliers/outliers during the network training. The class used as inliers is in brackets.

Remarks

Remarks on Inlier Priority. 1) Based on the discussion above, inliers will gain priority in terms of both the gradient magnitude and the updating direction of SSD's network weights. Such priority leads to a lower loss for inliers after training, which enables us to discern outliers by SSD's outputs and serves as a foundation of end-to-end UOD. **2)** Intuitively, inlier priority will also happen when using AE/CAE based end-to-end UOD methods. However, the effect of inlier priority is severely diminished in this case for two reasons: First, AE/CAE typically uses the raw image pixels as learning targets, but the intra-class difference of inlier images can be very large, which means AE/CAE usually does not have a unified learning target like SSD. Second, AE/CAE is ineffective in learning high-level representations (as we discussed in Sec. 3.1), which makes it difficult to capture common high-level semantics of inlier images. Both factors above disable inliers from being a joint force to dominate the training of AE and produce a strong inlier priority effect like SSD, which is also demonstrated by AE/CAE's poor UOD performance in empirical evaluation (see experimental results in Sec. 4.2).

UOD Scoring 1

Pseudo Label based Score (PL): Inlier priority suggests that SSD will prioritize reducing inliers' loss during training. For the datum $\mathbf{x}^{(y)}$, we note that the calculation of its cross entropy loss only depends on the probability $P^{(y)}(\mathbf{x}^{(y)}|\boldsymbol{\theta})$ that corresponds to its pseudo label y in $P(\mathbf{x}^{(y)}|\boldsymbol{\theta})$. Thus, we propose a direct scoring strategy $S_{pl}(\mathbf{x})$ by averaging $P^{(y)}(\mathbf{x}^{(y)}|\boldsymbol{\theta})$ for all K operations:

$$S_{pl}(\mathbf{x}) = \frac{1}{K}\sum_{y=1}^{K} P^{(y)}(\mathbf{x}^{(y)}|\boldsymbol{\theta}). \tag{4}$$

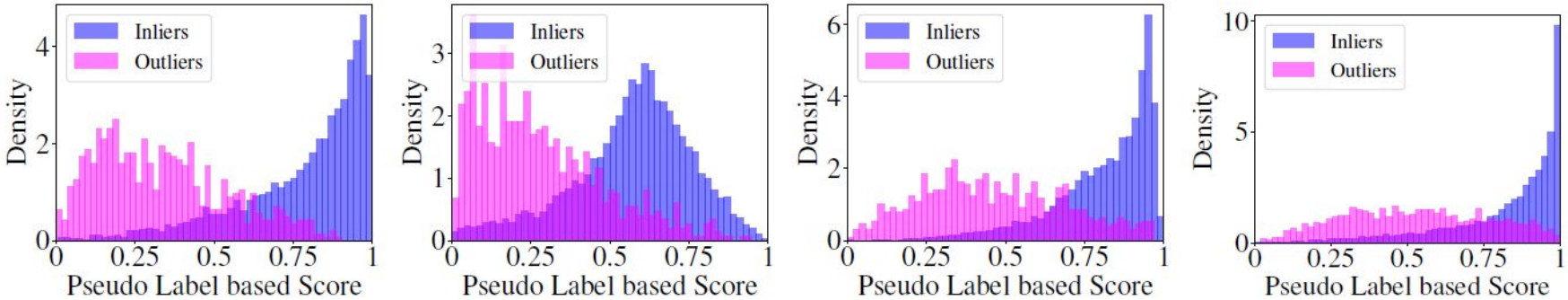


Figure 4: Normalized histograms of inliers/outliers' $S_{pl}(\mathbf{x})$. The class used as inliers is in brackets.

UOD Scoring 2

Maximum Probability based Score (MP): PL seems to be an ideal score. However, we note that operations for surrogate supervision do not always create sufficiently separable classes, e.g. image with a digit "8" is still an "8" when applying a flip operation. Hence, misclassifications will happen and the probability $P^{(y)}(\mathbf{x}^{(y)}|\boldsymbol{\theta})$ that corresponds to pseudo label y may not be the only or the best indicator to reflect how well the loss of a datum is reduced. Therefore, instead of $P^{(y)}(\mathbf{x}^{(y)}|\boldsymbol{\theta})$, we alternatively adopt the maximum probability of $P(\mathbf{x}^{(y)}|\boldsymbol{\theta})$ to calculate the score $S_{mp}(\mathbf{x})$ as follows:

$$S_{mp}(\mathbf{x}) = \frac{1}{K}\sum_{y=1}^{K} \max_{t} P^{(t)}(\mathbf{x}^{(y)}|\boldsymbol{\theta}). \tag{5}$$

UOD Scoring 3

Negative Entropy based Score (NE). Both strategies above rely on a single probability retrieved from $P(\mathbf{x}^{(y)}|\boldsymbol{\theta})$, while the information of the rest $(K-1)$ classes' probability is ignored. If we consider the entire probability distribution $P(\mathbf{x}^{(y)}|\boldsymbol{\theta})$, the training actually encourages SSD to output a probability distribution closer to the label's one-hot distribution. With inlier priority, we can expect SSD to output a sharper probability distribution $P(\mathbf{x}^{(y)}|\boldsymbol{\theta})$ for inliers and a more uniform $P(\mathbf{x}^{(y)}|\boldsymbol{\theta})$ for outliers. Thus, we propose to use information entropy $H(\cdot)$ [73] as a simple and effective measure to the sharpness of a distribution, which gives the negative entropy based score $S_{ne}(\mathbf{x})$:

$$S_{ne}(\mathbf{x}) = -\frac{1}{K}\sum_{y=1}^{K} H(P(\mathbf{x}^{(y)}|\boldsymbol{\theta})) = \frac{1}{K}\sum_{y=1}^{K}\sum_{t=1}^{K} P^{(t)}(\mathbf{x}^{(y)}|\boldsymbol{\theta}) \log(P^{(t)}(\mathbf{x}^{(y)}|\boldsymbol{\theta})). \tag{6}$$

A comparison of PL/MP/NE is given in Sec. 4.2. In Fig. 4(a)-4(d), we calculate the most intuitive $S_{pl}(\mathbf{x})$ of inliers/outliers on benchmarks and visualize the normalized histograms of $S_{pl}(\mathbf{x})$, which are favorably separable for UOD. Besides, such results also verify the effectiveness of inlier priority.

Setup

4.1 Experiment Setup

UOD Performance Evaluation on Image Benchmarks. We follow the standard procedure from previous image UOD literature [13, 16, 17] to construct an image set with outliers: Given a standard image benchmark, all images from a class with one common semantic concept (e.g. "horse", "bag") are retrieved as inliers, while outliers are randomly sampled from the rest of classes by an outlier ratio ρ. We vary ρ from 5% to 25% by a step of 5%. The assigned inlier/outlier labels are strictly unknown to UOD methods and only used for evaluation. Each class of a benchmark is used as inliers in turn and the performance on all classes is averaged as the overall UOD performance. The experiments are repeated for 5 times to report the average results. Five public benchmarks: MNIST [74], Fashion-MNIST (F-MNIST) [75], CIFAR10 [76], SVHN [77], CIFAR100 [76] are used for experiments[3]. Raw pixels are directly used as inputs with their intensity normalized into $[-1, 1]$. As for evaluation, we adopt the commonly-used Area under the Receiver Operating Characteristic curve (AUROC) and Area under the Precision-Recall curve (AUPR) as threshold-independent metrics [78].

Implementation Details and Compared Methods. For $E^3Outlier$, we use an $n = 10$ layer wide ResNet (WRN) with a widen factor $k = 4$ as the backbone DNN architecture. $K = 111$ operations are used for surrogate supervision, and NE is used as the scoring strategy. Since surrogate supervision augments original data by K times, we train WRN for $\lceil \frac{250}{K} \rceil$ epochs. The batch size is 128. A learning rate 0.001 and a weight decay 0.0005 are adopted. The SGD optimizer with momentum 0.9 is used for MNIST and F-MNIST, while the Adam optimizer with $\boldsymbol{\beta} = (0.9, 0.999)$ is used for CIFAR10, CIFAR100 and SVHN for better convergence. We compare $E^3Outlier$ with the baselines and existing state-of-the-art DNN based UOD methods (reviewed in Sec. 2) below: **1)** CAE [79]. It directly uses CAE's reconstruction loss to perform UOD. **2)** CAE-IF. It feeds CAE's learned representations into isolation forest (IF) [40] as explained in Sec. 3.1. **3)** Discriminative reconstruction based autoencoder (DRAE) [16]. **4)** Robust deep autoencoder (RDAE) [17]. **5)** Deep autoencoding gaussian mixture model (DAGMM) [18]. **6)** SSD-IF. It shares $E^3Outlier$'s SSD part but feeds SSD's learned representations into IF to perform UOD. For all AE based UOD methods above, we adopt the same CAE architecture from [58] with a 4-layer encoder and 4-layer decoder. We do not use more complex CAE (e.g. CAE using skip connection [80] or more layers) since they usually lower outliers' reconstruction error as well and do not contribute to CAE's UOD performance. The hyperparameters of the compared methods are set to recommended values (if provided) or the values that produce the best performance. More implementation details are given in Sec. 1 of the supplementary material. Our codes and results can be verified at `https://github.com/demonzyj56/E3Outlier`.

Outlier ratio

Table 1: AUROC/AUPR-in/AUPR-out (%) for UOD methods. The best performance is in bold.

Dataset	ρ	CAE	CAE-IF	DRAE	RDAE	DAGMM	SSD-IF	*E*^{3}*Outlier*
MNIST	10%	68.0/92.0/32.9	85.5/97.8/49.0	66.9/93.0/30.5	71.8/93.1/35.8	64.0/92.9/26.6	93.8/99.2/**68.7**	**94.1/99.3**/67.5
	20%	64.0/82.7/40.7	81.5/93.6/57.2	67.2/86.6/42.5	67.0/84.2/43.2	65.9/86.4/41.3	90.5/97.3/71.0	**91.3/97.6/72.3**
F-MNIST	10%	70.3/94.3/29.3	82.3/97.2/40.3	67.1/93.9/25.5	75.3/95.8/31.7	64.0/92.7/30.3	90.6/98.5/68.6	**93.3/99.0/75.9**
	20%	64.4/85.3/36.8	77.8/92.2/49.0	65.7/86.9/36.6	70.9/89.2/41.4	66.0/86.7/43.5	87.6/95.6/71.4	**91.2/97.1/78.9**
CIFAR10	10%	55.9/91.0/14.4	54.1/90.2/13.7	56.0/90.7/14.7	55.4/90.7/14.0	56.1/91.3/15.6	64.0/93.5/18.3	**83.5/97.5/43.4**
	20%	54.7/81.6/25.5	53.8/80.7/25.3	55.6/81.7/26.8	54.2/81.0/25.7	54.7/81.8/26.3	60.2/85.0/28.3	**79.3/93.1/52.7**
SVHN	10%	51.2/90.3/10.6	55.0/91.4/11.9	51.0/90.3/10.5	52.1/90.6/10.8	50.0/90.0/19.3	73.4/95.9/22.0	**86.0/98.0/36.7**
	20%	50.7/80.2/20.7	54.0/82.0/22.4	50.6/80.4/20.5	51.8/80.9/21.1	50.0/79.9/29.6	69.2/89.5/33.7	**81.0/93.4/47.0**
CIFAR100	10%	55.2/91.0/14.5	54.5/90.7/13.8	55.6/90.9/15.0	55.8/90.9/15.0	54.9/91.1/14.2	55.6/91.5/13.0	**79.2/96.8/33.3**
	20%	54.4/81.7/25.6	53.5/80.9/25.1	55.5/81.8/27.0	54.9/81.5/26.5	53.8/81.5/24.7	54.3/82.1/23.4	**77.0/92.4/46.5**

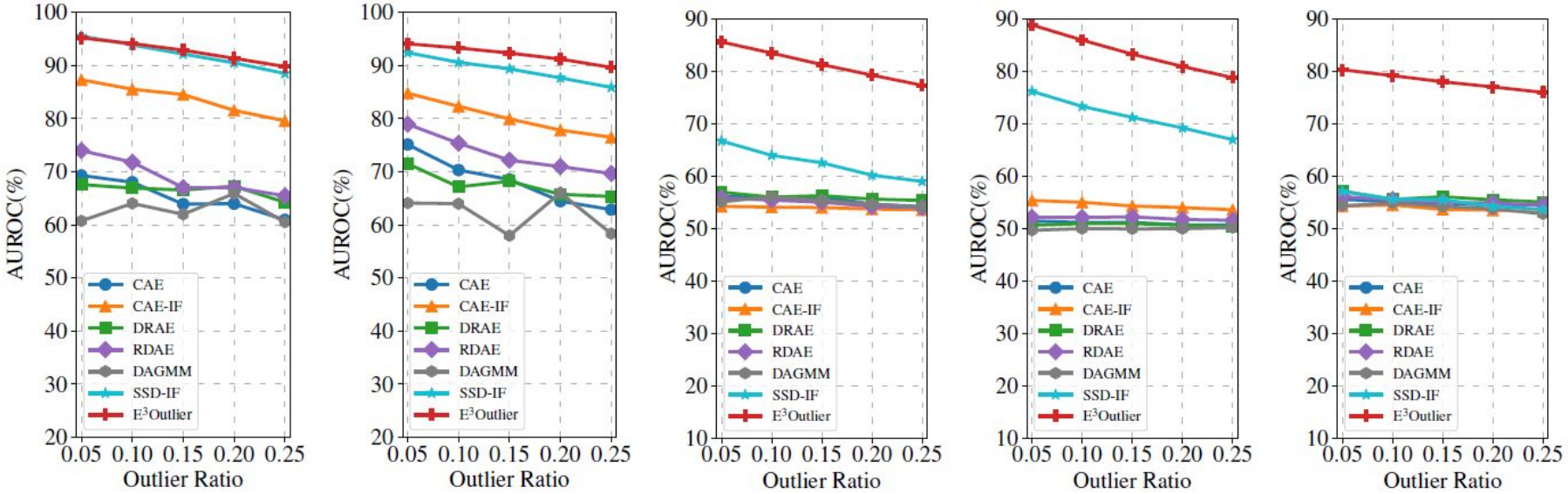


Figure 5: UOD performance (AUROC) comparison with varying ρ from 5% to 25%.

Discussion

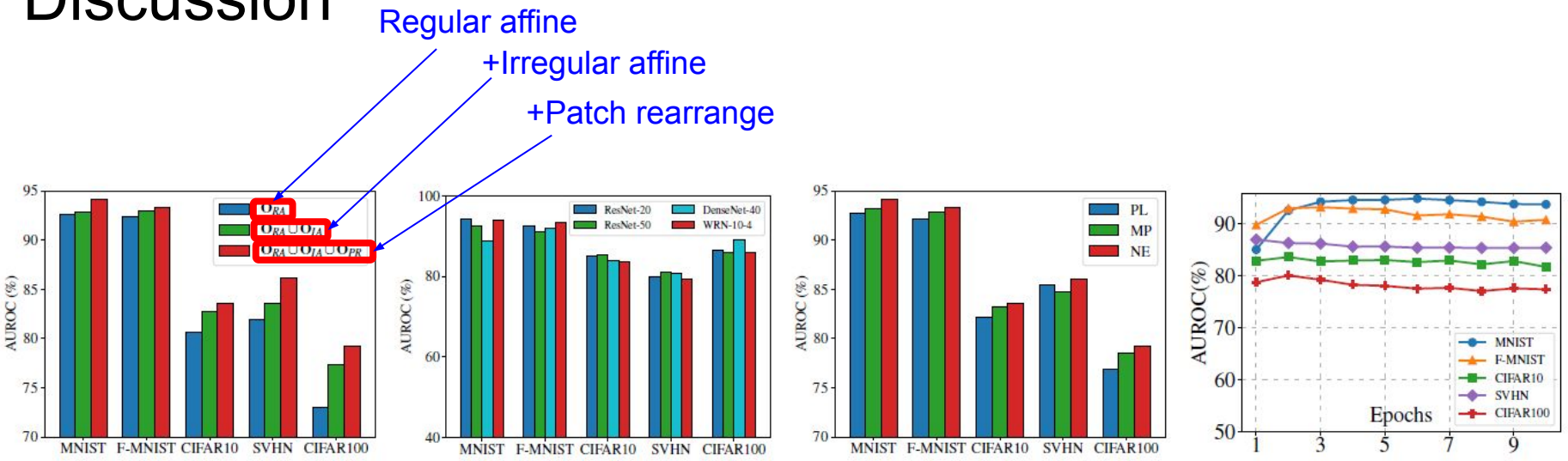


Figure 6: Different factors' influence on E^3*Outlier*'s performance under $\rho = 10\%$.

Questions?

Quantum Entropy Scoring for Fast Robust Mean Estimation and Improved Outlier Detection

Yihe Dong
Microsoft Research
yihedong@gmail.com

Samuel B. Hopkins
University of California, Berkeley
hopkins@berkeley.edu

Jerry Li
Microsoft Research
jerrl@microsoft.com

NIPS 2019

Abstract

We study two problems in high-dimensional robust statistics: *robust mean estimation* and *outlier detection*. In robust mean estimation the goal is to estimate the mean μ of a distribution on $\mathbb{R}^d$ given n independent samples, an ε-fraction of which have been corrupted by a malicious adversary. In outlier detection the goal is to assign an *outlier score* to each element of a data set such that elements more likely to be outliers are assigned higher scores. Our algorithms for both problems are based on a new outlier scoring method we call QUE-scoring based on *quantum entropy regularization*. For robust mean estimation, this yields the first algorithm with optimal error rates and nearly-linear running time $\widetilde{O}(nd)$ in all parameters, improving on the previous fastest running time $\widetilde{O}(\min(nd/\varepsilon^6, nd^2))$. For outlier detection, we evaluate the performance of QUE-scoring via extensive experiments on synthetic and real data, and demonstrate that it often performs better than previously proposed algorithms. Code for these experiments is available at `https://github.com/twistedcubic/que-outlier-detection`.

Goal

We study outlier-robust statistics in high dimensions, focusing on the question: *can theoretically sound outlier robust algorithms have practical running times for large, high-dimensional data sets?* We address two related problems: *robust mean estimation*, which is primarily theoretical, and an applied counterpart, *outlier detection*.

For us, an outlier is an element of a data set which was generated according to a different process than the majority of the data. For instance, we may imagine that our samples $X_1, \ldots, X_n$ were sampled i.i.d. from a distribution $(1-\varepsilon)D + \varepsilon N$ over $\mathbb{R}^d$, where D is the distribution of inliers, N is the distribution of outliers, and $\varepsilon > 0$ is a small number – that is, we imagine that a *constant fraction* of our data may be outliers.

Robust mean estimation Our main theoretical contribution is the first nearly-linear time algorithm for robust mean estimation with nearly-optimal error. Here the goal is to estimate the mean $\mu \in \mathbb{R}^d$ of a d-dimensional distribution D given ε-*corrupted* samples $X_1, \ldots, X_n$ – that is, i.i.d. samples, an unknown ε-fraction of which have been maliciously corrupted. Under (for instance) the assumption that the covariance of D is bounded by Id, it has been long known to be possible in exponential time to estimate μ by $\hat{\mu}$ having $\|\mu - \hat{\mu}\|_2 \leq O(\sqrt{\varepsilon})$. In particular, this rate of error is independent of d.

Polynomial-time algorithms provably achieving such d-independent error became known only recently, starting with the works [8, 15]. Until our work, the running time of algorithms with provably d-independent error remained suboptimal by polynomial factors in d or ε: the fastest running time achieved before this work was $\widetilde{O}(\min(nd^2, nd/\varepsilon^6))$ [6, 8, 15, 9]. (Here $\widetilde{O}(\cdot)$ notation hides logarithmic factors in n and d). While these running times represent a dramatic improvement over previous exponential-time algorithms, there are still many interesting regimes where the additional runtime overheads these algorithms incur render them impractically slow. *We give the first algorithm for robust mean estimation with running time* $\widetilde{O}(nd)$ *which achieves error* $\|\mu - \hat{\mu}\|_2 \leq O(\sqrt{\varepsilon})$. Note that this running time is nearly-linear in the input size nd. Similar to prior works, our algorithm has information-theoretically optimal sample complexity and nearly-optimal error rates in both the bounded-covariance and sub-Gaussian regimes.

Outlier detection Our main applied contribution is a new algorithm for high-dimensional outlier detection, which we assess via experiments on both synthetic and real data [1]. Our goal is to take a dataset $X_1, \ldots, X_n \in \mathbb{R}^d$ and assign to each X_i an *outlier score* $\tau_i \geq 0$, so that higher scores τ_i are assigned to points X_i more likely to be outliers. Of course, what constitutes an outlier varies across applications, so no single algorithm for outlier detection is likely to be the best in all domains. We show that our method performs well in settings where individual outliers are difficult to pick out on their own (by, say, their ℓ_2 norms or their distances to nearby points), but still collectively bias empirical statistics such as the mean and covariance.

We compare our method to baselines based on PCA and Euclidean distances, as well as more sophisticated algorithms from existing literature based on nearest-neighbor distances. Our algorithm has nearly-linear running time in theory, and simple implementations in practice incur minimal overhead beyond standard spectral methods, allowing us to run on 1024-dimensional data with no special optimizations and 8192-dimensional data with a fast approximate implementation. It can therefore be used in practice to complement existing approaches to outlier detection in exploratory data analysis.

Challenge 1

Outliers may not be identifiable in isolation. On its own, a typical outlier $X_i \sim N$ may look much like a typical inlier $X_j \sim D$. For instance, it could be $\|X_i\|_2 \approx \|X_j\|_2$, and X_i, X_j may have similar distance to the nearest few neighboring samples, especially in high dimensions where samples are far apart.

Challenge 2

Outliers still introduce bias, collectively Even if individual outliers look innocuous, the collective effect a modified ε-fraction of samples X_i can still substantially change the empirical distribution of $X_1, \ldots, X_n$. As a result, even simple statistical tasks like estimating the mean or covariance of D require sophisticated estimators: naively pruning individual outliers and then employing standard empirical estimators typically leads to far-suboptimal error rates. For example, an ε-fraction of $X_1, \ldots, X_n$ which are all slightly biased in a single direction may shift the empirical mean of $X_1, \ldots, X_n$, but this bias will be difficult to detect by looking at small numbers of samples at once. This also demonstrates that successful outlier detection can require global geometric information about a high-dimensional dataset, such as whether or not a direction exists in which many (say, εn) samples are unusually biased.

Challenge 3

Outliers may be inhomogeneous. Outliers need not exhibit unusual bias in only one direction, or all have the same norm, or lie in a single cluster. Rather, if a dataset exhibits several forms of corruption, there may be as many different-looking kinds of outliers. In the theoretical robust mean estimation setting, the adversary producing ε-corrupted samples may corrupt $\varepsilon n/10$ samples by biasing them in some direction, another $\varepsilon n/10$ samples by unusually enlarging their norms, and so forth.

Filter Algorithm

Assumption on inlier distribution

Recent innovations in robust mean estimation [15, 8] rely on the following crucial observation about ε-corrupted samples $X_1, \ldots, X_n$ from a distribution D with covariance $\Sigma \preceq \mathrm{Id}$. Namely: *any subset* $S \subseteq \{X_1, \ldots, X_n\}$ *of samples which shift the empirical mean by distance more than* $\sqrt{\varepsilon}$ *in some direction* v *also introduce an eigenvalue of magnitude greater than* 1 *to the empirical covariance.*

In robust mean estimation, this leads to (amongst others) the *filter* algorithm of [8, 9], one of the first to achieve dimension-independent error rates. Roughly speaking, the algorithm iterates the following until the empirical covariance $\overline{\Sigma}$ has small spectral norm: (1) compute the top eigenvector v of the empirical covariance of $\overline{\Sigma}$, then (2) throw out samples X_i whose projections $|\langle X_i - \overline{\mu}, v\rangle| \gg 1$ is unusually large, where $\overline{\mu}$ is the empirical mean of the corrupted dataset. For outlier detection this suggests a natural scoring rule – let the outlier score τ_i of sample X_i be proportional to $|\langle X_i - \overline{\mu}, v\rangle|$.

Filter Algorithm : Drawback

The main drawback of these algorithms is that they do not adequately account for inhomogeneity of outliers. For the filter, this leads to a worst-case running time of $\widetilde{O}(nd^2)$, because the filter operation (which can be implemented in $\widetilde{O}(nd)$ time) may have to be repeated as many as d times if the adversary introduces outliers lying in d orthogonal directions. The rule $\tau_i = |\langle X_i - \overline{\mu}, v\rangle|$ may miss outliers causing a large eigenvalue of $\overline{\Sigma}$, but in a direction orthogonal to the top eigenvector v.

In the opposite extreme, if outliers are maximally inhomogeneous – no group of them is unusually biased in some shared direction v – then the only way they can bias the empirical mean is for the individual ℓ_2 norms $\|X_i - \overline{\mu}\|_2$ to be larger than typical. This suggests a different scoring rule: $\tau_i = \|X_i - \overline{\mu}\|_2$. This approach, however, breaks down in the situation we started with, that groups of outliers are biased in a shared direction but they do not have larger norms than good samples.

Main concept

Our main conceptual contribution is an approach to utilize information about outliers beyond what is available in the top eigenvector of the empirical covariance $\overline{\Sigma}$ and in individual ℓ_2 norms. Appropriately adapted to their respective settings, this leads to our algorithms for both robust mean estimation and outlier detection.

Observation 1

Our first observation is that *any eigenvalue/eigenvector λ, v – not just the top ones – of the empirical covariance with $\lambda \gg 1$ must be due to outliers*. We therefore consider the intermediate goal of finding a *distribution* over directions $v \in \mathbb{R}^d$ containing information about as many outlier directions as possible. We formalize this as the following entropy-regularized convex program over $d \times d$ positive semidefinite matrices:

$$\max_{U \in \mathbb{R}^{d \times d}} \alpha \cdot \langle U, \overline{\Sigma} \rangle + S(U) \text{ such that } U \succeq 0, \operatorname{tr}(U) = 1 \,, \tag{1}$$

where $\alpha \geq 0$ is some constant and $\langle A, B \rangle = \operatorname{tr}(AB^\top)$ denotes the trace inner product of matrices. Here, $S(U) = -\langle U, \log U \rangle$ is the *quantum entropy* (also known as the *von Neumann entropy*) of the matrix U. If $U = \sum_{i=1}^d \mu_i v_i v_i^\top$ is the eigendecomposition of U, since it has $\operatorname{tr}(U) = 1$ we may interpret it as a distribution over orthonormal vectors $v_1, \ldots, v_d$ with weights $\mu_1, \ldots, \mu_d$ and hence with entropy $S(U)$. Under this interpetation, $\langle U, \overline{\Sigma} \rangle = \mathbb{E}_{v_i \sim \mu} \langle v_i, \overline{\Sigma} v_i \rangle$. As α varies, (1) trades off optimizing for a distribution supported on many distinct directions for a distribution supported on eigenvectors of $\overline{\Sigma}$ with large eigenvalues. The optimizer of (1) takes the form $U = \exp(\alpha \cdot \overline{\Sigma}) / \operatorname{tr} \exp(\alpha \cdot \overline{\Sigma})$ where $\exp(\cdot)$ is the matrix exponential function.

Quantum Entropy scores

Definition 1.1. Let $U = \exp(\alpha \cdot \overline{\Sigma})/\mathrm{tr}\exp(\alpha \cdot \overline{\Sigma})$ be the optimizer of (1), for some data set $\mathcal{X} = X_1, \ldots, X_n \in \mathbb{R}^d$ where $\overline{\Sigma}$ is the covariance of $\mathcal{X}$. The quantum entropy (QUE) scores with parameter α are given by $\tau_i = (X_i - \overline{\mu})^\top U (X_i - \overline{\mu})$, where $\overline{\mu}$ is the mean of $\mathcal{X}$.

Intuitively, the QUE scores will penalize any point which is causing a large eigenvalue in any direction, which should allow us to find more outliers than the naive spectral scores presented above. QUE scores also interpolate between two more naive scoring rules: when $\alpha = 0$ we have $U = \mathrm{Id}/d$ and so $\tau_i = \frac{1}{d}\|X_i - \overline{\mu}\|_2^2$ is the ℓ_2 norm (up to a scaling), while when $\alpha \to \infty$ we have $U \to vv^\top$ where v is the top eigenvector of $\overline{\Sigma}$, recovering naive spectral scoring. *In both experiments and theory we find that choosing α strictly between 0 and ∞ outperforms either of the extreme choices.*

Stronger model

Definition 1.2 (ε-corrupted samples)**.** Let D be a distribution on $\mathbb{R}^d$. We say that $X_1, \ldots, X_n$ are an ε-corrupted set of samples from D if they are first drawn i.i.d. from D, then modified by an adversary who may adaptively inspect all the samples, remove εn of them, and replace them with arbitrary vectors in $\mathbb{R}^d$.

Theorem

Theorem 1.1. *For every $n, d \in \mathbb{N}$ and $\varepsilon > 0$ there are algorithms* QUESCOREFILTER ,S.G.-QUESCOREFILTER *with running time $\widetilde{O}(nd)$, such that for every distribution D on $\mathbb{R}^d$ with mean μ and covariance Σ, given n ε-corrupted samples from D,* QUESCOREFILTER *produces $\hat{\mu}$ such that $\|\hat{\mu} - \mu\|_2 \leq O(\sqrt{\varepsilon}) + \widetilde{O}(\sqrt{d/n})$ if $\Sigma \preceq \mathrm{Id}$, and* S.G.-QUESCOREFILTER *produces $\hat{\mu}$ such that $\|\hat{\mu} - \mu\|_2 \leq O(\varepsilon\sqrt{\log(1/\varepsilon)} + \sqrt{d/n})$ if D is sub-Gaussian with $\Sigma = \mathrm{Id}$, all with probability at least 0.99.*

For the bounded covariance case, the $O(\sqrt{\varepsilon})$ term information-theoretically optimal up to constant factors. The other term, $\widetilde{O}(\sqrt{d/n})$, is information-theoretically optimal up to the logarithmic factors in the $\widetilde{O}(\cdot)$ even without corruptions. For the sub-Gaussian case, the $O(\varepsilon\sqrt{\log 1/\varepsilon})$ term is believed to be necessary for computationally efficient algorithms (see e.g the statistical-query lower bound [10]), although that term can be made $O(\varepsilon)$ by using computationally-intractable estimators such as Tukey median, and the latter is information-theoretically optimal [26]. The $\sqrt{d/n}$ term is information-theoretically optimal even without corruptions.

Definition 1.3 (Simplified robust mean estimation)**.** Let $S = \{X_1, \ldots, X_n\} \subseteq \mathbb{R}^d$ be a dataset with the property that S partitions into $S = S_g \cup S_b$ with $|S_b| \leq \varepsilon n$ and $\mathbb{E}_{i \sim S_g}(X_i - \mu_g)(X_i - \mu_g)^\top \preceq \mathrm{Id}$, where $\mu_g = \mathbb{E}_{i \sim S_g} X_i$. Given S, the goal is to find a vector $\hat{\mu}$ with $\|\mu_g - \hat{\mu}\|_2 \leq O(\sqrt{\varepsilon})$.

Like prior algorithms for robust mean estimation, ours maintains a *weight vector* $w_1, \ldots, w_n \geq 0$ with $\sum w_i \leq 1$, initialized to $w_i = 1/n$. The algorithm iteratively decreases the weight of points suspected to be outliers that are causing $\|\mu(w) - \mu_g\|_2$ to be large.[2] A key insight of recent work on robust mean estimation is that it suffices to find weights w which place almost as much mass on S_g as does the uniform weighting and whose empirical covariance is small. This is formalized in the following lemma. For a weight vector w, let $|w| = \sum w_i$, $\mu(w) = \frac{1}{|w|} \sum w_i X_i$, and $M(w) = \frac{1}{|w|} \sum w_i (X_i - \mu(w))(X_i - \mu(w))^\top$. Let $\|M\|_2$ be the spectral norm of a matrix M.

Lemma 1.2 (Implicit in prior work)**.** *Let $S = \{X_1, \ldots, X_n\}$ be as in Definition 1.3. Suppose that w is a weight vector such that $\|M(w)\|_2 \leq O(1)$ and w is* mostly good*, by which we mean $|\frac{1}{n}\mathbf{1}_{S_g} - w_g| \leq |\frac{1}{n}\mathbf{1}_{S_b} - w_b|$, where $\mathbf{1}_{S_g}, \mathbf{1}_{S_b}$ are the indicators of S_g, S_b and w_g, w_b are w restricted to S_g, S_b respectively. (Intuitively, w is mostly good if it results by removing from the uniform weighting $\mathbf{1}_S/n$ more weight from S_b than from S_g.) Then $\|\mu(w) - \mu_g\|_2 \leq O(\sqrt{\varepsilon})$.*

Lemma 1.2 captures the following geometric intuition: if the bad points S_b receive enough weight in w to cause $\|\mu(w) - \mu_g\|_2 \gg \sqrt{\varepsilon}$, then an $O(\varepsilon)$-fraction of the mass of w is on X_i which are unusually correlated with the vector $\mu(w) - \mu_g$, which leads to a large maximum eigenvalue in $M(w)$. Prior works employ a variety of methods to find a mostly good weight vector w with $\|M(w)\|_2 \leq O(1)$. Perhaps the simplest is the *filter* of [8], which iterates: While $\|M(w)\|_2 \gg 1$, compute its top eigenvector v and *naive spectral scores* $\tau_i = \langle X_i - \mu(w), v\rangle^2$. Throw out X_i with large τ_i and repeat.

The filter ensures that the weight vector it maintains is mostly good because (in an averaged sense) τ_i can be large only for X_i which are corrupted. This is because the (weighted) sum of all scores $\sum w_i \tau_i = \langle M(w), vv^\top \rangle \gg 1$, while the contribution to this sum from S_g has $\sum_{i \in S_g} w_i \tau_i \approx \langle \frac{1}{n} \sum_{i \in S_g} (X_i - \mu_g)(X_i - \mu_g)^\top, vv^\top \rangle \leq 1$. (Here we ignore some details about centering X_i at μ_g rather than $\mu(w)$.) Thus, the τ_i from S_b must make up almost all of $\sum w_i \tau_i$. Simple approaches to removing or downweighting X_i with large τ_i then remove strictly more weight from S_b than from S_g.

Main Idea

Our main idea is that by replacing naive spectral scores with slightly modified QUE scores, each iteration of the filter can take into account projections of each sample onto many large eigenvectors of $M(w)$. We show that our modified QUE scores τ_i maintain the property that $\sum_{i \in S_b} w_i \tau_i \gg \sum_{i \in S_g} w_i \tau_i$, and so downweighting according to τ_i removes more mass from S_b than S_g. However, filtering with QUE scores makes faster progress than with naive spectral scores: roughly speaking, we show that only $O(\log d)^2$ rounds of filtering according to QUE scores are required to find a mostly-good weight vector w with $\|M(w)\|_2 \leq O(1)$.

The core of our algorithm is a subroutine, DECREASESPECTRALNORM, to take a mostly good weight vector w with $\|M(w)\|_2 \gg 1$ and in $O(\log d)$ rounds of QUE filtering produce another mostly good w' with $\|M(w')\|_2 \leq \frac{3}{4}\|M(w)\|_2$. Repeating this subroutine $O(\log d)$ times and then outputting the resulting $\mu(w)$ yields our main algorithm. An outline of this subroutine is presented as Algorithm 1. We first establish a rigorous sense in which downweighting according to outlier scores τ_i makes progress: *it decreases the weighted average of the scores while removing more weight from bad points than good.*

Lemma 1.3 (Progress in one round of downweighting, informal)**.** *There is a downweighting algorithm which takes a density matrix U and a mostly good weight vector w and produces a mostly good weight vector w' by downweighting points with large score $\tau_i = \langle X_i - \mu(w), U(X_i - \mu(w))\rangle$ such that $\sum w_i'\tau_i \leq \frac{1}{3}\sum w_i\tau_i$ so long as $\sum w_i\tau_i \gg 1$. Furthermore, $M(w') \preceq M(w)$.*

This guarantee becomes more meaningful as the entropy $S(U)$ increases, because it suggests the quadratic form of $M(w)$ has decreased in more directions. To make this formal, we appeal to the matrix multiplicative weights framework. DECREASESPECTRALNORM applies downweighting iteratively using a sequence of entropy-maximizing density matrices $U_1, \ldots, U_T$ chosen according to the matrix multiplicactive weights update rule, leading to a series of mostly good weight vectors $w_1, \ldots, w_T$ such that $\|M(w_T)\|_2 \leq \frac{3}{4}\|M(w_0)\|_2$. We choose

$$U_t = \exp\left(\frac{1}{\|M(w)\|_2}\sum_{k=0}^{t-1} M(w_k)\right) \Big/ \operatorname{tr}\exp\left(\frac{1}{\|M(w)\|_2}\sum_{k=0}^{t-1} M(w_k)\right), \tag{3}$$

Algorithm 1 DECREASESPECTRALNORM

1: **Input:** $X_1, \ldots, X_n$ as in Definition 1.3, mostly good weight vector w_0.
2: For iteration $t = 0, \ldots, O(\log d)$, if $\|M(w_t)\|_2 \leq \frac{3}{4}\|M(w_0)\|_2$, output w_t and halt. Otherwise, let U_t as in (4). If $\langle U_t, M(w_{t-1})\rangle \leq \frac{1}{3}\|M(w_0)\|_2$, let $w_{t+1} = w_t$. Else let w_{t+1} be the output of downweighting from Lemma 1.3 with U_t.
3: Output w_T.

Experimental setup: We must work with data containing well-defined and known inliers and outliers so that we can compare our results to ground-truth. We generate such data sets in three distinct ways, leading to three main experiments. (In supplemental material we also study some outlier-detection data sets appearing in prior work [5].)

Synthetic: We create synthetic data sets in 128 dimensions and $10^3 - 10^4$ samples with an ε-fraction of inhomogeneous outliers in k directions by sampling from a mixture of $k+1$ Gaussians $(1-\varepsilon)\mathcal{N}(0, \mathrm{Id}) + \sum_{i=1}^{k} \varepsilon_i [\frac{1}{2}\mathcal{N}(C\sqrt{k/\varepsilon} \cdot e_i, \sigma^2 \mathrm{Id}) + \frac{1}{2}\mathcal{N}(-C\sqrt{k/\varepsilon} \cdot e_i, \sigma^2 \mathrm{Id})]$, where $e_1, \ldots, e_k$ are standard basis vectors, with $C \approx 1$ and $\sigma \ll 1$. The outliers are the samples from $\mathcal{N}(\pm C\sqrt{k/\varepsilon} e_i, \sigma^2 \mathrm{Id})$. By varying ε, k and the distribution $\varepsilon_1, \ldots, \varepsilon_k$ of outlier weights, we demostrate in this simplified model how max-entropy outlier scoring improves on baseline algorithms in the presence of inhomogeneous outliers. We choose the scaling $\sqrt{k/\varepsilon} \cdot e_i$ because then standard calculations predict that if $\varepsilon_i \approx \varepsilon/k$ the outliers from $\mathcal{N}(\pm C\sqrt{k/\varepsilon} e_i, \sigma^2 \mathrm{Id})$ will contribute an eigenvalue greater than 1 to the overall empirical covariance.

Mixed – word embeddings: We create a data set consisting of word embeddings drawn from several sources. Inliers are the 100-dimensional GloVe embeddings ([21]) of the words in a random $\approx 10^3$ word long section of a novel (we use *Sherlock Holmes*) and outliers are embeddings of the first paragraphs of k featured Wikipedia articles from May 2019 [27].

Perturbed – images: We create a data set consisting of CIFAR10 images some of which have artificially-introduced *dead pixels*. Inliers are ≈ 4500 random CIFAR images $X \in \{1, \ldots, 256\}^{1024}$

Metric: All the methods we evaluate produce a vector of *scores* $\tau_1, \ldots, \tau_n \in \mathbb{R}$. We use the standard *ROCAUC* metric to compare these scores to a ground-truth partition $S = S_g \cup S_b$ into inlier and outlier sets. $\text{ROCAUC}(\tau_1, \ldots, \tau_n, S_b, S_g) = \Pr_{i \sim S_b, j \sim S_g}(\tau_i \geq \tau_j)$ is simply the probability that a randomly chosen outlier is scored higher than a random inlier.

Baselines: We compare QUE scoring to the following other scoring rules. ℓ_2: $\tau_i = \|X_i - \overline{\mu}\|$ is the distance of X_i to the empirical mean; *top eigenvector naive spectral:* $\tau_i = \langle X_i - \overline{\mu}, v \rangle^2$ where v is the top eigenvector of the empirical covariance; *k-nearest neighbors (k-NN)* [22, 5] and *local outlier factor (LOF)* [4, 5] methods: τ_i is a function of the distances to its k nearest neighbors; *isolation forest and elliptic envelope:* standard outlier detection methods as implemented in scikit-learn [23, 18, 20].

Whitening: Scoring methods based on the projection of data points X_i onto large eigenvectors of the empirical covariance work best when those eigenvectors correspond to directions in which many outliers lie. In particular, if Σ_g, the covariance of S_g, itself has large eigenvalues then such spectral methods perform poorly. We assume access to a *whitening transformation* $W \in \mathbb{R}^{d \times d}$, which captures a small amount of prior knowledge about the distribution of inliers S_g. For best performance W should approximate $W^* = (\Sigma_g)^{-1/2}$ since $W^* X_i$ form an isotropic set of vectors. Of course, to compute W^* exactly would require knowing which points are inliers, but we find that relatively naive approximations suffice. In particular, if a clean dataset $Y_1, \ldots, Y_m$ whose distribution is similar to the distribution of inliers is available, its empirical covariance can be used to find a good whitening transformation W. In our synthetic data we use $W = \mathrm{Id}$. In our word embeddings experiment, we obtain W using the empirical covariance of the embedding of another random section of Sherlock Holmes. In our CIFAR-10 experiment, we obtain W from the empirical covariance of a fresh sample of ≈ 5000 randomly chosen images from CIFAR-10.

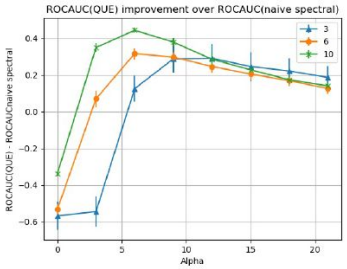
Algorithm 2 QUE-Scoring for Outlier Detection

1: **Input:** dataset $X_1, \ldots, X_n \in \mathbb{R}^d$, optional whitening transformation $W \in \mathbb{R}^{d \times d}$, scalar $\alpha > 0$.

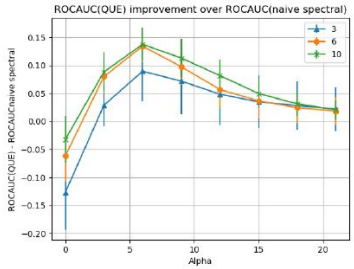
2: Let $X_i' = W X_i$ be whitened data, $\overline{\mu} = \frac{1}{n} \sum_{i=1}^{n} X_i'$ and $\overline{\Sigma} = \frac{1}{n} \sum_{i=1}^{d} (X_i' - \overline{\mu})(X_i' - \overline{\mu})^\top$.

3: For $i \leq n$, let $\tau_i = ({X_i'}^\top \exp(\alpha \overline{\Sigma} / \|\overline{\Sigma}\|_2) X_i') / \operatorname{Tr} \exp(\alpha \overline{\Sigma} / \|\overline{\Sigma}\|_2)$. Return $\tau_1, \ldots, \tau_n$.

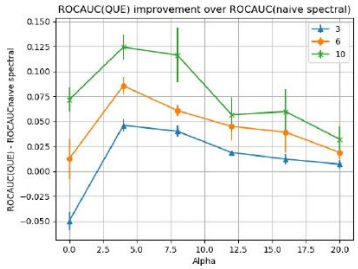
Note on α: in both synthetic and real data we find that $\alpha = 4$ is a good rule-of-thumb choice, consistently resulting in improved scores over baseline methods.



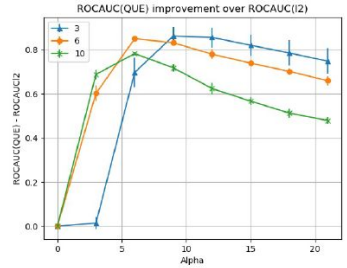
(a) synthetic



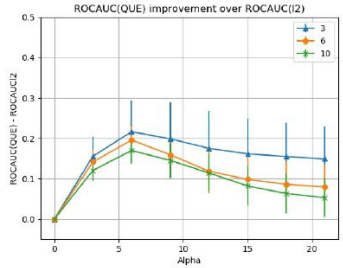
(b) whitened CIFAR-10



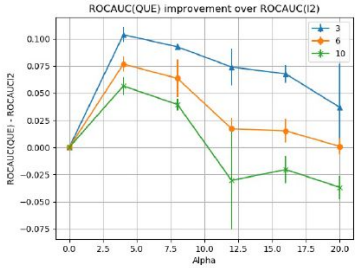
(c) whitened word embeddings



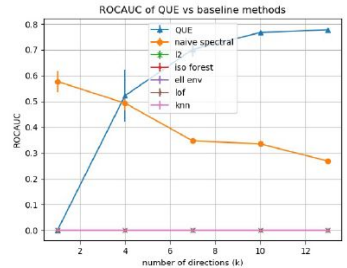
(d) synthetic



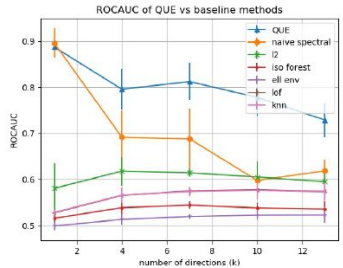
(e) whitened CIFAR-10



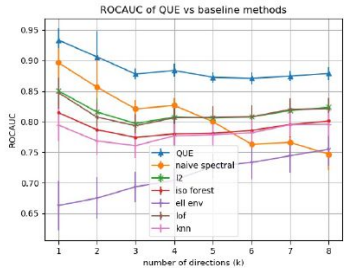
(f) whitened word embeddings



(g) synthetic



(h) whitened CIFAR-10



(i) whitened word embeddings

Questions?

Statistical Analysis of Nearest Neighbor Methods for Anomaly Detection

Xiaoyi Gu[1], **Leman Akoglu**[2], **Alessandro Rinaldo**[1]
[1]Department of Statistics and Data Science, Carnegie Mellon University
[2]Heinz College of Information Systems and Public Policy, Carnegie Mellon University
{xgu1,lakoglu}@andrew.cmu.edu, arinaldo@cmu.edu

NIPS 2019

Abstract

Nearest-neighbor (NN) procedures are well studied and widely used in both supervised and unsupervised learning problems. In this paper we are concerned with investigating the performance of NN-based methods for anomaly detection. We first show through extensive simulations that NN methods compare favorably to some of the other state-of-the-art algorithms for anomaly detection based on a set of benchmark synthetic datasets. We further consider the performance of NN methods on real datasets, and relate it to the dimensionality of the problem. Next, we analyze the theoretical properties of NN-methods for anomaly detection by studying a more general quantity called distance-to-measure (DTM), originally developed in the literature on robust geometric and topological inference. We provide finite-sample uniform guarantees for the empirical DTM and use them to derive misclassification rates for anomalous observations under various settings. In our analysis we rely on Huber's contamination model and formulate mild geometric regularity assumptions on the underlying distribution of the data.

Depending on the availability of data labels, there are multiple setups for anomaly detection. The first is the supervised setup, where labels are available for both normal and anomalous instances during the training stage. Because of its similarity to the standard classification setup, numerous classification methods with good empirical performance and well-studied theoretical properties can be adopted. The second setup is the semi-supervised setup, where training data only comprise normal instances and no anomalies. This setup is widely used in the intrusion detection literature. Well-known methods with theoretical guarantees include kNNG [1], BP-kNNG [2] and BCOPS [3], with the first two methods developed based on the geometric entropy minimization (GEM) principle proposed in [1], and the third on conformal prediction. Methods under this setups are essentially targeting the estimation of high density regions, and treating low density points as anomalies. The third setup is the unsupervised setup, which is the most flexible yet challenging setup. For the rest of the paper, we will only focus on this setup and do not assume any prior knowledge on data labels.

Two versions of the NN anomaly detection algorithms have been proposed: k^{th}NN [20] and kNN [6]. k^{th}NN assigns anomaly score of an instance by computing the distance to its k^{th}-nearest-neighbor, whereas kNN takes the average distance over all k-nearest-neighbors. Both methods are shown to have competitive performance in various comparative studies [21, 22, 12, 23]. In particular, the comparative study developed by Goldstein and Uchida [21] is the one of most comprehensive analysis to date that includes the discussion of NN-methods and, at the same time, aligns with the unsupervised anomaly detection setup. However, the authors omit the analysis of ensemble methods, some of which are considered as state-of-the-art algorithms (e.g., IForest and LODA). Emmott et al. [24] constructed a large corpus (over 20,000) of synthetic benchmark datasets that vary across multiple aspects (e.g., clusteredness, separability, difficulty, etc). The authors evaluate the performance of eight top-performing algorithms, including IForest and LODA, but omit the analysis of NN-methods.

In this section, we provide a comprehensive empirical analysis of NN-methods by comparing kNN, k^{th}NN, and DTM_2[1] to IForest, LOF and LODA on (1) the corpus of synthetic datasets developed in [24], (2) 23 real datasets from the ODDS library [25], and (3) 6 high dimensional datasets from the UCI library [26]. The code for all our experiments are publicly available[2]. In general, no one

2.1 Comparison on Benchmark Datasets

First, we complement Emmott et al.'s study [24] by extending it to NN-based detectors. First, we calculate the ROC-AUC (AUC) and Average Precision (AP) scores for each method on each benchmark, and compute their respective quantiles on the empirical distributions for AUC and AP scores (refer to Appendix E in [24] for more details on treating AUC and AP as random variables). We say that an algorithm fails on a benchmark with metric AUC (or AP) at significance level α if the computed AUC (or AP) quantiles are less than $(1-\alpha)$. Then, the failure rate for each algorithm is found as the percentage of failures over the entire benchmark corpus. The failure rate gives a better

Table 1: Algorithm Failure Rate with Significance Level $\alpha = 0.001$.

	AUC	**AP**	**Either**
ABOD	0.5898	0.6784	0.7000
IForest	**0.5520**	**0.6514**	**0.6741**
LODA	0.6187	0.6955	0.7194
LOF	0.6016	0.7071	0.7331
RKDE	0.6122	0.7030	0.7194
OCSVM	0.7218	0.7342	0.7969
SVDD	0.8482	0.8868	0.9080
EGMM	0.6188	0.7146	0.7303
kNN	0.5646	0.6744	0.6960
k^{th}NN	0.5831	0.6886	0.7100
DTM_2	0.5669	0.6761	0.6977

Table 2: AUC and AP performance on high dimensional datasets

AUC	n	d	IForest	LODA	LOF	kNN	k^{th}NN	DTM_2	DTMF_2
gisette	3850	4970	0.5023	0.5176	0.6753	0.5696	0.5429	0.5692	0.7051
isolet	4886	617	0.5485	0.5421	0.7330	0.6810	0.6480	0.6796	0.7645
letter	4586	617	0.5600	0.5459	0.7846	0.7162	0.6826	0.7149	0.8096
madelon	1430	500	0.5327	0.5427	0.5450	0.5608	0.5552	0.5607	0.5546
cancer	385	30	0.9528	0.9626	0.8097	0.9780	0.9756	0.9773	0.6937
ionosphere	242	33	0.9265	0.9118	0.9450	0.9832	0.9803	0.9824	0.9372

Slightly lower dimension

(a) AUC

AP	n	d	IForest	LODA	LOF	kNN	k^{th}NN	DTM_2	DTMF_2
gisette	3850	4970	0.0877	0.0907	0.1628	0.1093	0.1015	0.1092	0.1723
isolet	4886	617	0.1005	0.1003	0.2343	0.2074	0.1846	0.2070	0.2458
letter	4586	617	0.0956	0.0980	0.2921	0.2328	0.2054	0.2319	0.3010
madelon	1430	500	0.1067	0.0974	0.1171	0.1209	0.1181	0.1209	0.1166
cancer	385	30	0.6274	0.8277	0.3121	0.8813	0.8840	0.8864	0.2800
ionosphere	242	33	0.7222	0.7438	0.6058	0.8903	0.8801	0.8868	0.6105

(b) AP

in Table 2. The n and d columns stand for the number of samples and dimension of the datasets. On datasets *gisette*, *isolet* and *letter*, the performance of IForest and LODA have been significantly downgraded; the NN-methods give somewhat better performance, whereas LOF and DTMF_2 are showing significantly stronger performance. However, on datasets *cancer* and *ionoshphere*, where dimensions are slightly lower, the situations are reversed, with LOF and DTMF_2 giving significantly worse performance than the others. This is consistent with our findings in Section 2.2. The deficiency of IForest in high dimensions is expected, as the IForest trees are generated by random partitioning along a randomly selected feature. However, in high dimensions, there is a high probability that a large number of features are neglected in the process. From another perspective, [29] discusses the various effects of dimensionality in the context of anomaly detection. In particular, the authors describe a concentration effect of distances in high dimensions, which has a negative effect on IForest, or any other methods that rely on pairwise distances of points for computation of anomaly scores. NN-methods, on the other hand, are somewhat more robust in high dimensions, as the rankings of distance values are still feasible.

Overall, our experiments show that IForest and NN-methods are the top two methods with excellent overall performance on both low dimensional synthetic and real datasets. However, NN-methods exhibit better performance than IForest when the data is high dimensional. In the following sections, we provide a theoretical understanding of how the NN-methods work under the anomaly detection framework.

Theoretical Analysis

3.1 Problem Setup

We assume we observe n i.i.d. realizations $\mathbb{X}_n = (X_1, \ldots, X_n)$ from a distribution P on $\mathbb{R}^d$ that follows the Huber contamination model [30, 31]

$$P = (1 - \varepsilon)P_0 + \varepsilon P_1,$$

where P_0 and P_1 are, respectively, the underlying distribution for the normal and anomalous instances, and $\varepsilon \in [0, 1)$ is the proportion of contamination. Letting S_0 and S_1 be the support of P_0 and P_1, respectively, we further assume that $S_0 \cap S_1 = \emptyset$. The distributions P_0 and P_1, their support and the level of contamination ε are unknown.

p-NN radius

Definition 3.1 (p-NN radius)**.** *Let $p \in (0,1)$. For any x, define $r_p(x)$ to be the radius of the smallest ball centered at x with P-probability mass at least p. Formally,*

$$r_p(x) = \inf\{r > 0 : P(B(x,r)) \geq p\}.$$

Naturally, the empirical p-NN radius is defined as

$$\hat{r}_p(x) = \inf\{r > 0 : P_n(B(x,r)) \geq p\},$$

where P_n is the empirical measure that puts mass $1/n$ on each X_i. Setting $k = \lceil np \rceil$, $\hat{r}_p(x)$ is simply the kth-nearest neighbor radius of the point x with respect to the sample $(X_1, \ldots, X_n)$. Thus,

$$P_n(B(x, \hat{r}_p(x)) = \frac{1}{n} \left|\{X_1, \ldots, X_n\} \cap B(x, \hat{r}_p(x))\right| = \frac{k}{n}.$$

- **Assumption (A0):**
  The sets S_0 and S_1 have diameters bounded by some $L > 0$, and are disjoint from each other.
- **Assumption (A1):**
  There exist positive constants $C = C(P)$ and $\nu_0 = \nu_0(P)$ such that for all $0 < \nu < \nu_0$ and $\gamma \in \mathbb{R}$,
  $$|P(B(x, r_p(x) + \gamma)) - P(B(x, r_p(x)))| \leq \nu \Rightarrow |\gamma| < C\nu,$$
  for P-almost every x.
- **Assumption (A2):**
  P_0 satisfies the **(a,b)-condition**: For $b > 0$, for any $x \in S_0$, there exist $a = a(x) > 0$, and $r > 0$ such that $P_0(B(x, r)) \geq \min\{1, ar^b\}$.

Definition 3.2 (DTM [18]). *The distance-to-a-measure (DTM) with respect to a probability distribution P with parameter $m \in (0,1)$ and power $q \geq 1$ is defined as*

$$d(x) = d_{P,m,q}(x) = \left(\frac{1}{m} \int_0^m r_p(x)^q \, dp \right)^{1/q}. \tag{1}$$

When $q = \infty$, we set $d(x) = d_{P,m,\infty}(x) = r_m(x)$.

By substituting $r_p(x)$ with $\hat{r}_p(x)$ in (1), the empirical DTM can be seen to be

$$\hat{d}(x) = d_{P_n,m,q}(x) = \left(\frac{1}{k} \sum_{X_i \in N_k(x)} \|X_i - x\|^q \right)^{1/q},$$

Uniform bounds on p-NN radius

Theorem 3.3. *Let* $\delta \in (0,1)$ *and set* $\beta_n = \sqrt{(4/n)((d+1)\log 2n + \log(8/\delta))}$. *Under assumption (A1), with probability at least* $1-\delta$ *we have that*

$$\sup_x |\hat{r}_p(x) - r_p(x)| \leq C(\beta_n^2 + \beta_n\sqrt{p}),$$

where C *is the constant introduced in Assumption (A1), simultaneously over all* $p \in (0,1)$ *such that*

$$p + \beta_n^2 + \beta_n\sqrt{p} \leq 1 \quad \textit{and} \quad p - \beta_n^2 - \beta_n\sqrt{p} \geq 0. \tag{2}$$

Uniform bounds on p-NN radius at sample points

Theorem 3.4. *Let* $\delta \in (0,1)$ *and set* $\alpha_n = \sqrt{(4/(n-1))(\log 2(n-1) + \log(8n/\delta))}$. *Under assumption (A1), with probability at least* $1-\delta$ *we have that*

$$\max_{i=1,\ldots,n} |\hat{r}_p(X_i) - r_p(X_i)| \leq C(\alpha_n^2 + \alpha_n\sqrt{p} + \frac{2}{n})$$

where C *is the constant introduced in Assumption (A1), simultaneously over all* $p \in (0,1)$ *such that*

$$p + \alpha_n^2 + \alpha_n\sqrt{p} \leq 1 \quad \textit{and} \quad p - 2/n - \alpha_n^2 - \alpha_n\sqrt{p - 2/n} \geq 0. \tag{3}$$

Uniform bounds DTM

Theorem 3.5. *Under assumption (A0) and (A1), with probability at least* $1-\delta$,

$$\sup_x |d(x) - \hat{d}(x)| \leq C_1 \beta_n (\beta_n + \sqrt{m}), \tag{4}$$

and

$$\max_{i=1,\ldots,n} |d(X_i) - \hat{d}(X_i)| \leq C_2 \alpha_n (\alpha_n + \sqrt{m} + \frac{2}{n}). \tag{5}$$

where β_n *and* α_n *are defined in Theorem 3.3 and Theorem 3.4 and* C_1 *and* C_2 *are some positive constants depending on* q, *the diameter bound* L *in Assumption (A0) and the constant* C *in Assumption (A1).*

The methodology we consider is quite simple, and it is consistent with the prevailing practice of assigning to each sample point a score that expresses its degree of being anomalous compared to the other points. In detail, we rank the sample points based on their empirical DTM values, and we declare the points with largest empirical DTM values as anomalies. This simple procedure will work perfectly well if

$$\max_{X_i \in S_0} \hat{d}(X_i) < \min_{X_i \in S_1} \hat{d}(X_i)$$

and if the difference between the two quantities is large. In general, of course, one would expect that some sample points in S_0 may have smaller empirical DTMs of some of the points in S_1. The extent to which such incorrect labeling occurs depends on two key factors: how closely the empirical DTM tracks the true DTM and whether the population DTM could itself discriminate normal points versus anomalous ones. The former issue can be handled using the high probability bounds on the stochastic fluctuations of the empirical DTM obtained in the previous section. The latter issue will instead require to specify some degree of separation between the mixture components P_0 and P_1, both in terms of the distance between their supports but also in terms of how their probability mass gets distributed. There is more than one way to formalize this setting. Here we choose to remain

Proposition 3.6. *Under assumptions (A0) and (A2), suppose that* $a(x) = g(d(x, \partial S_0))$, *where* $g(z)$ *is a non-decreasing function on* $[0, z_0)$ *for some* z_0, *and* $g(z) \geq g(z_0)$ *for all* $z \geq z_0$. *Let*

$$\eta = \min_{x \in S_0, y \in S_1} \|x - y\| \tag{6}$$

be the distance between S_0 *and* S_1 *and* $h > 0$ *be a given threshold parameter. For any* $m > \varepsilon$, *additionally assume that*

$$g(z_0) \geq g_0 := \begin{cases} \frac{m}{1-\varepsilon} \left(\frac{b+q}{b} \left(\frac{m-\varepsilon}{m} \eta^q - h \right) \right)^{-b/q} & 1 \leq q < \infty \\ \frac{m}{1-\varepsilon} (\eta - h)^{-b} & q = \infty. \end{cases} \tag{7}$$

Next, define the "safety zone" A_η *as*

$$A_\eta = \left\{ x \in S_0 : d(x, \partial S_0) \geq g^{-1}(g_0) \right\} \tag{8}$$

Then, we have

$$\sup_{x \in A_\eta} d_{P,m,q}(x) + h < \inf_{y \in S_1} d_{P,m,q}(y). \tag{9}$$

The main message from the previous result is that there exists a subset A_η of the support of the normal distribution, which intuitively corresponds to a region deep inside the support of P_0 of high density, over which the population DTM will be smaller than at any point in the support S_1 of the contaminating distribution. Thus, the true DTM is guaranteed to perfectly separate A_η from S_1, making mistakes (possibly) only for the normal points in $S_0 \setminus A_\eta$.

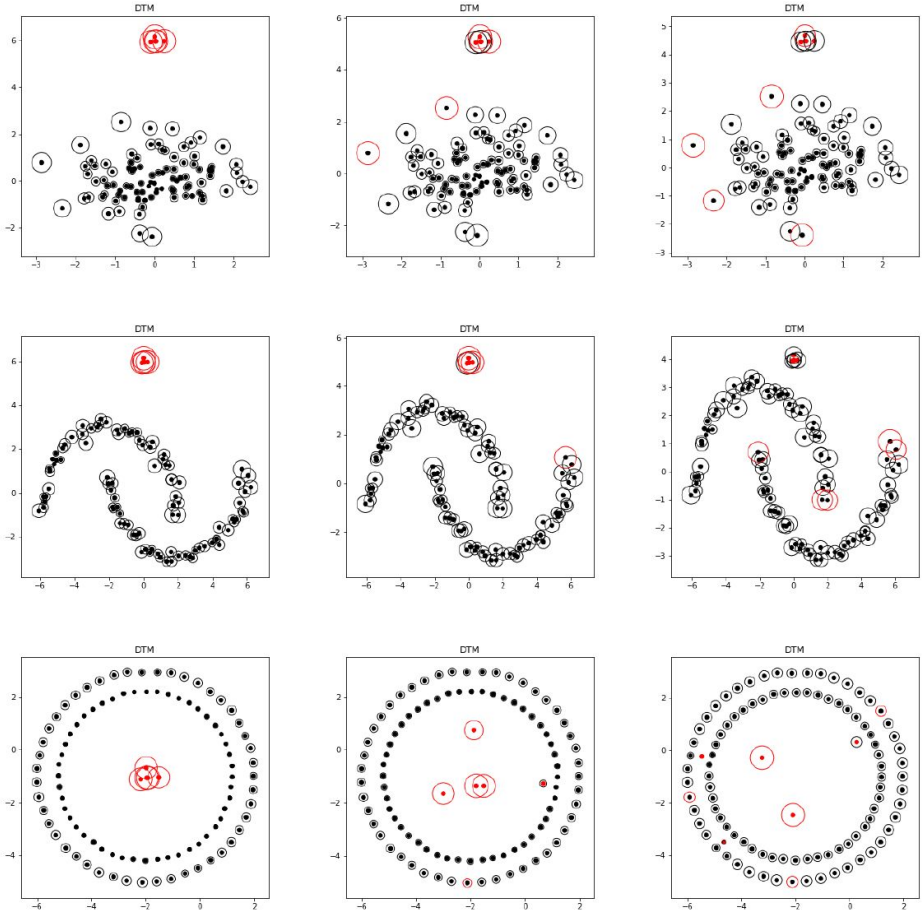
Main Theorem: empirical DTM

Corollary 3.6.1. *Taking h to be twice the upper bound in (5), we get, with probability at least $1-\delta$,*

$$\max_{X_i \in A_\eta} \hat{d}_{P,m,q}(X_i) < \min_{X_i \in S_1} \hat{d}_{P,m,q}(X_i).$$

Similarly, if h is twice the upper bound in (4), we have that

$$\sup_{x \in A_\eta} \hat{d}_{P,m,q}(x) < \inf_{y \in S_1} \hat{d}_{P,m,q}(y). \tag{10}$$



High Separation Medium Separation Low Separation

Questions?

Greedy Sampling for Approximate Clustering in the Presence of Outliers

Aditya Bhaskara
University of Utah
bhaskaraaditya@gmail.com

Sharvaree Vadgama
University of Utah
sharvaree.vadgama@gmail.com

Hong Xu
University of Utah
hxu.hongxu@gmail.com

NIPS 2019

Abstract

Greedy algorithms such as adaptive sampling (k-means++) and furthest point traversal are popular choices for clustering problems. One the one hand, they possess good theoretical approximation guarantees, and on the other, they are fast and easy to implement. However, one main issue with these algorithms is the sensitivity to noise/outliers in the data. In this work we show that for k-means and k-center clustering, simple modifications to the well-studied greedy algorithms result in nearly identical guarantees, while additionally being robust to outliers. For instance, in the case of k-means++, we show that a simple thresholding operation on the distances suffices to obtain an $O(\log k)$ approximation to the objective. We obtain similar results for the simpler k-center problem. Finally, we show experimentally that our algorithms are easy to implement and scale well. We also measure their ability to identify noisy points added to a dataset.

Issue

k-center

k-means++

The algorithms of Gonzalez (also known as *furthest point traversal*) and [4] are appealing also due to their simplicity and efficiency. However, one main drawback in these algorithms is their sensitivity to corruptions/outliers in the data. Imagine $10k$ of the points of a dataset are corrupted and the coordinates take large values. Then both furthest point traversal as well as k-means++ end up choosing *only* the outliers. The goal of our work is to remedy this problem, and achieve the simplicity and scalability of these algorithms, while also being robust in a provable sense.

Goal: robust clustering

Formulating clustering with outliers. Let $\textsc{Opt}_{\text{full}}(X)$ denote the k-center or k-means objective on a set of points X. Now, given a set of points that also includes outliers, the goal in clustering with outliers (see [7, 17, 22]) is to partition the points X into X_{in} and X_{out} so as to minimize $\textsc{Opt}_{\text{full}}(X_{\text{in}})$. To avoid the trivial case of setting $X_{\text{in}} = \emptyset$, we rquire $|X_{\text{out}}| \leq z$, for some parameter z that is also given. Thus, we define the optimum $\textsc{Opt}$ of the k-clustering with outliers problem as

$$\textsc{Opt} := \min_{|X_{\text{out}}| \leq z} \textsc{Opt}_{\text{full}}(X \setminus X_{\text{out}}).$$

Recent results

The recent result of [22] (and the earlier result of [10] for k-median) go beyond bi-criteria approximation. They prove that for k-means clustering, one can obtain a factor 50 approximation to the value of OPT, while declaring at most z points as outliers, as desired. While this effectively settles the complexity of the problem, there are many key drawbacks. First, the algorithm is based on an iterative procedure that solves a linear programming relaxation in each step, which can be very inefficient in practice (and hard to implement). Further, in many applications, it may be necessary to improve on the (factor 50) approximation guarantee, potentially at the cost of choosing more clusters or slightly weakening the bound on the number of outliers.

Approximation of OPT

Definition 1. *Consider an algorithm for the k-clustering (means/center) problem that on input X, k, z, outputs k' centers (allowed to be slightly more than k), along with a partition $X = X'_{in} \cup X'_{out}$ that satisfies (a) $|X'_{out}| \leq \beta z$, and (b) the objective value of assigning the points X'_{in} to the output centers is at most $\alpha \cdot \mathrm{OPT}$.*

Then we say that the algorithm obtains an (α, β) approximation using k' centers, for the k-clustering problem with outliers.

k-center

K-center clustering in metric spaces. For k-center, our algorithm is a variant of furthest point traversal, in which instead of selecting the furthest point from the current set of centers, we choose a *random point* that is not too far from the current set. Our results are the following.

Theorem 1.1. *Let $z, k, \varepsilon > 0$ be given parameters, and $X = X_{in} \cup X_{out}$ be a set of points in a metric space with $|X_{out}| \leq z$. There is an efficient randomized algorithm that with probability $\geq 3/4$ outputs a $(2 + \varepsilon, 4 \log k)$-approximation using precisely k centers to the k-center with outliers problem.*

K-center: with additional clusters

Theorem 1.2. *Let $z, k, c, \varepsilon > 0$ be given parameters, and $X = X_{in} \cup X_{out}$ be a set of points in a metric space with $|X_{out}| \leq z$. There is an efficient randomized algorithm that with probability $\geq 3/4$ outputs a $(2+\varepsilon, (1+c)/c)$-approximation using $(1+c)k$ centers to the k-center w/ outliers problem.*

Algorithm 1 k-center with outliers

Input: points $X \subseteq \mathbb{R}^d$, parameters k, z, r; r is a guess for OPT
Output: a set $S_\ell \subseteq X$ of size ℓ

1: Initialize $S_0 = \emptyset$
2: **for** $t = 1$ to ℓ **do**
3: Let $\mathcal{F}_t$ be the set of all points that are at a distance $> 2r$ from S_{t-1}. I.e.,

$$\mathcal{F}_t := \{x \in X : d(x, S_{t-1}) > 2r\}$$

4: Let x be a point sampled u.a.r from $\mathcal{F}_t$
5: $S_t = S_{t-1} \cup \{x\}$
6: **return** S_ℓ

S_t안의 outlier개수

멀리 있는 inlier개수

$$\Psi_t := \frac{w_t |\mathcal{F}_t \cap X_{\text{in}}|}{n_t}.$$

S_t와 겹치지 않는 optimal cluster수

Along with the observation that $\Psi_0 = 0$ (since $w_0 = 0$), we have

$$\mathbb{E}[\Psi_k] = \sum_{t=0}^{k-1} \mathbb{E}[\Psi_{t+1} - \Psi_t] \leq \sum_{t=0}^{k-1} \frac{z}{k-t} \leq zH_k,$$

where H_k is the kth Harmonic number. Thus by Markov's inequality, $\Pr[\Psi_k \leq 4zH_k] \geq 3/4$. By the definition of Ψ_k, this means that with probability at least $3/4$,

$$\frac{w_k |\mathcal{F}_t \cap X_{\text{in}}|}{n_k} \leq 4z \ln k.$$

The key observation is that we always have $w_k = n_k$. This is because if the set S_k did not intersect n_k

k-means

K-means clustering. Here, our main contribution is to study an algorithm called T-kmeans++, a variant of D^2 sampling (i.e. k-means++), in which the distances are thresholded appropriately before probabilities are computed. For this simple variant, we will establish robust guarantees that nearly match the guarantees known for k-means++ without any outliers.

Theorem 1.3. *Let z, k, β be given parameters, and $X = X_{in} \cup X_{out}$ be a set of points in Euclidean space with $|X_{out}| \leq z$. There is an efficient randomized algorithm that with probability $\geq 3/4$ gives an $(O(\log k), O(\log k))$-approximation using k centers to the k-means with outliers problem on X.*

Theorem 1.4. *Let z, k, β, c be given parameters, and $X = X_{in} \cup X_{out}$ be a set of points in a metric space with $|X_{out}| \leq z$. Let $\delta > 0$ be an arbitrary constant. There is an efficient randomized algorithm that with probability $\geq 3/4$ gives a $((\beta+64), (1+c)(1+\beta)/c(1-\delta))$-approximation using $(1+c)k$ centers to the k-center with outliers problem on X.*

Now, for any set of centers C, we define

$$\tau(x, C) = \min\left(d(x, C)^2, \frac{\beta \cdot \textsc{Opt}}{z}\right) \tag{3}$$

Algorithm 2 Thresholded Adaptive Sampling – T-kmeans++

Input: a set of points $X \subseteq \mathbb{R}^d$, parameters k, z, and a guess for the optimum $\textsc{Opt}$.
Output: a set $S \subseteq X$ of size ℓ.

1: Initialize $S_0 = \emptyset$.
2: **for** $t = 1 \dots \ell$ **do**
3: sample a point x from the distribution

$$p(x) = \frac{\tau(x, S_{t-1})}{\sum_{x \in X} \tau(x, S_{t-1})}. \qquad \text{(with } \tau \text{ as defined in (3))}$$

4: set $S_t = S_{t-1} \cup \{x\}$.
5: **return** S_ℓ

key-observation

Lemma 2. *Let C be a set of centers, and suppose that $\tau(X, C) \leq \alpha \cdot \text{OPT}$. Then we can partition X into X'_{in} and X'_{out} such that*

1. $\sum_{x \in X'_{in}} d(x, C)^2 \leq \alpha \cdot \text{OPT}$, *and*
2. $|X'_{out}| \leq \frac{\alpha z}{\beta}$.

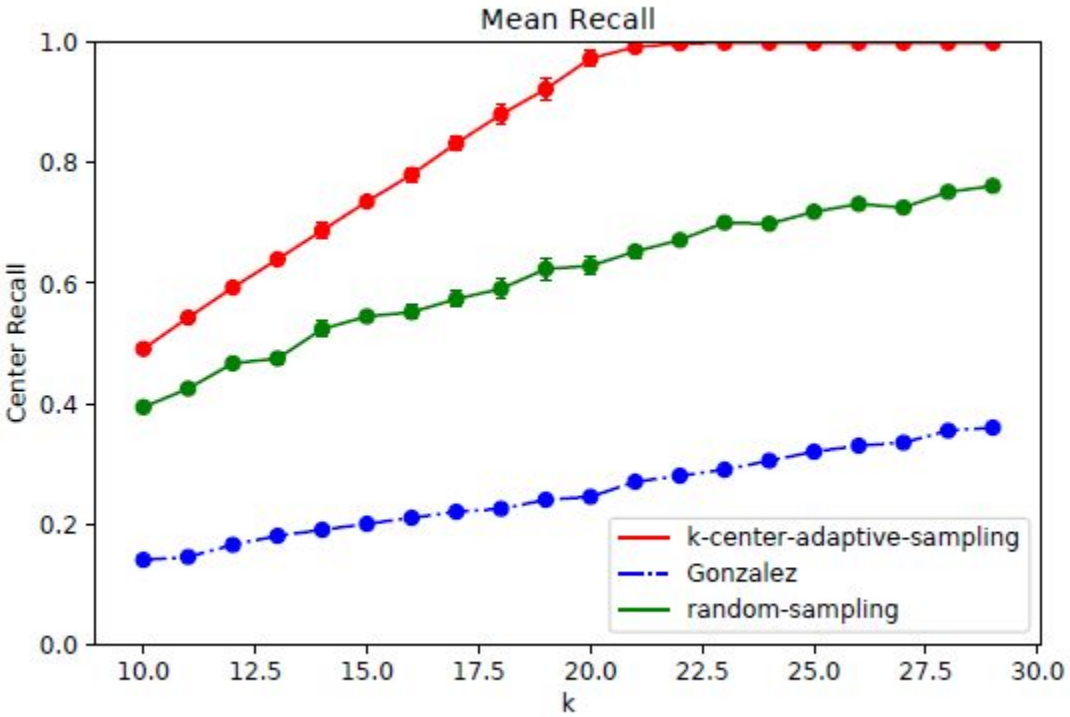
Theorem 3.1. *Running algorithm 2 for k iterations outputs a set S_k that satisfies*

$$\mathbb{E}[\tau(X, S_k)] \leq (\beta + O(1)) \log k \cdot \textsc{Opt}.$$

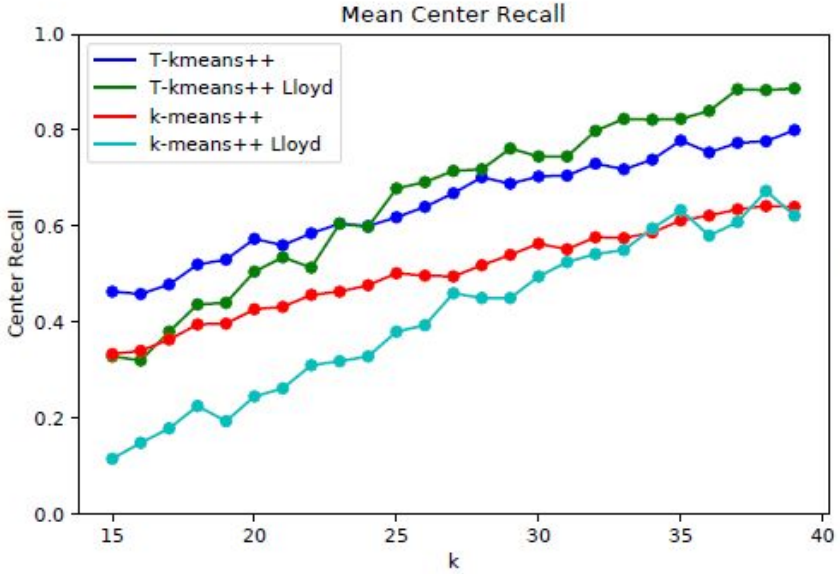
Theorem 3.2. *Consider running Algorithm 2 for $\ell = (1 + c)k$ iterations, where $c > 0$ is a constant. Then for any $\delta > 0$, with probability $\geq \delta$, the set S_ℓ satisfies*

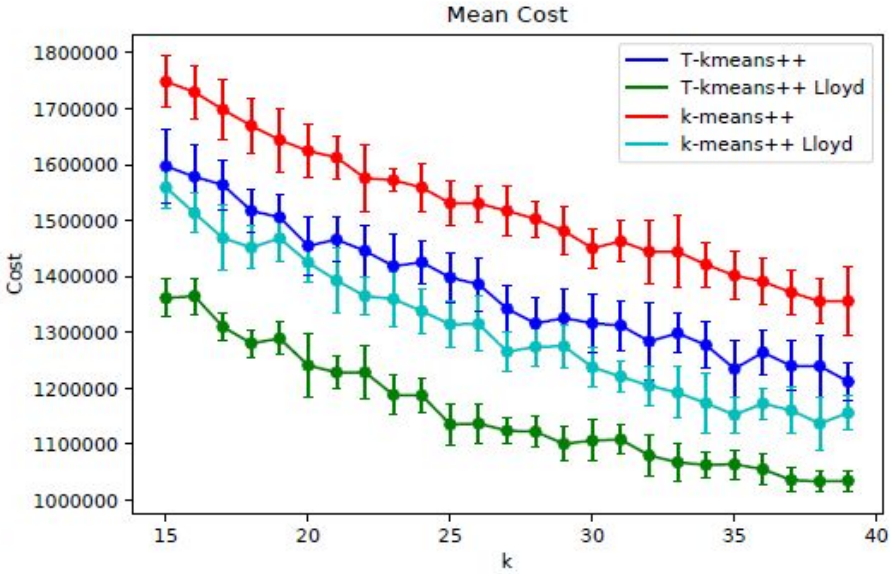
$$\tau(X, S_\ell) \leq \frac{(\beta + 64)(1 + c)\textsc{Opt}}{(1 - \delta)c}.$$

K-center experiment (Synthetic data)



K-means experiment (Synthetic data)





k-means with outliers. Once again, we demonstrate the *cluster recall* on a synthetic dataset. In this case, we compare our algorithm with a heuristic proposed in [17]: running k-means++ followed by an iteration of "outlier-senstive Lloyd's iteration", proposed in [8]. We also ran the latter procedure as a post-processing step for our algorithm. Figure 2 reports the cluster recall and the value of the k-means objective for the algorithms. Unlike the case of k-center, the T-kmeans++ algorithm can potentially choose points in one cluster multiple times. However, we consistently observe that T-kmeans++ outperforms the other heuristics.

Finally, we perform experiments on three datasets:

1. NIPS (a dataset from the conference NIPS over 1987-2015): clustering was done on the rows of a 11463×50 matrix (the number of columns was reduced via SVD).
2. The MNIST digit-recognition dataset: clustering was performed on the rows of a 60000×40 (again, SVD was used to reduce the number of columns).
3. Skin Dataset (available via the UCI database): clustering was performed on the rows of a $245,057 \times 3$ matrix (original dataset).

In order to simulate corruptions, we randomly choose 2.5% of the points in the datasets and corrupt all the coordinates by adding independent noise in a pre-defined range. The following table outlines the results. We report the *outlier recall*, i.e., the number of true outliers designated as outliers by the algorithm. For fair comparison, we make all the algorithms output precisely z outliers. Our results indicate slightly better recall values for T-kmeans++. For some data sets (e.g. Skin), the k-means objective value is worse for T-kmeans++. Thus in this case, the outliers are not "sufficiently corrupting" the original clustering.[1]

Dataset	k	KM recall	TKM recall	KM objective	TKM objective
NIPS	10	0.960	0.977	4173211	4167724
	20	0.939	0.973	4046443	4112852
	30	0.924	0.978	3956768	4115889
Skin	10	0.619	0.667	7726552	7439527
	20	0.642	0.690	5936156	5637427
	30	0.630	0.690	5164635	4853001
MNIST	10	0.985	0.988	1.546×10^8	1.513×10^8
	20	0.982	0.989	1.475×10^8	1.449×10^8
	30	0.977	0.986	1.429×10^8	1.412×10^8

Table showing outlier recall for KM (k-means++) and TKM (T-kmeans++) along with the k-means cost.

Questions?

The Cells Out of Sample (COOS) dataset and benchmarks for measuring out-of-sample generalization of image classifiers

Alex X. Lu
Computer Science,
University of Toronto
alexlu@cs.toronto.edu

Amy X. Lu
Computer Science,
University of Toronto
Vector Institute
amyxlu@cs.toronto.edu

Wiebke Schormann
Biological Sciences,
Sunnybrook Research Institute
wiebke.schormann@sri.utoronto.ca

Marzyeh Ghassemi
CIFAR AI Chair,
University of Toronto
Vector Institute
marzyeh@cs.toronto.edu

David W. Andrews
Biological Sciences,
Sunnybrook Research Institute
Biochemistry and Medical Biophysics,
University of Toronto
david.andrews@sri.utoronto.ca

Alan M. Moses
Cell and Systems Biology
Computer Science
CAGEF
University of Toronto
alan.moses@utoronto.ca

NIPS 2019

Abstract

Understanding if classifiers generalize to out-of-sample datasets is a central problem in machine learning. Microscopy images provide a standardized way to measure the generalization capacity of image classifiers, as we can image the same classes of objects under increasingly divergent, but controlled factors of variation. We created a public dataset of 132,209 images of mouse cells, COOS-7 (**C**ells **O**ut **O**f **S**ample **7**-Class). COOS-7 provides a classification setting where four test datasets have increasing degrees of covariate shift: some images are random subsets of the training data, while others are from experiments reproduced months later and imaged by different instruments. We benchmarked a range of classification models using different representations, including transferred neural network features, end-to-end classification with a supervised deep CNN, and features from a self-supervised CNN. While most classifiers perform well on test datasets similar to the training dataset, all classifiers failed to generalize their performance to datasets with greater covariate shifts. These baselines highlight the challenges of covariate shifts in image data, and establish metrics for improving the generalization capacity of image classifiers.

Challenge

are drawn from the same distribution. In practice, even small natural variations in data distributions can challenge the generalization capacity of classifiers: Recht *et al*. show that deep learning models trained on CIFAR or ImageNet drop in classification accuracy when evaluated on a new dataset carefully curated with the same methods as the original datasets [1], suggesting that even state-of-the-art classifiers are not robust to out-of-sample data from a more realistic setting.

Biomedical domain

While understanding the robustness of classification models to covariate shifts (situations where the distribution of out-of-sample data differs from that of training data) is broadly applicable, biomedical domains exemplify cases where the failure of image classifiers to generalize can have serious consequences. Diagnostic systems, such as those that predict pneumonia from chest radiographs, may not generalize to data from different institutions [2]. In pharmaceutical research, drugs are screened based on the effects they have on diseased cells [3]; models classifying these effects perform better on microscope images from the same sample than on reproduced experiments [4]. These issues are not exclusive to images, and are prevalent in many biomedical datasets: similar challenges include batch effects in genomics data [5], site effects in MRI data [6], and covariate shifts over time in medical records [7, 8]. Thus, validating models with realistic out-of-sample datasets is important not only for estimating performance in real use-cases where covariate shifts are unavoidable, but also for model selection: performance gains on randomly held-out test data may not translate to improvements on datasets with covariate shifts [7].

Sensitivity

Here, we sought to create a standardized dataset for measuring the robustness of image classifiers under various degrees of covariate shift. We reasoned that microscopy experiments would allow us to image a large set of naturally variable objects (cells) under controlled factors of variation. Cells naturally vary in aspects like shape or size [9]. While still stochastic, these variations are influenced by environmental factors like temperature or humidity [10], meaning that images taken on the same day are more likely to be similar than those taken on different days or seasons. Compounding these biological variations are technical biases, such as microscope settings. Different instruments may produce subtle illumination or contrast differences, which classifiers can overfit [11, 12].

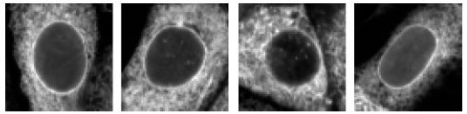
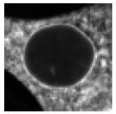
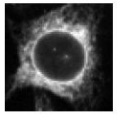
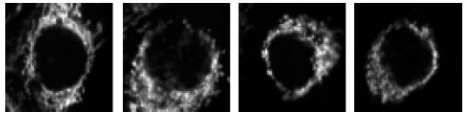
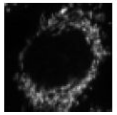
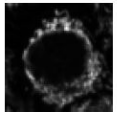
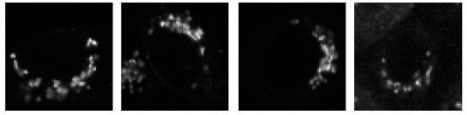
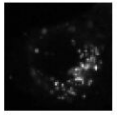
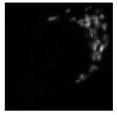
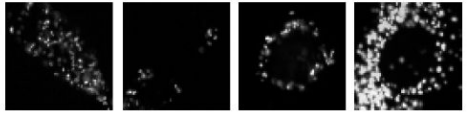
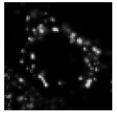
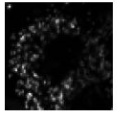
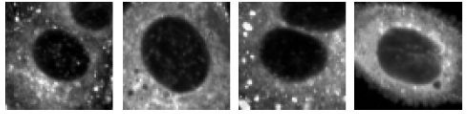
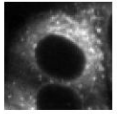
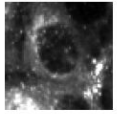
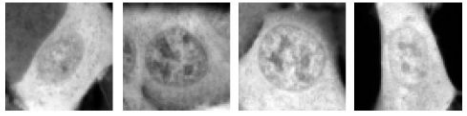
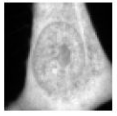
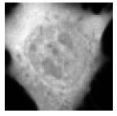
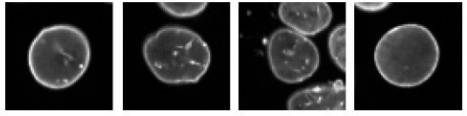
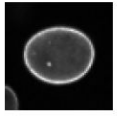
2 COOS-7

2.1 Overview of images and classification setting

To create COOS-7, we curated 132,209 images of mouse cells. Each image in COOS-7 is a 64x64 pixel crop centered around a unique mouse cell. Each image contains two channels. The first channel shows a fluorescent protein that targets a specific component of the cell. Each mouse cell is stained with one of seven fluorescent proteins, which highlight distinct parts of the cell ranging from the ER to the nuclear membrane. The goal of our classification problem is to predict which fluorescent protein a cell has been stained with: Table 1 summarizes our class labels and shows example images of the first channel for each class, from three of the datasets in COOS-7.

The second channel is a fluorescent dye that stains the nucleus, consistent across all cells in our dataset. On its own, the second nucleus channel is not expected to discriminate any of the classes in our dataset, but we provide this channel to help models learn useful correlations. For example, while the Golgi (class 3) and the peroxisomes (class 5) are both characterized as bright dots in the cell, the Golgi tends to surround the nucleus, while the peroxisomes are distributed more evenly in the cell.

Table 1: Classes and examples from COOS-7

Label	Class	Training Examples	Test3 Example	Test4 Example
0	Endoplasmic Reticulum (ER)			
1	Inner Mitochondrial Membrane (IMM)			
2	Golgi			
3	Peroxisomes			
4	Early Endosome			
5	Cytosol			
6	Nuclear Envelope			

Data setup

COOS-7 is curated from a larger set of microscopy experiments, spanning the course of two years. In these experiments, cells were grown on plates, containing 384 wells. Each well is a fluorescent protein, with the configuration differing from plate to plate. A robot-controlled microscope slides over the wells, taking 10-20 images for each well (see methods of [13] for a similar experimental set-up.) The original images taken by the microscope typically contain multiple cells; we process these into crops centered around individual cells by segmenting the second nucleus channel using a trained mask-RCNN, YeastSpotter [14]. We systematically imaged 95 plates, typically with about a week between plates. Four of these plates were grown and imaged at a different microscope in a collaborating institution. All images were taken using PerkinElmer OPERA® QHS spinning disk automated confocal microscope with 40x water objectives (NA=0.9).

Table 2: Description of datasets provided in COOS-7

Dataset	Description	Images
Training	Images from 4 independent plates for each class	41,456
Test1	Randomly held-out images from the same plates in training dataset	10,364
Test2	Images from the same plates, but different wells than training dataset	17,021
Test3	Images from 2 independent plates for each class, reproduced on different days than training dataset	32,596
Test4	Images from 1 plate for each class, reproduced on different day and imaged under different microscope than training dataset	30,772

Classifiers 1, 2

First, as our classic computer vision baseline, we extracted Haralick texture features (abbreviated as Texture), which are popular for microscopy image analysis due to their rotation invariance [21].

Second, we trained a fully supervised 11-layer CNN, DeepLoc, which has achieved state-of-the-art results in classifying protein localization for 64x64 images of yeast cells [22]. We followed all preprocessing, parameterization, and model selection practices by the authors. We trained DeepLoc for 10,000 iterations with a batch size of 128 on 80% of the Training dataset, and chose the iteration (at intervals of 500) with the best performance on the remaining 20% (stratified to preserve percentage of samples per class). We report end-to-end performance, and we also extracted features from the last fully-connected layer of our trained model for building further classifiers.

Classifiers 3, 4

Third, we extracted features from pretrained CNNs on ImageNet (abbreviated as VGG16), which has been shown to outperform classic unsupervised feature representation methods for cancer cell morphology [23]. We used a pre-trained VGG16 model, as this was the best-performing previously-reported model that would accept 64x64 images. We converted and rescaled the first channel of our images to 8-bit RGB. Contrary to the results of Pawlowski *et al.*, we observed that including features from the second channel decreased performance, so we did not include models with these features in our final baselines. We extracted features from all layers (max-pooling convolutional layers), but only report benchmarks for the top 3 overall performing layers.

Fourth, we extracted features learned from a self-supervised method designed for microscopy images (abbreviated as PCI), which has achieved unsupervised state-of-the-art results in classifying protein localization for 64x64 images of yeast cells and human cells [24]. We trained the model unsupervised on the Training dataset exclusively, following practices by the authors. We extracted features from all layers of the source cell encoder of this model (max-pooling convolutional layers), but only report benchmarks for the top 3 overall performing layers.

3.2 All classifiers drop in performance on out-of-sample data with larger covariate shifts

All methods we tried performed well on the test datasets most similar to the Training dataset, Test1 and Test2. Features from our deep learning models had as little as 1.1% error on these datasets, but even logistic regression classifiers built on classic computer vision features achieved 6.8% error or lower. However, when attempting to generalize to the test datasets with larger covariate shifts, Test3 and Test4, all classifiers had large drops in performance (although the fully-supervised CNN achieved the lowest error on these datasets.)

Table 3: Class-Balanced Error (%) of Classification Models on COOS-7 Datasets

Features	Model	Train	Test1	Test2	Test3	Test4
DeepLoc	End-to-End	1.2	1.2	1.5	7.4	5.4
DeepLoc (FC2)	kNN	1.1	1.3	1.5	6.9	4.7
DeepLoc (FC2)	L1 LR	1.1	**1.1**	**1.4**	7.7	**4.1**
DeepLoc (FC2)	RF	0.0	**1.1**	**1.4**	**6.8**	5.0
Texture	kNN	10.4	11.8	11.2	17.6	25.6
Texture	L1 LR	6.4	**6.8**	**6.5**	**12.0**	**12.1**
Texture	RF	0.0	7.3	7.1	16.4	17.1
PCI Conv3	kNN	2.2	2.4	2.7	8.9	8.0
PCI Conv3	L1 LR	1.0	**1.4**	**1.7**	9.2	7.4
PCI Conv3	RF	0.1	2.1	2.2	11.0	8.6
PCI Conv4	kNN	2.1	2.4	2.5	10.1	7.8
PCI Conv4	L1 LR	1.6	1.5	1.9	**8.7**	6.0
PCI Conv4	RF	0.1	2.5	2.7	10.7	7.5
PCI Conv5	kNN	2.6	2.7	2.9	12.1	8.9
PCI Conv5	L1 LR	2.5	2.5	2.6	11.4	**5.7**
PCI Conv5	RF	0.0	2.5	2.7	10.8	7.4

Features	Model	Train	Test1	Test2	Test3	Test4
VGG16 Conv3_3	kNN	6.8	7.9	8.2	12.4	10.0
VGG16 Conv3_3	L1 LR	5.7	6.6	6.9	11.3	9.3
VGG16 Conv3_3	RF	0.2	9.5	9.2	15.9	10.9
VGG16 Conv4_1	kNN	6.5	7.3	7.6	9.3	8.4
VGG16 Conv4_1	L1 LR	3.1	4.2	4.1	8.2	**6.7**
VGG16 Conv4_1	RF	0.1	7.5	7.3	11.3	7.8
VGG16 Conv4_2	kNN	6.6	7.8	7.8	9.1	8.4
VGG16 Conv4_2	L1 LR	2.8	**3.9**	**3.9**	**8.0**	6.8
VGG16 Conv4_2	RF	0.2	7.4	7.5	10.2	8.4

3.3 Confusion matrices reveal non-uniform errors

Next, to understand which specific classes our classifiers were failing to generalize on, we examined the confusion matrices for some classifiers on various test datasets (Supplementary Tables 1-9). We observed that covariate shifts sometimes have non-uniform effects on classification performance: in some cases, the majority of classes were predicted with very little error, with only a few classes sharply decreasing in performance.

Across the classifiers we examined, we observed that the two most common errors were classifying the early endosome as the ER or the Golgi, or the Golgi as the IMM or the peroxisomes. These errors were between the more visually similar classes in COOS-7; in contrast, the classes that were distinct from any other class in our dataset, such as the cytosol or the nuclear envelope, were generally classified well by all classifiers, across all datasets.

As an example of a case where errors were predominantly concentrated in one class, we observed for the DeepLoc classifiers on Test 4, while every other class was classified with > 0.97 sensitivity, the early endosome class was classified with only 0.684 sensitivity, compared to 0.989 sensitivity in Test1.

The non-uniform effects of covariate shifts observed here suggest that overall metrics may not always adequately describe how classifiers fail to generalize on out-of-sample data. Here, these effects are detectable due to the small number of classes, but for classification problems with many classes (such as ImageNet), a large drop in performance on only a few classes may not be detectable from metrics like the overall classification error.

We also observed that some models were robust to errors in the same classes in one dataset, but not in another. For example, the logistic regression classifier built on the VGG16 features had lower sensitivity on the Golgi class in both Test3 (0.848) and Test4 (0.866). In contrast, the logistic regression classifier built on the self-supervised (PCI) features had lower sensitivity in Test3 (0.749), but not in Test4 (0.984).

These results suggest that the exact nature of covariate shifts can differ in different out-of-sample datasets. New out-of-sample datasets may challenge classifiers in unpredictable ways, inducing errors not seen in previous out-of-sample datasets. Thus, validation on a single out-of-sample dataset may not be sufficient to conclude that a classification model is robust in general.

Questions?