Clustered Hierarchical Anomaly and Outlier Detection Algorithms

Thomas J. Howard III*
thoward27@uri.edu
Computer Science and Statistics
University of Rhode Island
Kingston, RI

Najib Ishaq*
najib_ishaq@zoho.com
Computer Science and Statistics
University of Rhode Island
Kingston, RI

Noah M. Daniels
noah_daniels@uri.edu
Computer Science and Statistics
University of Rhode Island
Kingston, RI

ABSTRACT

In this paper we use CHESS for introspective anomaly detection. Using the underlying clustering mechanism from CHESS, along with some manually supplied stopping criteria, we present strong evidence that CHESS is able to accurately map the underlying manifold of the data and that we are able to use this manifold and its inherent properties to successfully flag anomalous data. We empirically demonstrate the effectiveness of our method using several synthetic data sets. Finally, we explore some future opportunities that could greatly increase the scope of this work.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Anomaly detection**.

KEYWORDS

Anomaly Detection, Outlier Detection, Novelty Detection, Manifold Learning

ACM Reference Format:
Thomas J. Howard III, Najib Ishaq, and Noah M. Daniels. 2020. Clustered Hierarchical Anomaly and Outlier Detection Algorithms. In *KDD '20*. ACM, New York, NY, USA, 8 pages. https://doi.org/X

1 INTRODUCTION

Detecting anomalies or outliers from a distribution of data is a well-studied problem in machine learning. When data occupy easily-described distributions, such as Gaussian, the task is relatively easy, requiring only that one identify that a datum is sufficiently far from the mean or median. However, in "big data" scenarios, data occupy high-dimensional spaces that do not behave intuitively. Furthermore, data may obey the "manifold hypothesis" [5], occupying a low-dimensional manifold in the higher-dimensional embedding space. Detecting anomalies or outliers in such a landscape is not so easy; in particular, correctly identifying an anomalous datum that sits between "branches" of a tree-like manifold presents a challenge.

*These authors contributed equally to this research.

KDD '20, X, X

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/X

Anomalies (data which don't belong to a particular distribution) and outliers (data which represent extrema of a distribution) can arise from many sources: errors in measurement or data collection; novel, previously-unseen instances of data; normal behavior evolving into abnormal; or adversarial attacks as seen in examples of malicious inputs to machine-learning systems [**?**]

1.1 Related Works

In this section, we consider previous works in distance-based, clustering-based, and graph-based approaches in outlier/anomaly detection.

1.2 Clustering-Based Approaches

Clustering data is universally centered around techniques for grouping instances in a way that provides value. This is generally done by assigning *similar* points to the same cluster. Once clusters are formed, they can be used to inspect the data in great detail. For example, for any given point, one could estimate it's anomalousness by it's distance to it's nearest cluster. To determine if a cluster contains outliers, one could examine the relative cardinality of the cluster.

We summarize several major approaches to clustering data. Note that there have been relatively few advancements in clustering techniques over the past decade [15]. This may explain, or by attributable to, the observably poor performance, thus far, of clustering data in higher dimensional space [17].

1.2.1 Distance-Based Clustering. Distance-based clustering approaches rely on some distance measure to partition the data into some number of clusters. Within this approach, the numbers of clusters and/or the sizes of clusters are normally predefined: either user-specified, or chosen at random [15]. Some examples of distance-based clustering are: K-Means [10], PAM [9], CLARANS [12], CLARA [9], etc.

1.2.2 Hierarchical Clustering. Hierarchical clustering methods typically utilize a tree-like structure, where points are allocated into leaf nodes [15]. These tree-like structures can be created bottom-up (agglomerative clustering), or top-down (divisive clustering) [13]. The major drawback of these methods is the high cost of pairwise difference computations at each level of the tree. Examples of hierarchical clustering include: MST [3], CURE [1], CHAMELEON [8], etc.

1.2.3 Density-Based Clustering. Density-Based clustering methods rely on finding areas of high point-density separated by areas of low point-density. These algorithms generally do not work well

when normal data are sparse or uniformly distributed. Some examples of density-based clustering algorithms are: DBSCAN [4], DENCLUE [6], etc.

1.2.4 Grid-Based Clustering. Grid-based clustering works via segmenting the entire space into a finite number of cells and then iterating over the cells to find regions of higher density. Utilizing the grid structure for clustering means that these algorithms have typically scaled better to larger datasets. Some examples of grid-based clustering: STING [16], Wavecluster [14], DCluster, CLIQUE [13], etc.

1.3 Distanced-Based Approaches

Distance-based methods attempt to find anomalous points via distance comparisons. These methods largely employ k-Nearest Neighbors as their substrate [15]. They proceed in the following, slightly different, ways:

(1) Points with less than p other points within a distance d are outliers.
(2) The top n points whose distances to their k^{th}-nearest neighbor are the greatest are outliers.
(3) The top n points whose average distances to their k^{th}-nearest neighbor are the greatest are outliers.

We introduce a novel technique, Clustered Learning of Approximate Manifolds (CLAM). This approach uses a divisive hierarchical clustering approach to learn a manifold in a Banach space [] defined by a distance metric. In actuality, our requirements are less strict; the space can be defined by a distance *function* that does not obey the triangle inequality, though this is not always optimal. Given a learned manifold, we can almost trivially implement several anomaly-detection algorithms; in this manuscript, we present a collection of five such algorithms implemented on CLAM: CHAODA (Clustered Hiearchical Anomaly and Outlier Detection Algorithms).

The manifold learning component derives from prior work aimed at accelerating the task of approximate search on large data sets of high dimension, CHESS [7]. CLAM begins by divisively clustering the data until every point is within its own cluster, as a singleton. CLAM then delineates *layers* of clusters at each depth in the tree. Each layer comprises all clusters that would have been leaf nodes if the tree building were to have been halted at the given depth.

CLAM then build a graph for each layer in the tree by creating edges between clusters that have overlapping volumes. This process effectively learns the manifold on which the data lie at various resolutions, given by the depth of the layer. This is analogous to a "filtration" in computational topology [2]. Once we have learned a manifold, one can ask about the cardinality of various clusters at different depths, how connected a given cluster is, or even how often a cluster is visited over many different random walks across the manifold.

We test our methods on 26 real-world datasets. Each dataset contains a different number of anomalous data, for a different domain. We consider several different definitions for outliers and anomalies: **distance-based**, examining several classical distance-based definitions of outliers, relying on CLAM's use of distance to cluster data; **density-based**, examining the cardinality of clusters, under the hypothesis that clusters with lower cardinality are more likely to contain outliers; **graph-based**, examining several graph-theoretic methods for anomaly detection, given graphs constructed from layers of clusters.

Historically, clustering approaches have suffered from one several problems. The most common deficiencies are: the effective treatment of high dimensionality, interpretability of results, or scalability to exponentially-growing datasets [13]. With CLAM, we have largely alleviated these problems while also decoupling search time from the size of the data being searched.

2 METHODOLOGY

For this work we have applied a multitude of different Distance-Based, Density-Based, and Graph-Based, approaches to detect anomalous and outlying points/clusters. In this section, we detail the specific algorithms employed from these various categories.

2.1 Distance-Based Outliers

Algorithm 1: Cluster Cardinality

```
Data: manifold, depth
Result: scores
scores ← {}
d ← depth
forall cluster c ∈ manifold.graph(d).clusters do
    forall p ∈ c.points do
        scores[p] ← |c|
    end
end
normalize(scores)
forall p ∈ scores do
    scores[p] ← 1 − p
end
```

Algorithm 2: Hierarchical

```
Data: manifold,depth
Result: scores
scores ← {}
d ← depth
for d ∈ 1 ··· depth do
    forall cluster c ∈manifold.graph(d).clusters do
        forall point p ∈ c do
            scores[p] ← |c.parent| / (|c|×d)
        end
    end
end
normalize(scores)
```

(1) Sort all points $p \in \mathbb{P}$, where $\mathbb{P}$ is the data, by $f \equiv |B_D(p, r)|$ in ascending order.
    - If needed, increase r to break ties.
    - The points with the smallest values of f are the outliers.

Algorithm 3: k-Neighborhood

```
Data: manifold,depth,k
Result: scores
scores ← {}
d ← depth
g ← manifold.graph(d).clusters
forall cluster c ∈ g do
    v ← |{c′ ∈ g|δ(c, c′) ≤ k}|
    forall p ∈ c do
        scores[p] ← v
    end
end
normalize(scores)
forall p ∈ scores do
    scores[p] ← 1 − p
end
```

Algorithm 4: Random Walk

```
Data: manifold,depth
Result: scores
scores ← {}
d ← depth
g ← manifold.graph(d).components
forall component k ∈ g do
    visits ← outrank(k)
    forall cluster c ∈ k do
        v ←visits[c]
        forall p ∈ c do
            scores[p] ← v
        end
    end
end
normalize(scores)
forall p ∈ scores do
    scores[p] ← 1 − p
end
```

(2) Sort all points $p \in \mathbb{P}$ in descending order by the distance to their k^{th} nearest neighbor.
- Consider how this distance changes as k in increased.
- The points with the highest such distances are the outliers.

2.2 Hierarchical-Clustering-Based Outliers

(1) For a *parent* cluster and its $left$ and *right* child-clusters (assume without loss of generality that $|left| \leq |right|$) define $f \equiv \frac{|left|}{|parent|}$. If $f \ll 0.5$ then points in the $|left|$ cluster are outliers.
- Use this definition recursively down the tree. The number of levels in the tree at which a point is labelled an outlier gives us a measure of the "anomalousness" of that point.

Algorithm 5: Subgraph Cardinality

```
Data: manifold, depth
Result: scores
scores ← {}
d ← depth
forall component k ∈ manifold.graph(d).components do
    forall cluster c ∈ k do
        forall p ∈ c do
            scores[p] ← |k|
        end
    end
end
normalize(scores)
forall p ∈ scores do
    scores[p] ← 1 − p
end
```

2.3 Graph-Based Outliers

(1) The Outrank algorithm [11] i.e. on a given graph-representation of data, initiate a random walk. The clusters that are visited least often are the outlier clusters.
(2) Define the "k-neighborhood" of a clusters c as the clusters that can be reached from c in k steps. Investigate how $|k - neighborhood|$ increases as k in increased.
- $|k - neighborhood|$ should increase by k^d where d is *local fractal dimension* of the data at the length-scale of the radii of the clusters that form the graph.
- If the increase in $|k - neighborhood|$ of a cluster c does not keep pace with k^d as k is increased then c can be considered an outlier cluster.

(3) Consider the connected components of the graph at a depth in the tree just before the graph shatters into many small components or isolated clusters.
- If a component contains too few points/clusters then those points/clusters can be considered outliers.
- If a small component is connected to a larger component with a small number of edges then the smaller component may contain outliers.

Due to the wide range of possible measurements for "anomalousness" from our methods, we normalize our measurements.

2.4 Normalization Methods

(1) Min-Max Scaling:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

(2) Mean-Scaling:

$$x' = \frac{x - \overline{x}}{x_{max} - x_{min}} \tag{2}$$

(3) z-Score Standardization:

$$x' = \frac{x - \overline{x}}{\sigma} \tag{3}$$

3 RESULTS

We have implemented and tested our methods on 26 different real-world datasets. The vast majority of datasets studied were sourced from Outlier Detection Datasets (ODDS), which provided clear labels for normal and anomalous behavior. We examine the effectiveness of our methods by their ability to identify these outliers.

Figure 1 shows We chose to use so many different real world datasets to ensure that our methods could generalize to many target audiences.

For each dataset, we clustered using Manhattan, Euclidean, and Cosine distance functions. Additionally, for this work, we allowed CLAM to cluster down until the manifold had thoroughly shattered. We then iterated over all depths to compute the area under the ROC-Curve.

After computing anomalousness scores, we normalized each metric to a $[0, 1]$ range. For each dataset, we present a confusion matrix, a ROC curve, and the area under the ROC curve for a sample of optimal depths. To demonstrate our methods are not overly-sensitive to depth we present all metrics in a 5-depth wide window.

Table 1 lists

Figures 2, 3, 5, and 6 shows histograms of measured anomalousness for points in each dataset for each method used. Higher values of anomalousness indicate higher confidence that the point is an anomaly.

4 DISCUSSION

We have presented CHAODA (Clustered Hierarchical Anomaly and Outlier Detection Algorithms), a collection of five algorithms that share the property of exploiting properties of a hierarchical cluster tree which learns a manifold in potentially high-dimensional space. All five algorithms are trivial to implement on top of a manifold-learning framework we call CLAM (Clustered Learning of Approximate Manifolds); CHAODA builds on this framework just like CHESS [7], which learns a manifold in the same way, but for the purpose of accelerating approximate search. In CHESS, the geometric and topological properties of low fractal dimension and low metric entropy are advantages; indeed, CHESS does not perform particularly well if those properties are not present. CHAODA, on the other hand, while competitive with other state-of-the-art anomaly-detection approaches on "easy" data sets (we define as *easy* any data set where a one-class SVM performs well), outperforms other current methods when the data exhibit precisely those properties that CHESS depends on for acceleration.

Some discussion about where CHAODA doesn't perform as well...

One current limitation in CHAODA is that the depth of the cluster tree at which anomaly detection performs best is not the same for every data set, and thus our results could be seen as "cherry-picking" from a scattershot approach. Future work should certainly explore optimal stopping criteria so that this process can be further automated. However, one can clearly observe that the choice of depth is robust to minor deviations, and one can treat depth as a hyperparameter to the methods described.

The strong correlation between local fractal dimension and optimal tree depth suggests a guideline for determining an optimal tree depth directly from the data.

The choice of distance function can also have a significant impact on anomaly-detection performance. In this case, domain knowledge is likely the best way to determine the distance function of choice. In future work, we seek to explore a more diverse collection of domain-appropriate distance functions, such as Wasserstein distance on images, or Levenshtein edit distance on strings.

Say something about applying CHAODA for inputs to an ANN, in particular detecting just-off-manifold malicious inputs, like the school bus / ostrich example.

REFERENCES

[1] [n.d.]. CURE: An Efficient Clustering Algorithm for Large Databases. ([n. d.]), 12.

[2] Gunnar Carlsson. 2009. Topology and data. *Bull. Amer. Math. Soc.* 46, 2 (2009), 255–308.

[3] Charles Zahn. [n.d.]. Graph Theoretic Methods for Detecting and Describing Gestalt Clusters. C-20, 1 ([n. d.]). https://www.cs.bgu.ac.il/~icbv161/wiki.files/Readings/1971-Zahn-Graph_Theoretic_Methods_for_Detecting_and_Describing_Gestalt_Clusters.pdf

[4] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. [n.d.]. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. ([n. d.]), 6.

[5] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. 2016. Testing the manifold hypothesis. *Journal of the American Mathematical Society* 29, 4 (2016), 983–1049.

[6] Alexander Hinneburg and Daniel A. Keim. 1998. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In *Proceedings of the 4 th International Conference on Knowledge Discovery and Datamining (KDD ' 98), New York, NY, September, 1998.* 58–65.

[7] Najib Ishaq, George Student, and Noah M. Daniels. 2019. Clustered Entropy-Scaling Search of Astronomical and Biological Data. In *2019 IEEE International Conference on Big Data (Big Data).* IEEE, 780–789.

[8] George Karypis and Vipin Kumar. [n.d.]. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. ([n. d.]), 22.

[9] Leonard Kaufman and Peter J Rousseeuw. [n.d.]. Finding Groups in Data, An Introduction to Cluster Analysis. ([n. d.]), 15.

[10] J Macqueen. [n.d.]. SOME METHODS FOR CLASSIFICATION AND ANALYSIS OF MULTIVARIATE OBSERVATIONS. ([n. d.]), 17.

[11] H. D. K. Moonesinghe and Pang-Ning Tan. [n.d.]. OutRank: A GRAPH-BASED OUTLIER DETECTION FRAMEWORK USING RANDOM WALK. 17, 1 ([n. d.]), 19–36. https://doi.org/10.1142/S0218213008003753

[12] Raymond T Ng and Jiawei Han. [n.d.]. Efficient and Effective Clustering Methods for Spatial Data Mining. ([n. d.]), 12.

[13] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. [n.d.]. Automatic subspace clustering of high dimensional data for data mining applications. 27 ([n. d.]), 94–105. https://doi.org/10.1145/276304.276314

[14] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. 2000. WaveCluster: A Wavelet-based Clustering Approach for Spatial Data in Very Large Databases. *The VLDB Journal* 8, 3-4 (Feb. 2000), 289–304. https://doi.org/10.1007/s007780050009

[15] Hongzhi Wang, Mohamed Jaward Bah, and Mohamed Hammad. [n.d.]. Progress in Outlier Detection Techniques: A Survey. 7 ([n. d.]), 107964–108000. https://doi.org/10.1109/ACCESS.2019.2932769

[16] Wei Wang, Jiong Yang, and Richard R. Muntz. 1997. STING: A Statistical Information Grid Approach to Spatial Data Mining. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB '97).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 186–195. http://dl.acm.org/citation.cfm?id=645923.758369

[17] Ji Zhang. [n.d.]. Advancements of Outlier Detection: A Survey. 13, 1 ([n. d.]), e2. https://doi.org/10.4108/trans.sis.2013.01-03.e2

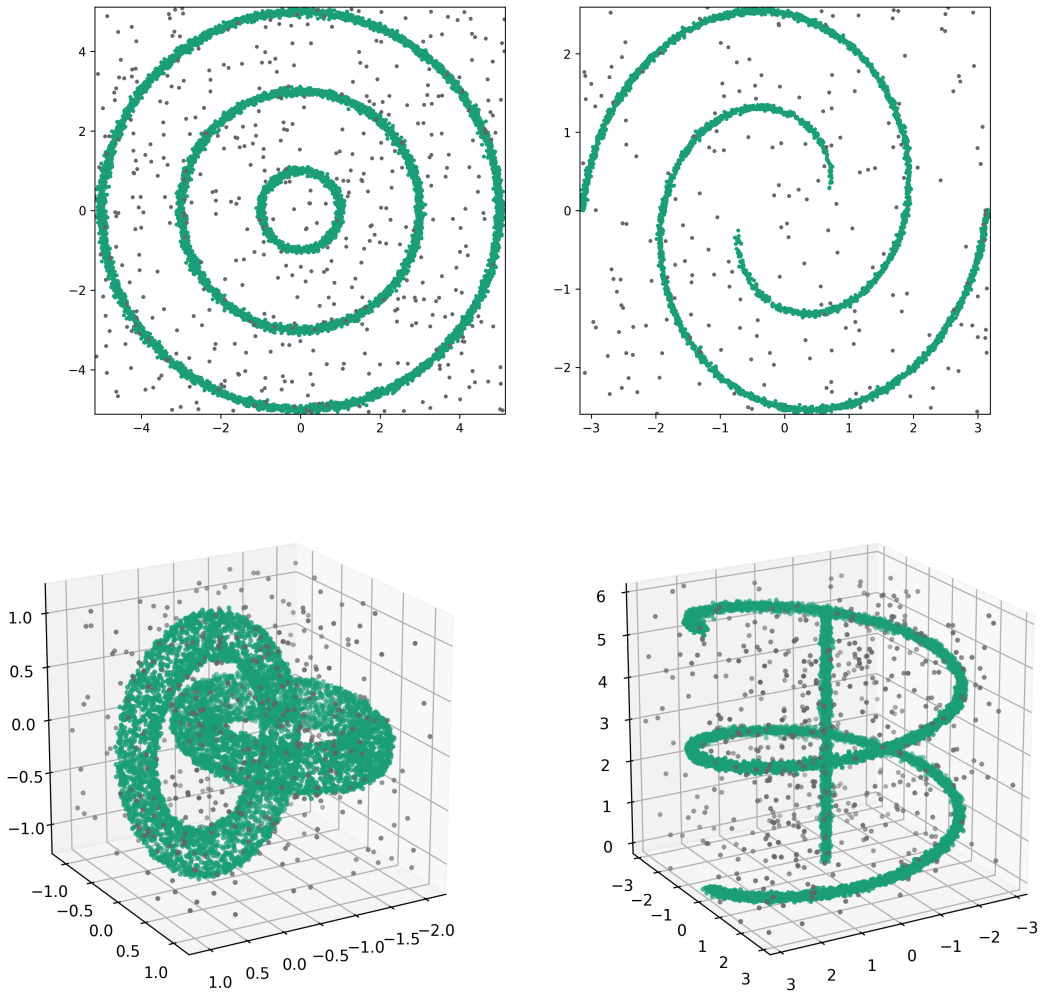

Figure 1: The Bullseye (top-left), Spiral (top-right), Interlocking Tori (bottom-left) and Skewer (bottom-left) datasets.

Table 1: Anomaly Detection performance on the synthetic datasets.

	Bullseye		**Spiral**		**Interlocking Tori**		**Skewer**	
	TPR	**TNR**	**TPR**	**TNR**	**TPR**	**TNR**	**TPR**	**TNR**
kth-nearest Distances	1.000	0.979	1.000	0.020	1.000	0.979	1.000	0.979
Hierarchical Sparseness	0.985	0.940	1.000	0.969	0.992	0.961	1.000	0.967
Outrank Algorithm	0.998	0.990	1.000	0.987	1.000	0.989	1.000	0.989
k-Neighborhood Size	0.996	0.990	0.990	0.984	0.761	0.986	0.975	0.983

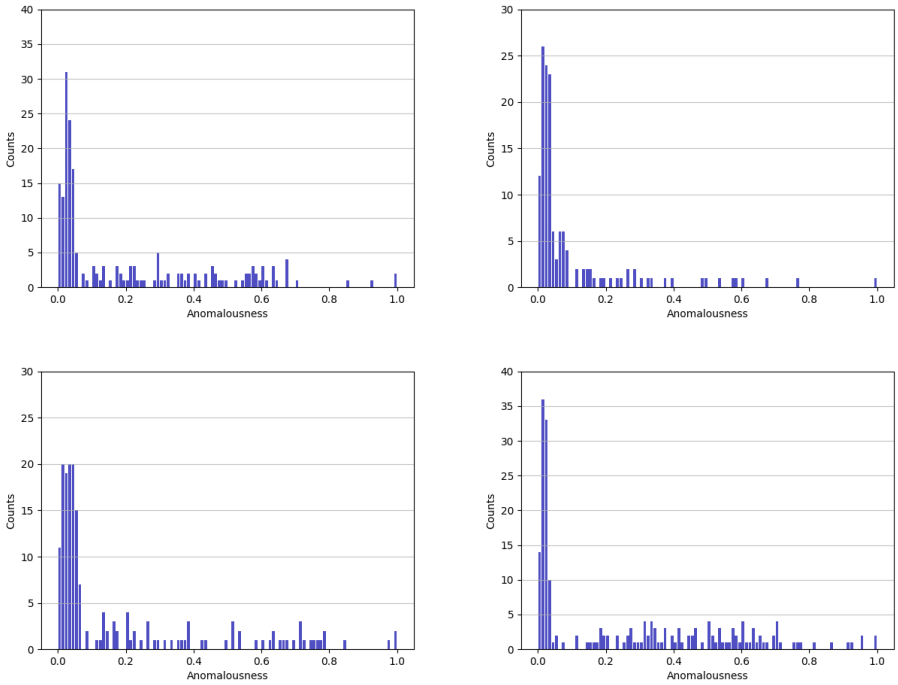

Figure 2: Measures of anomalousness using the kth-nearest Distances method on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.

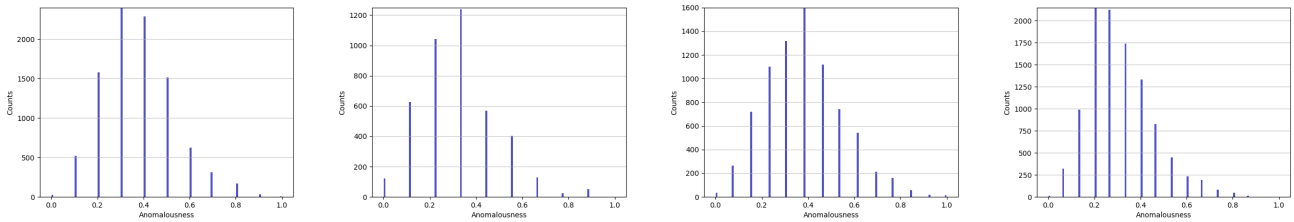

Figure 3: Measures of anomalousness using the Hierarchical Sparseness method on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.

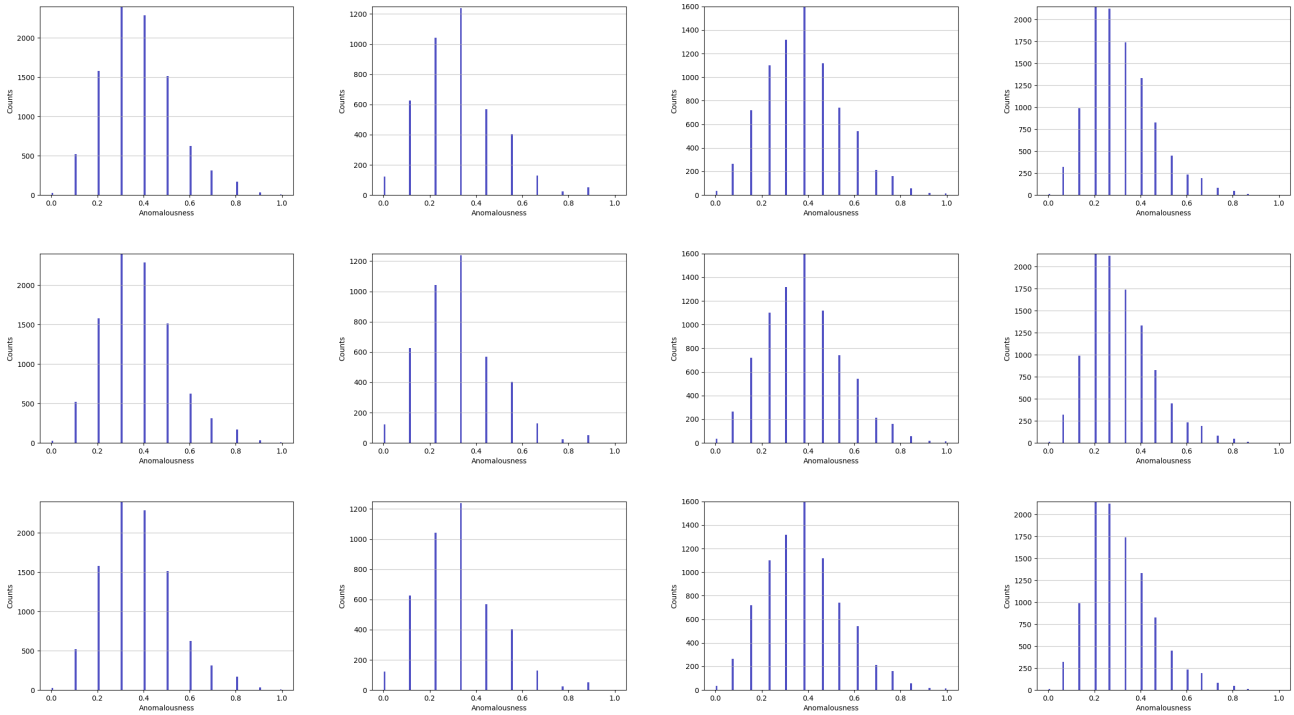

Figure 4: Example of a grid of regular-sized images.

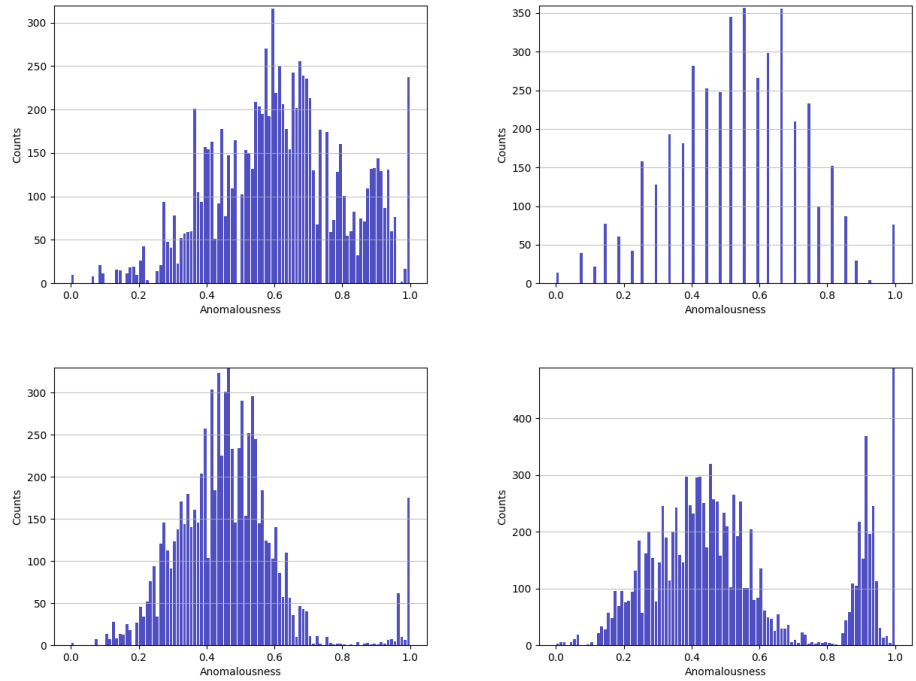

Figure 5: Measures of anomalousness using the Outrank Algorithm on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.

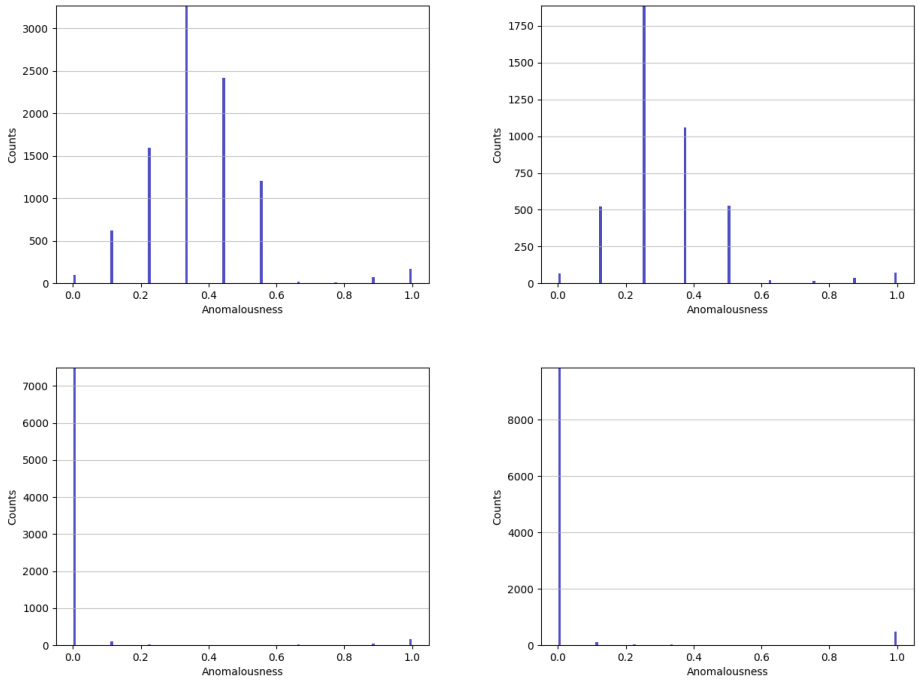

Figure 6: Measures of anomalousness using the k-Neighborhood Size method on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.