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 median or mean. However, in "big data" scenarios, where data can occupy high-dimensional spaces, anomalous behavior becomes harder to quantify. If the data happens to be uniformly distributed throughout even a high dimensional space one can conceive of simple mechanisms that would be effective, such as a Single-Class SVM, yet data rarely exhibits such patterns. Instead, data often obey the "manifold hypothesis" [7], occupying a low-dimensional manifold in the higher-dimensional embedding space. This low-dimensional manifold may weave itself through the higher dimensional space much like a crumpled piece of paper can appear to be a sphere from a distance. Detecting anomalies or outliers in such a landscape is not so easy; in particular, correctly identifying an anomalous datum that sits within the gaps of a manifold presents a challenge.

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 [5]. Current systems designed to detect anomalous behavior systematically fail for a wide variety of reasons. While some systems perform well with large quantities of data, such as grid-based approaches, they falter as dimensionality increases. Others may succeed with a larger number of dimensions, but perform too slowly to be truly useful.

1.1 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 [17]. This may explain, or by attributable to, the observably poor performance, thus far, of clustering data in higher dimensional space [19].

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 [17]. Some examples of distance-based clustering are: K-Means [13], PAM [12], CLARANS [15] and CLARA [12].

Hierarchical clustering methods typically utilize a tree-like structure, where points are allocated into leaf nodes [17]. These tree-like structures can be created bottom-up (agglomerative clustering), or top-down (divisive clustering) [1]. The major drawback of these methods is the high cost of pairwise difference computations at

*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

each level of the tree. Examples of hierarchical clustering include MST [4], CURE [8] and CHAMELEON [11].

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 [6] and DENCLUE [9].

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 include STING [18], Wavecluster [16], and CLIQUE [1].

1.2 Distanced-Based Approaches

Distance-based methods attempt to find anomalous points via distance comparisons. These methods largely employ k-Nearest Neighbors as their substrate [17]. Each method uses one of the following approaches: points with fewer than p other points within some distance d are outliers; the n points with the greatest distances to their $k^{th}-$nearest neighbor are outliers; or the n points with the greatest average distance to their k nearest neighbors are outliers.

1.3 CHAODA

In this paper 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 [2] 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 approximate 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 Hierarchical 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 [10]. 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 [3]. 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, the ability to interpret results, and the ability to scale to exponentially-growing datasets [1]. With CLAM, we have largely alleviated these problems.

2 METHODOLOGY

Given the manifold and induced graph learned by the CLAM approach described in Section 1.3, we have implemented five distinct algorithms for detecting anomalies or outliers, all of which take advantage of either the tree of hierarchical clusters or the graph learned from those clusters at a given depth.

2.1 Cluster Cardinality

For this method we equate anomalousness with the cardinality of the cluster to which a point belongs. If a point belongs to a cluster with a very small number of points, it is more likely that the point is an anomaly or an outlier. Clusters that have a large cardinality are treated as more likely to be normal. Algorithm 1 details the cluster cardinality approach.

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
```

2.2 Hierarchical

In this approach, we examine the ratio of the cardinality of a cluster and that of its parent cluster. For a given depth of the tree, a point is considered to be more anomalous if the cluster it belongs to at that depth is significantly smaller than the cluster it belongs to one level higher in the tree. This relationship tries to capture the intution that the point lives away from the manifold inhabited by most of the data. This approach is detailed in Algorithm 2.

2.3 k-Neighborhood

For this approach, we rely on the graph learned by CLAM at a particular depth of the tree, as explained in Section 1.3. Given that graph, and a point in question, we consider the number of clusters

Algorithm 2: Hierarchical

Data: manifold,depth
Result: scores
$scores \leftarrow \{\}$
$d \leftarrow depth$
for $d \in 1 \cdots depth$ **do**
  forall *cluster* $c \in manifold.graph(d).clusters$ **do**
    forall *point* $p \in c$ **do**
      $scores[p] \leftarrow \frac{|c.parent|}{|c| \times d}$
    end
  end
end
normalize(scores)

reachable within a graph distance of k from the point's cluster. The more clusters which are reachable from a given cluster, the *less* likely it is to contain anomalous points. This approach is detailed in Algorithm 3

Algorithm 3: k-Neighborhood

Data: manifold,depth,k
Result: scores
$scores \leftarrow \{\}$
$d \leftarrow$ depth
$g \leftarrow$ manifold.graph(d).clusters
forall *cluster* $c \in g$ **do**
  $v \leftarrow |\{c' \in g | \delta(c, c') \leq k\}|$
  forall $p \in c$ **do**
    scores[p] $\leftarrow v$
  end
end
normalize(scores)
forall $p \in scores$ **do**
  $scores[p] \leftarrow 1 - p$
end

2.4 Random Walk

Here, we implement the Outrank [14] algorithm, with a key difference: since the graph learned by CLAM at a given depth of the tree may contain more than one connected component, some parts of the graph may be completely unreachable from a point in question. The general idea behind this approach is to examine the frequency with which clusters are visited during n random walks. Clusters that are visited more frequently, are again less likely to be anomalous. Clusters that are not visited frequently are more likely to be disconnected, and distant from the underlying manifold. This approach is detailed in Algorithm 4, which assumes that $outrank(k)$ returns a hash table whose keys are the clusters of the graph, and whose values are the number of visits by a random walk to each cluster.

Algorithm 4: Random Walk

Data: manifold,depth
Result: scores
$scores \leftarrow \{\}$
$d \leftarrow$ depth
$g \leftarrow$ manifold.graph(d).components
forall *component* $k \in g$ **do**
  visits $\leftarrow$ outrank(k)
  forall *cluster* $c \in k$ **do**
    $v \leftarrow$visits[c]
    forall $p \in c$ **do**
      scores[p] $\leftarrow v$
    end
  end
end
normalize(scores)
forall $p \in scores$ **do**
  $scores[p] \leftarrow 1 - p$
end

2.5 Subgraph Cardinality

This approach is similar to Cluster Cardinality. Here, we utilize the cardinality of each connected component of the graph at various depths to determine anomalousness. Similarly to some of our other approaches, we postulate that clusters which are members of sparse components are themselves more likely to be anomalous. Algorithm 5 details this approach.

Algorithm 5: Subgraph Cardinality

Data: manifold, depth
Result: scores
$scores \leftarrow \{\}$
$d \leftarrow depth$
forall *component* $k \in manifold.graph(d).components$ **do**
  forall *cluster* $c \in k$ **do**
    forall $p \in c$ **do**
      scores[p] $\leftarrow |k|$
    end
  end
end
normalize(scores)
forall $p \in scores$ **do**
  $scores[p] \leftarrow 1 - p$
end

2.6 Normalization Methods

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

(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 [10], 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] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. 1998. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*. 94–105.

[2] Stefan Banach. 1929. Sur les fonctionnelles linéaires II. *Studia Mathematica* 1 (1929), 223–239.

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

[4] 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

[5] Gamaleldin Elsayed, Shreya Shankar, Brian Cheung, Nicolas Papernot, Alexey Kurakin, Ian Goodfellow, and Jascha Sohl-Dickstein. 2018. Adversarial examples that fool both computer vision and time-limited humans. In *Advances in Neural Information Processing Systems*. 3910–3920.

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *Kdd*, Vol. 96. 226–231.

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

[8] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. 1998. CURE: an efficient clustering algorithm for large databases. *ACM Sigmod record* 27, 2 (1998), 73–84.

[9] Alexander Hinneburg, Daniel A Keim, et al. 1998. An efficient approach to clustering in large multimedia databases with noise. In *KDD*, Vol. 98. 58–65.

[10] 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.

[11] George Karypis, Eui-Hong Han, and V Kumar Chameleon. 1999. A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer* 32, 8 (1999), 68–75.

[12] Leonard Kaufman and Peter J Rousseeuw. 2009. *Finding groups in data: an introduction to cluster analysis*. Vol. 344. John Wiley & Sons.

[13] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA, 281–297.

[14] HDK Moonesinghe and Pang-Ning Tan. 2008. Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools* 17, 01 (2008), 19–36.

[15] RT Ng and J Han. 1994. :" Efficient and Effective Clustering Methods for Spatial Data Mining", Proc. 20th Int. Conf. On Very Large Data Bases, Santiago, Chile, Morgan Kaufmann Publishers. (1994).

[16] 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 (2000), 289–304.

[17] Hongzhi Wang, Mohamed Jaward Bah, and Mohamed Hammad. 2019. Progress in Outlier Detection Techniques: A Survey. *IEEE Access* 7 (2019), 107964–108000.

[18] Wei Wang, Jiong Yang, Richard Muntz, et al. 1997. STING: A statistical information grid approach to spatial data mining. In *VLDB*, Vol. 97. 186–195.

[19] Ji Zhang. 2013. Advancements of outlier detection: A survey. *ICST Transactions on Scalable Information Systems* 13, 1 (2013), 1–26.

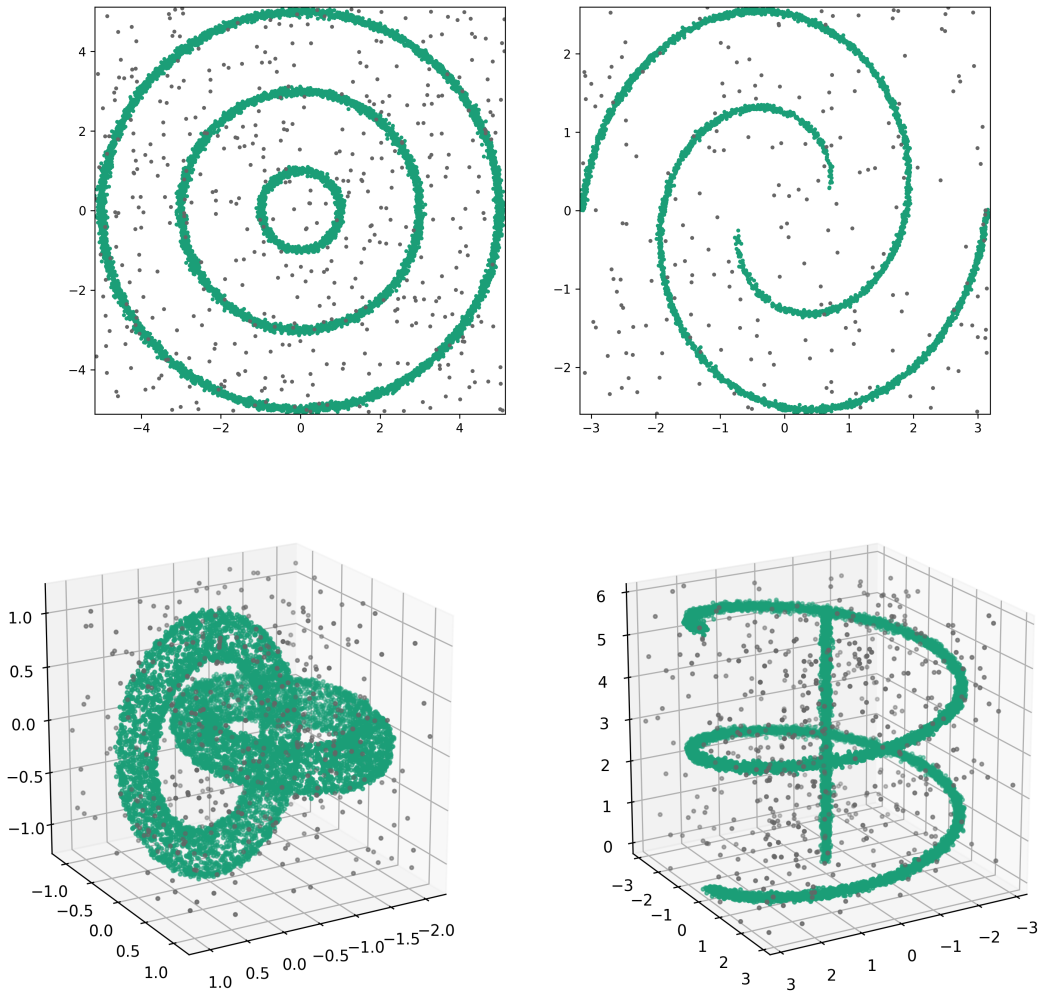

Figure 1: The Bullseye (top-left), Spiral (top-right), Interlocking Tori (bottom-left) and Skewer (bottom-left) datasets.

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

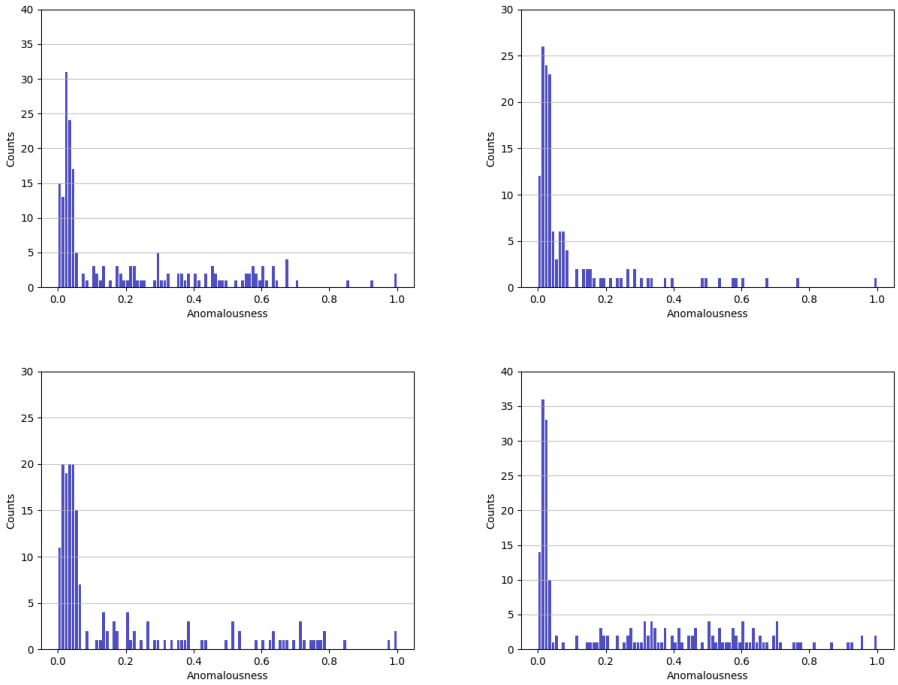

Figure 2: Measures of anomalousness using the kth-nearest Distances method on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.

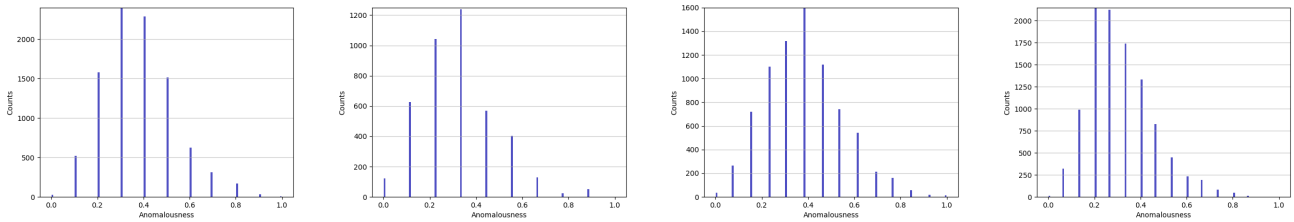

Figure 3: Measures of anomalousness using the Hierarchical Sparseness method on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.

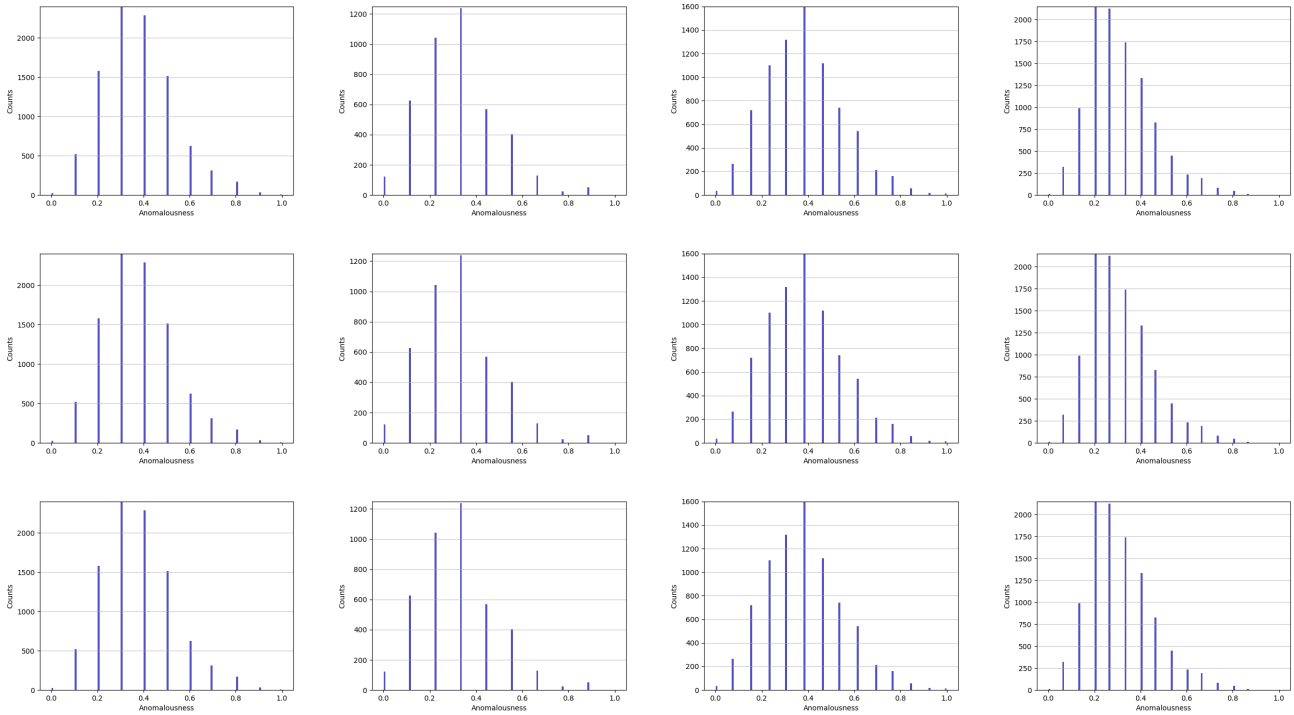

Figure 4: Example of a grid of regular-sized images.

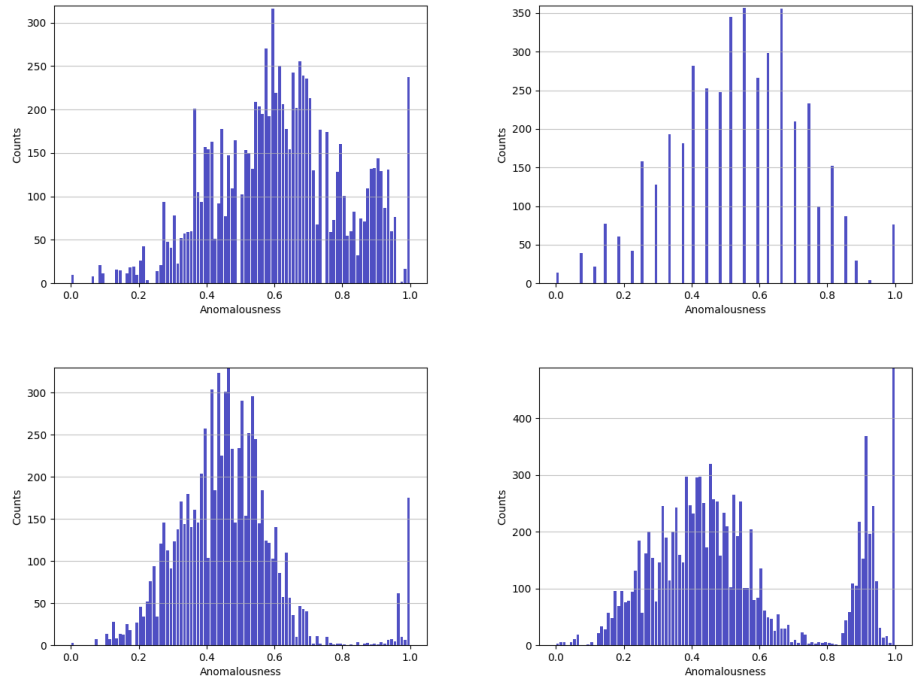

Figure 5: Measures of anomalousness using the Outrank Algorithm on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.

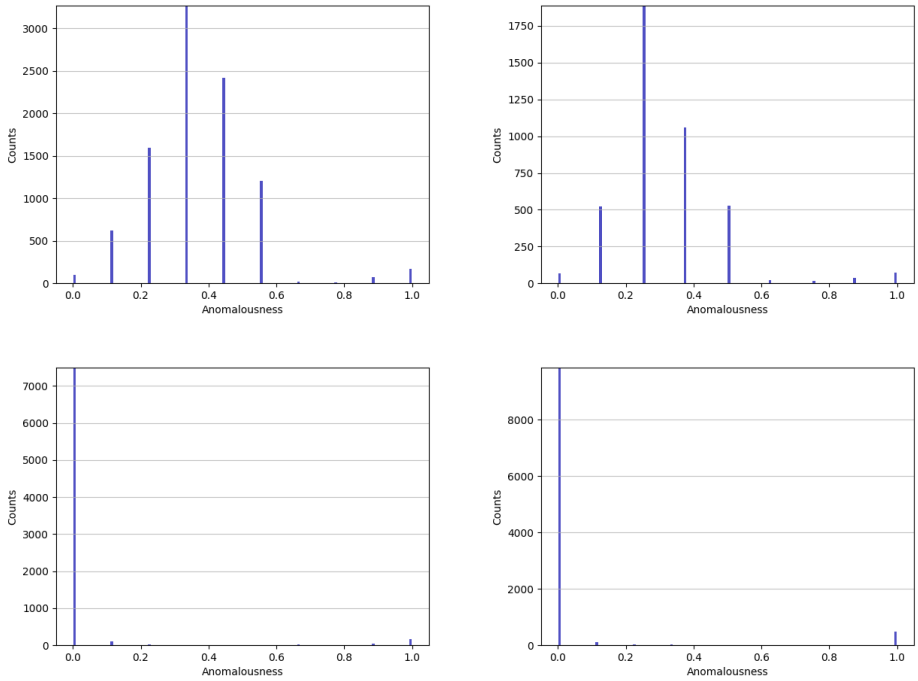

Figure 6: Measures of anomalousness using the k-Neighborhood Size method on the Bullseye (top-left), Spiral (top-right), Interlocking-Tori (bottom-left), and the Skewer (bottom-right) datasets.