

# Fast Outlier Detection Despite the Duplicates

Jay-Yoon Lee<sup>1</sup>

U Kang<sup>2</sup>

Danai Koutra<sup>1</sup>

Christos Faloutsos<sup>1</sup>

<sup>1</sup>Carnegie Mellon University

<sup>2</sup>KAIST

leejayyoon@cmu.edu {danai,christos}@cs.cmu.edu ukang@cs.kaist.ac.kr

## ABSTRACT

Given a large cloud of multi-dimensional points, and an off-the-shelf outlier detection method, why does it take a week to finish? After careful analysis, we discovered that duplicate points create subtle issues, that the literature has ignored: if  $d_{max}$  is the multiplicity of the most over-plotted point, typical algorithms are quadratic on  $d_{max}$ . We propose several ways to eliminate the problem; we report wall-clock times and our time savings; and we show that our methods give either exact results, or highly accurate approximate ones.

## 1. INTRODUCTION

Outlier detection, also known as anomaly detection, is an important area of data mining which has been receiving a lot of research attention [6, 5, 1, 2]. The size of the data to apply outlier detection algorithms is growing at an unprecedented rate. These big data pose new problems, such as the “duplicate data point” problem, which refers to the existence of many data points with same coordinates. For example, assume a 2-D dataset  $\langle \text{degree, PageRank} \rangle$  of nodes in a graph. Although for up to medium-sized graphs duplicates are not an issue, in billion-node graphs multiple nodes have the same feature pairs.

Traditional outlier detection algorithms did not consider the duplicate data point problem for two reasons: (i) they dealt with relatively small amount of data with few -if any- duplicates, and (ii) most outlier detection algorithms work on Geographical Information System (GIS) data which do not have many duplicates as buildings, obviously, never exist over a building.

**Challenges.** The existence of the duplicate data points in traditional algorithms poses two challenges. (a) **Degeneracy.** The outlier score may not be defined at all for duplicate points. (b) **Running Time.** The duplicate points retard the computation of the existing algorithms; e.g., the running time grows from near-linear to near-quadratic.

## 2. OBSERVATIONS

To address the “duplicate data point” issue, we focus on the widely used outlier detection scheme Local Outlier Factor algorithm (LOF) [3] which is based on k-nearest neighbor ( $kNN$ ), and suffers most from the problem of duplicates.

In a nutshell, LOF compares the local density of each data point to the densities of its neighbors by employing the  $kNN$  technique. Data points whose densities differ *much* from their neighbors’ densities are flagged as outliers.

**OBSERVATION 1 (LOF WITH DUPLICATES).** *LOF works fine with normal use, but has problems with large duplicate points. Numerous duplicates impede the calculation of LOF score by increasing the number of data access to  $O(\max(c_i^2))$ , where  $c_i$  is count of duplicates for unique element  $u_i$ .*

Table 1 shows the prevalence of the problem of duplicate points in real datasets.

	Stack Overflow		US Patent	
Top5	count	count <sup>2</sup>	count	count <sup>2</sup>
1	4221	17.8 M	60598	3.7 B
2	3799	14.4 M	59744	3.6 B
3	3147	9.9 M	56191	3.2 B
4	2844	8.1 M	49192	2.4 B
5	2374	5.6 M	41929	1.8 B
sum	16385 (6.70%)	55.8 M (61.7%)	267654 (12.9%)	14.7 B (79.5%)

Table 1: (M: million, B: billion.) The number of points (*count*) and the number of points squared (*count*<sup>2</sup>) for the top 5 largest duplicates in real world data. The top 5 duplicate counts comprise 6.70% and 12.90% of total number of points in Stack Overflow and US patent data, respectively. Also, they occupy 61.7% and 79.5% of the total sum of *count*<sup>2</sup>, so more than half of the computation is spent on these points.

## 3. PROPOSED METHOD: FAST OUTLIER DETECTION

The challenges mainly arise from regarding as unique the points that are similar in the projected feature space, and handling them separately (e.g., all the  $\langle \text{pagerank, degree} \rangle$  points with the same coordinates are viewed as distinct points). Our algorithms cleverly handle these problems.

### 3.1 FADD: Fast Anomaly Detection given Duplicates

FADD considers identical coordinates in the  $n$ -dimensional space a super node with their duplicate count information,  $c_i$ . More specifically, rather than visiting all of the  $N$  points separately, FADD only deals with  $M$  unique super nodes.

### 3.2 G-FADD: Grid based FADD

In the same spirit of FADD, we adopt a grid-based method, G-FADD, in order to analyze larger aggregate behavior, and reveal anomalous points in different granularities. The granularity is controlled by the parameter  $l$ , which denotes the number of grids each dimension holds: for  $n$ -dimensional space, the number of boxes would be  $l^n$ . The basic idea is to observe the behavior of each grid instead of the behavior of each point: we count the number of points that reside in each grid and run FADD only for the grids that

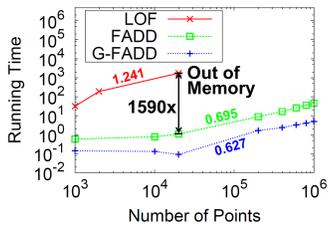


Figure 1: Runtime comparison of FADD and G-FADD vs. LOF, in log-log scale. LOF died out of memory for more than 20K data points. FADD and G-FADD runs 1590 $\times$  and 19132 $\times$  faster than LOF, respectively.

have a low count. The threshold for the count is set to  $k + 1$  - where  $k$  is defined by the  $k$ NN algorithm - , since it means that the grid is self-sufficient for querying nearest neighbors and dense enough to be exempt from outlier detection at the chosen granularity.

## 4. EXPERIMENTS

We use the Twitter data described in Table 2. The dataset contains multi-dimensional points whose dimensions (features extracted by Pegasus [4]) are specified in the ‘Description’ column. We run the experiments on a machine with 2 dual-core Intel Xeon 3.00 GHz, 16 GB memory and 480 GB hard disk, running Red Hat Linux 4.1.2.

Data	# Dimensions	# Points	Description
Twitter 2009	3	39,972,230	degree - PageRank - triangle
US Patent	2	2,076,613	degree - PageRank
Stack Overflow	2	243,776	degree - PageRank

Table 2: Summary of the data used. Each dataset contains multi-dimensional points whose dimensions are specified in the last column.

### 4.1 Running Time

We show how our proposed algorithms, FADD and G-FADD, outperform the existing algorithm. Figure 1 shows the running times of FADD and G-FADD, compared to LOF, on a synthetic 2-dimensional data with 50% duplicate points.

### 4.2 Impact of Granularity

Figure 2 shows the top 3 outliers from FADD and G-FADD with different grid granularities. Note that for all the data, the top outliers from FADD are local outliers buried in the middle of the main clouds of points because of the duplicate points in the main cloud. However, G-FADD with the coarsest grid granularity ( $l=8$ ) gives global outliers which are separated from the main clouds of points. Also note that as the grid granularity becomes finer, the output from G-FADD gets closer to that from FADD.

Fine granularity (local outliers)  $\longrightarrow$  Coarse granularity (global outliers)

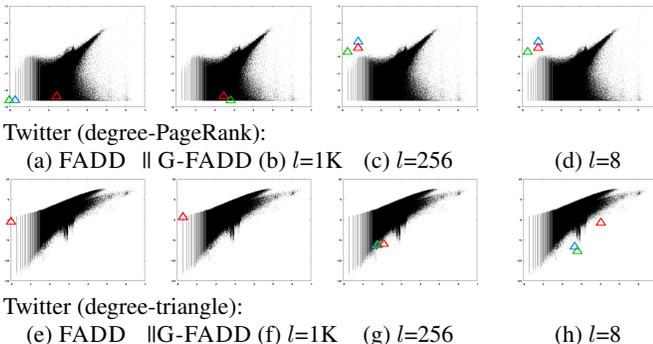


Figure 2: [Best viewed in color.] 2-D scatter plot highlighting top 3 outliers from FADD and G-FADD with different grid granularity  $l$ . The blue, green, and red triangles denote the points with the 1st, 2nd, and 3rd largest outlier scores, respectively.

## 4.3 G-FADD on Real Data

**Twitter degree-PageRank..** The top 3 outliers in Figure 2 (d), Twitter degree-PageRank plot, are unpopular accounts with very small degrees (7, 2, and 7, respectively), but they have relatively high PageRank values which make themselves outstanding. It turns out their neighbors have relatively high degrees: the average degrees of neighbors are 1646, 89, and 343014, respectively. Due to the many neighbors, they have higher PageRanks despite their low degrees.

**Twitter degree-triangle..** In Figure 2 (h) degree-triangle plot, each point in the plot represents the degree and the number of participating triangles of a node in the Twitter who-follows-whom graph. All the top 3 outliers have relatively small number of triangles compared to their neighbors. The top outlier (blue triangle) is likely to be an advertisement spammer since it has only 3 tweets which are all about free gift card offers from Wal-Mart and Best Buy, and it has no followees at all. It has few triangles since the followers are not likely to know each other, although some of them had same interest and gave some triangle in a stroke of luck. The third outlier (red triangle) is an account of a comics character which has 11207 followers and 6 followees. It seems to have few triangles because the fans (followers) of the character might not be close friends with each other.

## 5. CONCLUSION

In this paper we propose FADD and G-FADD, scalable algorithms that detect outliers from multi-dimensional points despite large duplicates. Our main contributions are: (1) **No Degeneracy.** We re-design the standard outlier detection algorithm to overcome the problem of duplicate points in large, real world data; (2) **Running Time.** Our algorithms enjoy near-linear runtime compared to the near-quadratic running time of the existing algorithm; (3) **Discovery.** We analyze large, real world data, and find interesting outliers.

## 6. ACKNOWLEDGMENTS

Funding was provided by the Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088. Research was also sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## 7. REFERENCES

- [1] L. Akoglu, M. McGlohon, and C. Faloutsos. OddBall: Spotting Anomalies in Weighted Graphs. In *PAKDD*, 2010.
- [2] V. Barnett and T. Lewis. *Outliers in statistical data*. John Wiley & Sons Ltd., 2nd edition edition, 1978.
- [3] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *SIGMOD Conference*, pages 93–104, Dallas, TX, 2000.
- [4] U. Kang, C. Tsourakakis, and C. Faloutsos. Pegasus: A peta-scale graph mining system - implementation and observations. In *ICDM*, 2009.
- [5] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos. LocI: Fast outlier detection using the local correlation integral. In *ICDE*, pages 315–326, 2003.
- [6] J. Tang, Z. Chen, A. W. Fu, and D. W. Cheung. Enhancing Effectiveness of Outlier Detections for Low Density Patterns. In *PAKDD*, pages 535–548, 2002.