Anomaly Detection in Large Graphs

Leman Akoglu Mary McGlohon Christos Faloutsos

November 2009
CMU-CS-09-173

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Discovering anomalies is an important and challenging task for many settings, from network intrusion to fraud detection. However, most work to date has focused on clouds of multi-dimensional points, with little emphasis on graph data; even then, the focus is on un-weighted, node-labeled graphs. Here we propose `OddBall`, an algorithm to detect anomalous nodes in weighted graphs. The contributions are the following: (a) we carefully choose features, that easily reveal nodes with strange behavior; (b) we discover several new rules (power laws) in density, weights, ranks and eigenvalues that seem to govern the so-called "neighborhood graphs" and we show how to use them for anomaly detection; (c) we empirically show that our method scales linearly with the number of edges in the graph, and (d) we report experiments on many real graphs with up to *1.5 million* nodes, where `OddBall` indeed spots unusual nodes that agree with intuition.

Keywords: outlier, anomaly detection, graph mining

1 Introduction

Given a real graph, with weighted edges, which nodes should we consider as "strange"? Applications of this setting abound: For example, in network intrusion detection, we have computers sending packets to each other, and we want to know which nodes misbehave (e.g., spammers, port-scanners). In a who-calls-whom network [33], strange behavior may indicate defecting customers, or telemarketers, or even faulty equipment dropping connections too often. In a social network, like FaceBook and LinkedIn, again we want to spot users whose behavior deviates from the usual behavior, such as people adding friends indiscriminately, in "popularity contests".

The list of applications continues: Anomalous behavior could signify irregularities, like credit card fraud, calling card fraud, campaign donation irregularities, accounting inefficiencies or fraud [6], extremely cross-disciplinary authors in an author-paper graph [34], suspicious cargo shipments [16], electronic auction fraud [12, 31], and many others.

In addition to revealing suspicious, illegal and/or dangerous behavior, anomaly detection is useful for spotting rare events, as well as for the thankless, but absolutely vital task of data cleansing [14, 15]. Moreover, anomaly detection is intimately related with the pattern and law discovery: unless the majority of our nodes closely obey a pattern (say, a power law), only then can we confidently consider as outliers the few nodes that deviate [9].

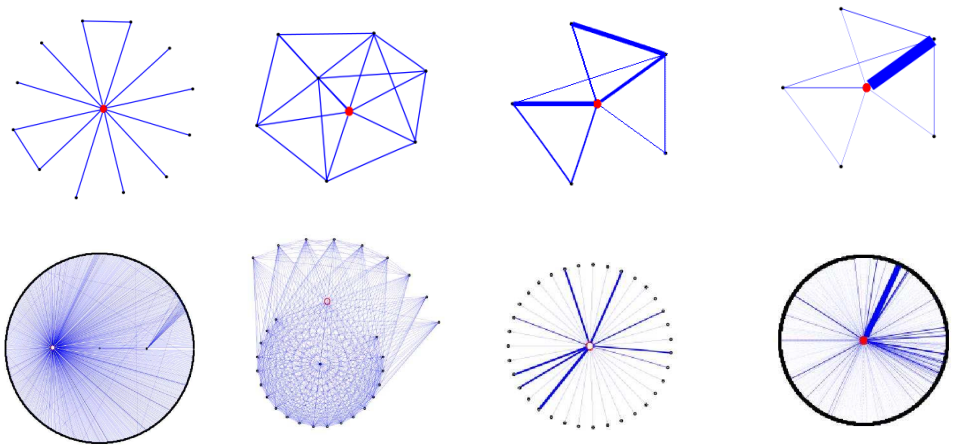
Most anomaly detection algorithms focus on clouds of multi-dimensional points, as we describe in the survey section. Our goal, on the other hand, is to spot strange nodes in a *graph*, with weighted edges. What patterns and laws do such graphs obey? What features should we extract from each node? We answer all these questions.

Main contributions of this work are:

1. *Feature extraction*: We propose to focus on neighborhoods, that is, a sphere, or a ball (hence the name `OddBall`) around each node(the *ego*): that is, for each node, we consider the induced subgraph of its neighboring nodes, which is referred to as the *egonet*. Out of the huge number of numerical features one could extract from the egonet of a given node, we give a carefully chosen list, with features that are both fast to compute, as well as effective in revealing outliers. Thus, every node becomes a point in a low-dimensional feature space.
2. *Egonet patterns*: We show that egonets obey some surprising patterns (like the *Egonet Density Power Law* (*EDPL*), *EWPL*, *ELWPL*, and *ERPL*), which gives us confidence to declare as outliers the ones that deviate. We support our observations by showing that the *ERPL* yields the *EWPL*.
3. *Our hybrid algorithm*: Based on those patterns, we propose `OddBall`, a novel, hybrid method for outlier node detection. The method works well for graphs with *millions* of nodes, and it scales *linearly* with number of edges. Moreover, we propose several approximations that improve its speed, with small or zero impact on its accuracy.
4. *Application on real data*: We apply `OddBall` to numerous real graphs (DBLP, political donations, and other domains) and we show that it indeed spots nodes that a human would agree are strange and/or extreme.

Jumping ahead, the major types of anomalous nodes we can spot are as follows (see Fig.1 for examples).

1. *Near-cliques* and *stars*: Detecting those nodes whose neighbors are very well connected (near-cliques) or not connected (stars) turn out to be "strange": in most social networks, friends of friends are often friends, but either extreme (clique/star) is suspicious.
2. *Heavy vicinities*: If person i has contacted n distinct people in a who-calls-whom network, we would expect that the number of phonecalls (weight) would be proportional to n (say, 3xn or 5xn). Extreme total weight would be suspicious, indicating, e.g., faulty equipment that forces redialing.
3. *Dominant heavy links*: In the who-calls-whom scenario above, a very heavy single link in the 1-step neighborhood of person i is also suspicious, indicating, e.g., a stalker that keeps on calling only one of his/her contacts an excessive count of times.



(a) Near-star (b) Near-clique (c) Heavy vicinity (d) Dominant edge

Figure 1: Types of anomalies that `OddBall` detects. Top row: toy sketches of egonets (ego shown in larger, red circle). Bottom row: actual anomalies spotted in real datasets. (a) A near-star in *Postnet*: Instapundit, on Hurricane Katrina relief agencies (`instapundit.com/archives/025235.php`): An extremely long post, with many updates, and numerous links to diverse other posts about donations. (b) A near-clique in *Postnet*: `sizemore.co.uk`, who often linked to its own posts, as well as to its own posts in other blogs. (c) A heavy vicinity in *Postnet*: *blog.searchenginewatch.com/blog* has abnormally high weight wrt the number of edges in its egonet. (d) Dominant Edge(s) in *Com2Cand*: In FEC 2004, George W. Bush received a huge donation from a single donor committee: Democratic National Committee (~$\$87M$) (!) - in fact, this amount was *spent against* him; Next heaviest link (~$\$25M$): from Republican National Committee.

The rest of the paper is organized as follows: Section 2 gives the survey. Section 3 describes the datasets we studied to spot outlier nodes. Section 4 gives the primary observations and the description of `OddBall`. Section 5 shows experimental results, and finally Section 6 concludes.

2 Related Work

2.1 Outlier Detection

Outlier detection has attracted wide interest, being a difficult problem, despite its apparent simplicity. Even the definition of the outlier is hard to give: For instance, Hawkins [20] defines an outlier as "an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism." Similar, but not identical, definitions have been given by Barnett and Lewis [5], and Johnson [21].

Outlier detection methods form two classes, *parametric* and *non-parametric*. Statistical parametric methods assume there exists a standard underlying distribution of the observations that fit the data [20, 5]. The latter class includes *distance-based* and *density-based* data mining methods. These methods typically define as an outlier the (n-D) point that is too far away from the rest, and thus lives in a low-density area [23]. Typical methods include LOF [8] and LOCI [32]. These methods not only flag a point as an outlier but they also give outlierness scores; thus, they can sort the points according to their "strangeness".

Many other *density-based* methods that perform well in detecting outliers in very large datasets of high dimension are proposed in [1, 4, 13, 19, 29, 36, 22]. Feature bagging [24] also proves to be useful to tackle high dimensionality, where features are randomly grouped into multiple sets of different sizes and outlier detection algorithms are performed on each different set after which the scores are combined. Finally, most clustering algorithms [19, 29, 36, 22] reveal outliers as a by-product.

2.2 Anomaly Detection in Graph Data

Noble and Cook [30] detect anomalous sub-graphs using variants of the *Minimum Description Length* (MDL) principle. Eberle and Holder [17] also use the MDL principle as well as other probabilistic measures to detect several types of anomalies (e.g. unexpected/missing nodes/edges). Liu et. al [26] detect noncrashing bugs in software using frequent execution flow graphs combined with classification. Chakrabarti [10] uses MDL to spot anomalous edges. Sun et al. [34] use proximity and random walks, to assess the normality of nodes in bipartite graphs. OutRank and LOADED [18, 28] use similarity graphs of objects to detect outliers.

In contrast to the above, we work with *unlabeled* graphs. We explicitly focus on nodes, while interactions are also considered implicitly as we study *neighborhood subgraphs*. Finally, we consider both bipartite and unipartite graphs as well as edge *weights*.

2.3 Real-world graph properties

Several properties of real-world unweighted graphs have been discovered, surveyed in [11]. In our study, we explicitly concentrate on anomalous node detection in *weighted* graphs, adding to laws of weighted graphs discovered in [2, 27] and relying on these observations to detect anomalies.

Name	$\|V\|$	$\|E\|$	Weights	Structure	Description
Blognet	27K	126K	Yes	Unipartite	Network of blogs based on citations
Postnet	223K	217K	Yes	Unipartite	Network of posts based on citations
Auth2Conf	421K	1M	Yes	Bipartite	DBLP Author/Conference associations
Com2Cand	6K	125K	Yes	Bipartite	2004 US FEC Committee to Candidate donations
Don2Com	1,6M	2M	Yes	Bipartite	2004 US FEC Donor to Committee donations
Enron	36K	183K	No	Unipartite	Email associations at Enron
Oregon	11K	38K	No	Unipartite	AS peering connections

Table 1: Datasets studied in this work.

3 Data Description

We studied several unipartite/bipartite, weighted/unweighted large real-world graphs in a variety of domains, described in detail in Table 1. Particularly, unipartite networks include the following: *Postnet* contains post-to-post links in a set of blogs[25], *Blognet* contains blog-to-blog links in the same set, *Enron* contains emails at Enron collected from about 1998 to 2002 (made public by the Federal Energy Regulatory Commission during its investigation), and *Oregon* contains AS peering information inferred from Oregon route-views BGP data. Bipartite networks include the following: *Auth2Conf* contains the publication records of authors to conferences from DBLP, and *Don2Com* and *Com2Cand* are from the U.S. Federal Election Commission in 2004[1], a public record of donations between donors and committees and between committees and political candidates, respectively.

For *Don2Com* and *Com2Cand*, the weights on the edges are actual weights representing donation amounts in dollars. For the remaining weighted datasets, the edge weights are simply the number of occurrences of the edges. For instance, if blog i contains k posts with links to another blog j, the weight of the edge $e_{i,j}$ is set to k.

In our study, we specifically focused on undirected graphs, but the ideas can easily be generalized to directed graphs.

4 Proposed Method

Borrowing terminology from social network analysis (SNA), "ego" is an individual node.

Informally, an ego (=node) of a given network is anomalous if its neighborhood significantly differs from those of others. The basic research questions are: (a) *what features should we use to characterize a neighborhood?* and (b) *what does a 'normal' neighborhood looks like?*

[1] Parsed dataset from all cycles may be found at `www.cs.cmu.edu/~mmcgloho/fec/data/fec_data.html`

Both questions are open-ended, but we give some answers below. First, let's define terminology: the "*k*-step neighborhood" of node i is the collection of node i, all its k-step-away nodes, and all the connections among all of these nodes – formally, this is the "*induced subgraph*". In SNA, the 1-step neighborhood of a node is specifically known as its "*egonet*".

How should we choose the value of k steps to study neighborhoods? Given that real-world graphs have small diameter [3], we need to stay with small values of k, and specifically, we recommend k=1. We report our findings only for k=1, because using $k > 1$ does not provide any more intuitive or revealing information, while it has heavy computational overhead, possibly intractable for very large graphs.

4.1 Feature Extraction

The first of our two inter-twined questions is *which statistics/features to extract* from a neighborhood.

Intuitively, we should select easy-to-compute features which help spot the type of anomalies we are interested in. Types of anomalies we want to find and related features are the following: (1) for *near-clique* and *stars*, we want to see the relation between the number of entities and the number of links between them, as strange patterns among "friends of friends" would arise suspicion; (2) for *heavy vicinities*, or a highly active egonet, total weight should significantly surpass the number of links; and (3) for *dominant heavy link*, or a single highly active link in an egonet, the eigenvalue of the weighted adjacency matrix of the egonet should be very close to the total weight. To accommodate detection of all of these anomalies, we propose to use the following features in our study.

1. N_i: number of neighbors (degree) of ego i.
2. E_i: number of edges in egonet i.
3. W_i: total weight of egonet i,
4. $\lambda_{w,i}$: principal eigenvalue of the *weighted* adjacency matrix of egonet i.

The next question is how to look for outliers, in such an n-dimensional feature space, with one point for each node of the graph. In our case, n=4, but one might have more features depending on the application and types of anomalies one wants to detect. A quick answer to this would be to use traditional outlier detection methods for clouds of points using all the features.

In our setting, we can do better. As we show next, we group features into carefully chosen pairs, where we expect to find new patterns of normal behaviour. We flag those points that significantly deviate from the discovered patterns as anomalous. Pairs of features we studied and the types of anomalies each pair helps to detect are the following:

- E vs N: *CliqueStar*: detects near-cliques and stars
- W vs E: *HeavyVicinity*: detects many reoccurences of interactions
- λ_w vs W: *DominantPair*: detects single dominating heavy edge (strongly connected pair)

Here, we note that we studied other features such as the effective radius of each node, number of neighbors of degree 1, the principal eigenvalues of egonets, etc. as well as triplets of features. However, we do not show these results for the sake of brevity.

4.2 Laws and Observations

The second of our research questions is *what do normal neighborhoods look like*. Thus, it is important to find patterns ("laws") for neighborhoods of real graphs, and then report the deviations, if any. In this work, we report some new, surprising patterns:

For a given graph $\mathcal{G}$, node $i \in \mathcal{V}(\mathcal{G})$, and the egonet $\mathcal{G}_i$ of node i,

Observation 1 (*EDPL*: Egonet Density Power Law) *the number of nodes N_i and the number of edges E_i of $\mathcal{G}_i$ follow a power law.*

$$E_i \propto N_i^{\alpha}, \;\; 1 \leq \alpha \leq 2.$$

In our experiments the *EDPL* exponent α ranged from 1.10 to 1.66. Fig. 3 illustrates this observation, for several of our datasets. Plots show E_i versus N_i for every node (green points); the black circles are the median values for each bucket of points (separated by vertical dotted lines) after appyling logarithmic binning on the x-axis as in [27]; the red line is the least squares(LS) fit on the medians. The plots also show a blue line of slope 2, that corresponds to cliques, and a black line of slope 1, that corresponds to stars. All the plots are in log-log scales.

Observation 2 (*EWPL*: Egonet Weight Power Law) *the total weight W_i and the number of edges E_i of $\mathcal{G}_i$ follow a power law.*

$$W_i \propto E_i^{\beta}, \;\; \beta \geq 1.$$

In our experiments the *EWPL* exponent β ranged up to 1.29 (See Fig. 4). Values of $\beta > 1$ indicate superlinear growth in the total weight with respect to increasing total edge count in the egonet.

Observation 3 (*ELWPL*: Egonet λ_w Power Law) *the principal eigenvalue of the weighted adjacency matrix $\lambda_{w,i}$ and the total weight W_i of $\mathcal{G}_i$ follow a power law.*

$$\lambda_{w,i} \propto W_i^{\gamma}, \;\; 0.5 \leq \gamma \leq 1.$$

In our experiments the *ELWPL* exponent γ ranged from 0.53 to 0.98 (See Fig. 5). γ=0.5 indicates uniform weight distribution whereas γ~1 indicates a dominant heavy edge, in which case the weighted eigenvalue follows the maximum edge weight. γ=1 if the egonet contains only one edge.

Observation 4 (*ERPL*: Egonet Rank Power Law) *the rank $R_{i,j}$ and the weight $W_{i,j}$ of edge j in $\mathcal{G}_i$ follow a power law.*

$$W_{i,j} \propto R_{i,j}^{\theta}, \;\; \theta \leq 0.$$

The *ERPL* suggests that edge weights in the egonet have a skewed distribution. This is intuitive; for example in a friendship network, a person could have many not-so-close friends (small weight links), but only a few close friends (heavy links). In Fig. 2 we show the *ERPL* for top three nodes with the highest number of edges in their egonet from *Blognet* – other datasets have similar results.

Next we show that if the *ERPL* holds, then the *EWPL* also holds. Given an egonet graph $\mathcal{G}_i$, the total weight W_i and the number of edges E_i of $\mathcal{G}_i$, let $\mathcal{W}_i$ denote the ordered set of weights of the edges, $W_{i,j}$ denote the weight of edge j, W_{min} be the minimum edge weight, and $R_{i,j}$ denote the rank of weight $W_{i,j}$ in set $\mathcal{W}_i$. Then,

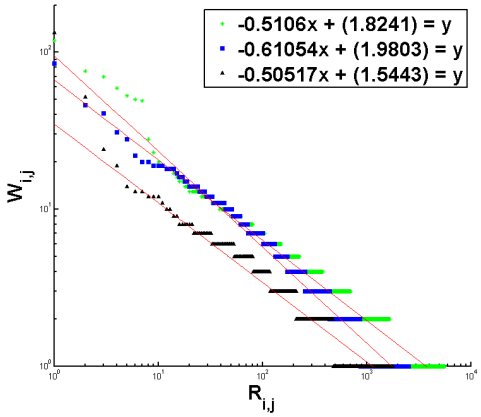


Figure 2: Weight $W_{i,j}$ vs. Rank $R_{i,j}$ for each edge j in the egonet of node i. Top three nodes with the highest edgecount in their egonet from *Blognet* are shown. LS line is fit in log-log scales pointing out a power-law relation (*ERPL*).

Lemma 1 *If* $W_{i,j} \propto R_{i,j}^{\theta}$, $\theta \leq 0$, *then*

$$W_i \propto E_i^{\beta} \begin{cases} \beta = 1, \textit{ if } -1 \leq \theta \leq 0 \\ \beta > 1, \textit{ if } \theta < -1 \end{cases}$$

Proof 1 *For brevity, we give the proof for* $\theta < -1$ *– other cases are similar. Given that* $W_{i,j} = cR_{i,j}^{\theta}$, $W_{min} = cE_i^{\theta}$, *i.e.* $c = W_{min}E_i^{-\theta}$. *Then we can write* W_i *as*

$$\begin{aligned} W_i &= W_{min}E_i^{-\theta}\left(\sum_{j=1}^{E_i} j^{\theta}\right) \approx W_{min}E_i^{-\theta}\left(\int_{j=1}^{E_i} j^{\theta} dj\right) \\ &= W_{min}E_i^{-\theta}\left(\frac{j^{\theta+1}}{\theta+1}\Big|_{j=1}^{E_i}\right) \\ &= W_{min}E_i^{-\theta}\left(\frac{1}{-\theta-1} - \frac{1}{(-\theta-1)E_i^{-\theta-1}}\right) \end{aligned}$$

For large E_i *and considering* $\theta < -1$, *the second term in parenthesis goes to* 0. *Therefore;*

$$W_i \approx c' E_i^{-\theta}$$

where $c' = \frac{W_{min}}{-\theta-1}$, *and since* $\theta < -1$, $\beta > 1$.

4.3 Anomaly Detection

We can easily use the observations given in part 4.2 in anomaly detection since anomalous nodes would behave away from the normal pattern. To score the outlierness of a node, we take the distance-to-fitting-line as a measure. Let us define the y-value of a node i as y_i and similarly, let

x_i denote the x-value of node i for a particular feature pair $f(x, y)$. Given the power law equation $y = Cx^\theta$ for $f(x, y)$, we define the outlierness score of node i to be

$$\textit{out-line(i)} = \frac{max(y_i, Cx_i^\theta)}{min(y_i, Cx_i^\theta)} * log(|y_i - Cx_i^\theta| + 1)$$

Here we penalize the nodes with both the *number of times* that they deviate from their expected y-value for their x-value and with the logarithm of the *amount* of deviation. This way, the minimum outlierness score is 0 –for which the actual value y_i is equal to the expected value Cx_i^θ.

This simple and easy-to-compute method not only helps in detecting outliers, but also provides a way to sort the nodes according to their outlierness scores. However, this method is prone to yield false positives for the following reason: Assume that there exists some points that are far away from the remaining points but that are still located close to the fitting line. In our experiments with real data, we observe that this is almost always the case for high values of x and y. For example, in Fig. 3(a), the points marked with left-triangles ($\lhd$) are almost on the fitting line even though they are far away from the rest of the points.

We want to flag both types of points as outliers, and thus we propose to combine our heuristic with a density-based outlier detection technique. We used LOF [8], which also assigns outlierness scores *out-lof(i)* to data points; but any other outlier detection method would do, as long as it gives such a score. To obtain the final outlierness score of a data point i, one might use several methods such as taking a linear function of both scores and re-ranking the nodes according to the new score, or merging the two ranked lists of nodes. In our work, we simply used the sum of the two normalized(by dividing by the maximum) scores, that is, *out-score(i)* = *out-line(i)*+*out-lof(i)*.

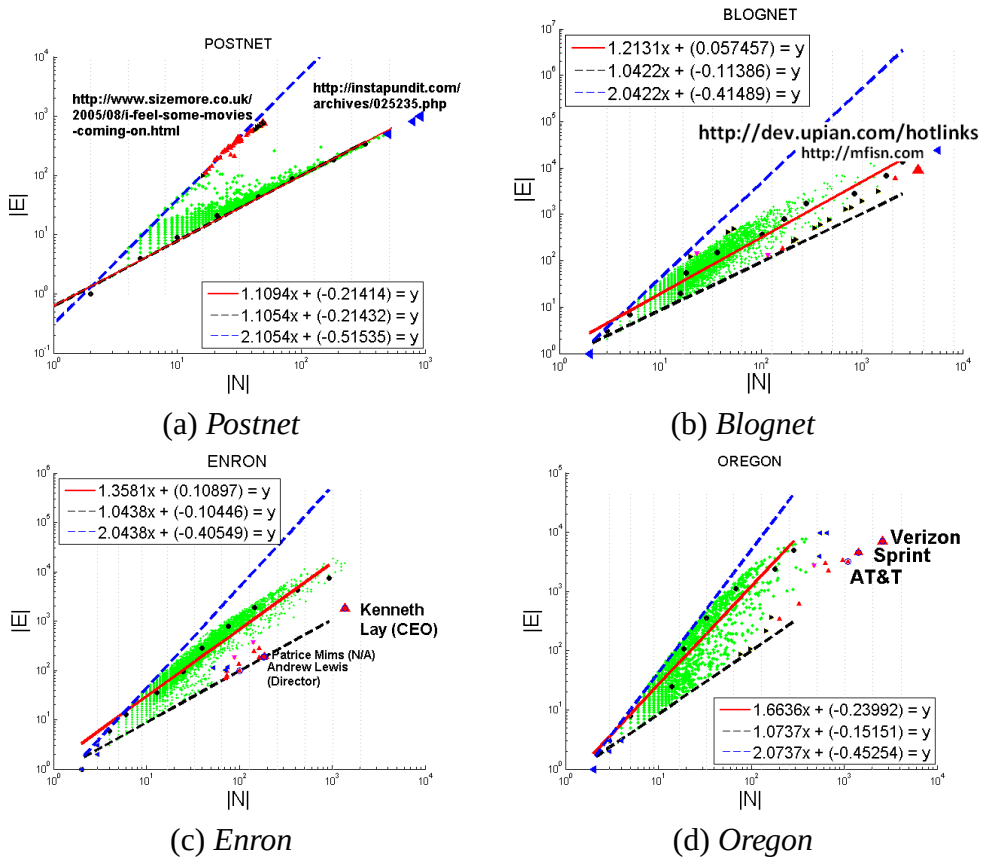
5 Experimental Results

5.0.1 *CliqueStar*

Here, we are interested in the communities that neighbors of a node form. In particular, *CliqueStar* detects anomalies having to do with near-cliques and near-stars. Using *CliqueStar*, we were successful in detecting many anomalies over the unipartite datasets (although it is irrelevant for bipartite graphs since by nature the egonet forms a "star").

In social media data *Postnet* and *Blognet*, we detected posts or blogs that had either all their neighbors connected (cliques) or mostly disconnected (stars). We show some illustrative examples along with descriptions from *Postnet* in Fig. 1. See Fig.3a for the detected outliers on the scatter-plot from the same dataset. In *Blognet*(Fig.3b), the method detected several "link blogs", blogs devoted to posting links to a wide array of sources that do not always have similar content. For instance `mfisn.com` links to tech blogs, politics stories, and flash cartoons. `dev.upian.com/hotlinks` also links to a wide range of other posts each day.

In *Enron*(Fig.3c), the node with the highest outlier score turns out to be "*Kenneth Lay*", who was the CEO and is best known for his role in the Enron scandal in 2001. Our method reveals that none of his over 1K contacts ever sent emails to each other. In *Oregon* (Fig.3d), the top outliers are the three large ISPs ("*Verizon*", "*Sprint*" and "*AT&T*").



(a) *Postnet* (b) *Blognet*

(c) *Enron* (d) *Oregon*

Figure 3: Illustration of the Egonet Density Power Law (*EDPL*), and the corresponding anomaly *CliqueStar*, with outliers marked with triangles. Edge count versus node count (log-log scale); red line is the LS fit on the median values (black circles); dashed black and blue lines have slopes 1 and 2 respectively, corresponding to stars, and cliques. Top outlying points are enlarged. Most striking outlier: Ken Lay (CEO of Enron), with a star-like neighborhood. See Section 5.1.1 for more discussion and Fig.1 for example illustrations from *Postnet*.

5.0.2 *HeavyVicinity*

HeavyVicinity detects nodes that have considerably high total edge weight compared to the number of edges in their egonet. In our datasets, *HeavyVicinity* detected "heavy egonets", with anomalies marked in Fig.4. In *Postnet*(Fig.4h), often anomalous posts were the ones that linked to another post repeatedly [2] or were listed as the blog's home page[3].

In *Blognet* (Fig.4g), we detected blogs that linked to just a few other sources, either a single post or multiple posts from the same blog. Interesting anomalies are "*Automotive News Today*"[4], which linked 241 times to the GM blog[5], but never to any other blog in our dataset. One political

[2] `leados.blogs.com/blog/2005/08/overview_of_cia.html`
[3] `blog.searchenginewatch.com/blog`
[4] `www.automotive-news-today.com`
[5] `fastlane.gmblogs.com`

blog [6] had 1816 total in- and out-links, to only 30 other blogs. On the other extreme, "*Bandelier*"[7] had an abnormally high number of edges, but a low weight upon each. The blog had a single post in the dataset, so it was naturally uncommon for blogs to link to that blog multiple times.

HeavyVicinity revealed interesting observations in bipartite graphs as well, spotting duplicates and irregularities. In *Com2Cand*(Fig.4a,b), we see that "*Democratic National Committee*" gave away a lot of money compared to the number of candidates that it donated to. In addition, "*(John) Kerry Victory 2004*" donated a large amount to a single candidate, whereas "*Liberty Congressional Political Action Committee*" donated a very small amount ($5) to a single candidate. Looking at the *Candidates* plot for the same bipartite graph, we also flagged "*Aaron Russo*", the lone recipient of that PAC. (In fact, Aaron Russo is the founder of the Constitution Party which never ran any candidates, and Russo shut it down after 18 months.)

In *Don2Com*(Fig.4c,d), we see that "*Bush-Cheney '04 Inc.*" received a lot of money from a single donor. This is strange, as it was also listed as an anomaly in *LoneStar*, having many degree 1 donors. Looking at the data, we notice that that committee is listed twice with two different IDs.

On the other hand, we notice that the "*Kerry Committee*" received less money than would be expected looking at the number of checks it received in total. Further analysis shows that most of the edges in its egonet are of weight 0, showing that most of the donations to that committee have actually been returned.

In *Auth2Conf* (Fig.4e,f), "*Averill M. Law*" published 40 papers to the "*Winter Simulation Conference*" and nowhere else. This might be due to the fact that there exists no other conference that would capture the interest of that author. In fact, under *LoneStar*, we saw that "*Winter Simulation Conference*" was one of those conferences with most of its authors with degree 1, pointing to the same possibility that it is a unique conference in a particular area.

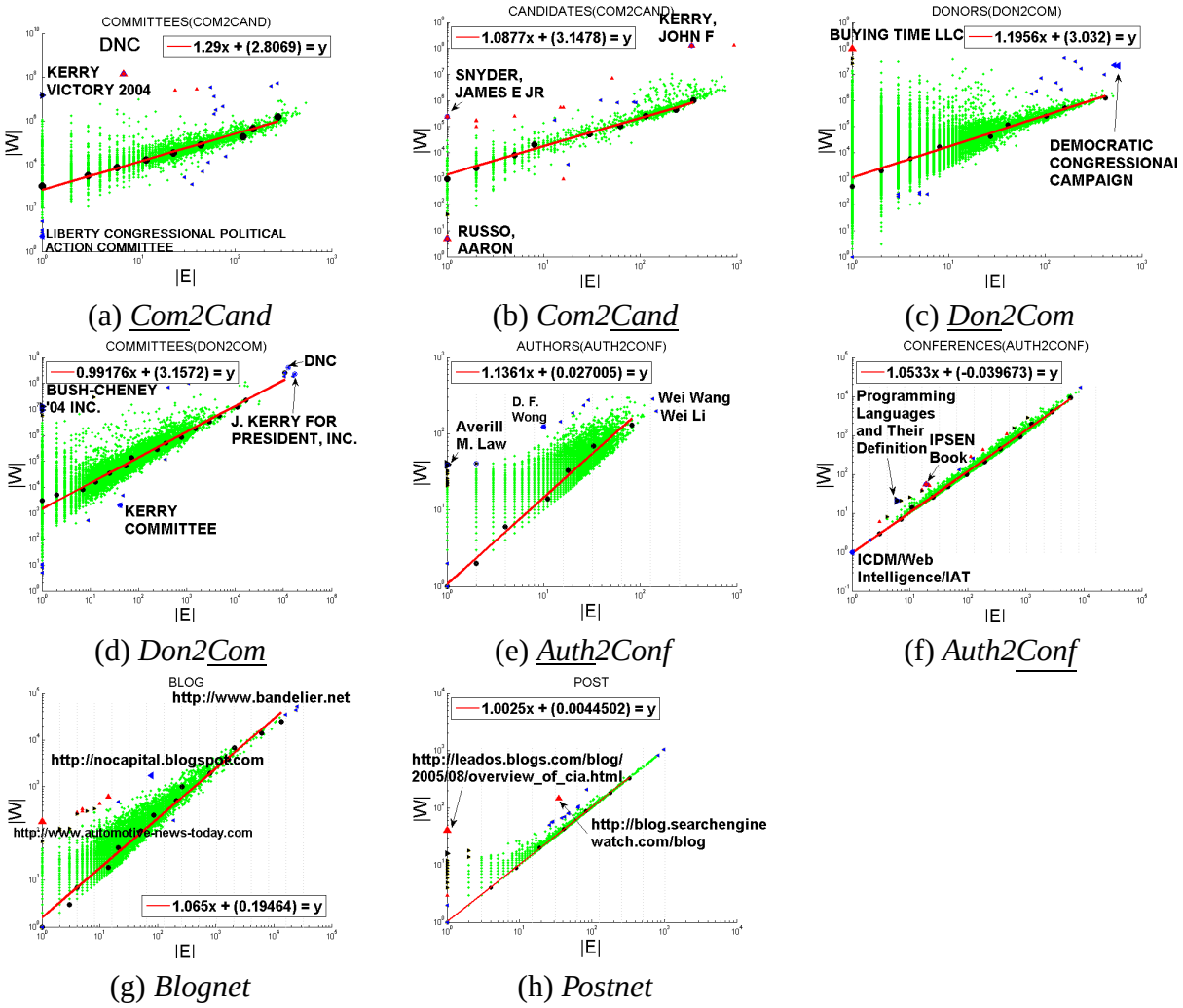
Here, other interesting points are "*Wei Wang*" and "*Wei Li*". Those authors have many papers, but they get them published to as many distinct conferences, probably once or twice to each conference.

5.0.3 *DominantPair*

DominantPair measures whether there is a single dominant heavy edge in the egonet. In other words, this method detected "bursty" if not exclusive edges. In *Postnet*(Fig.5h) nodes such as "*ThinkProgress*"'s post on a leak scandal [8] and "*A Freethinker's Paradise*" post [9] linking several times to the "*ThinkProgress*" post were both flagged. In *Blognet*(Fig. 5g), we detected a "*Drudge*" blogger [10], who had 298 links, all but 4 to another blogger in the same site [11]. "*Nocapital*" also appeared here, since it had around 300 links each to two other blogs.

In *Com2Cand*(Fig.5a,b), "*Democratic National Committee*" is one of the top outliers. We would guess that the single large amount of donation was made to "*John F. Kerry*". Counterintu-

[6] nocapital.blogspot.com
[7] www.bandelier.com
[8] www.thinkprogress.org/leak-scandal
[9] leados.blogs.com/blog/2005/08/overview_of_cia.html
[10] drudge.com/user/rcade
[11] drudge.com/user/gzlives



(a) *Com2Cand* (b) *Com2Cand* (c) *Don2Com*

(d) *Don2Com* (e) *Auth2Conf* (f) *Auth2Conf*

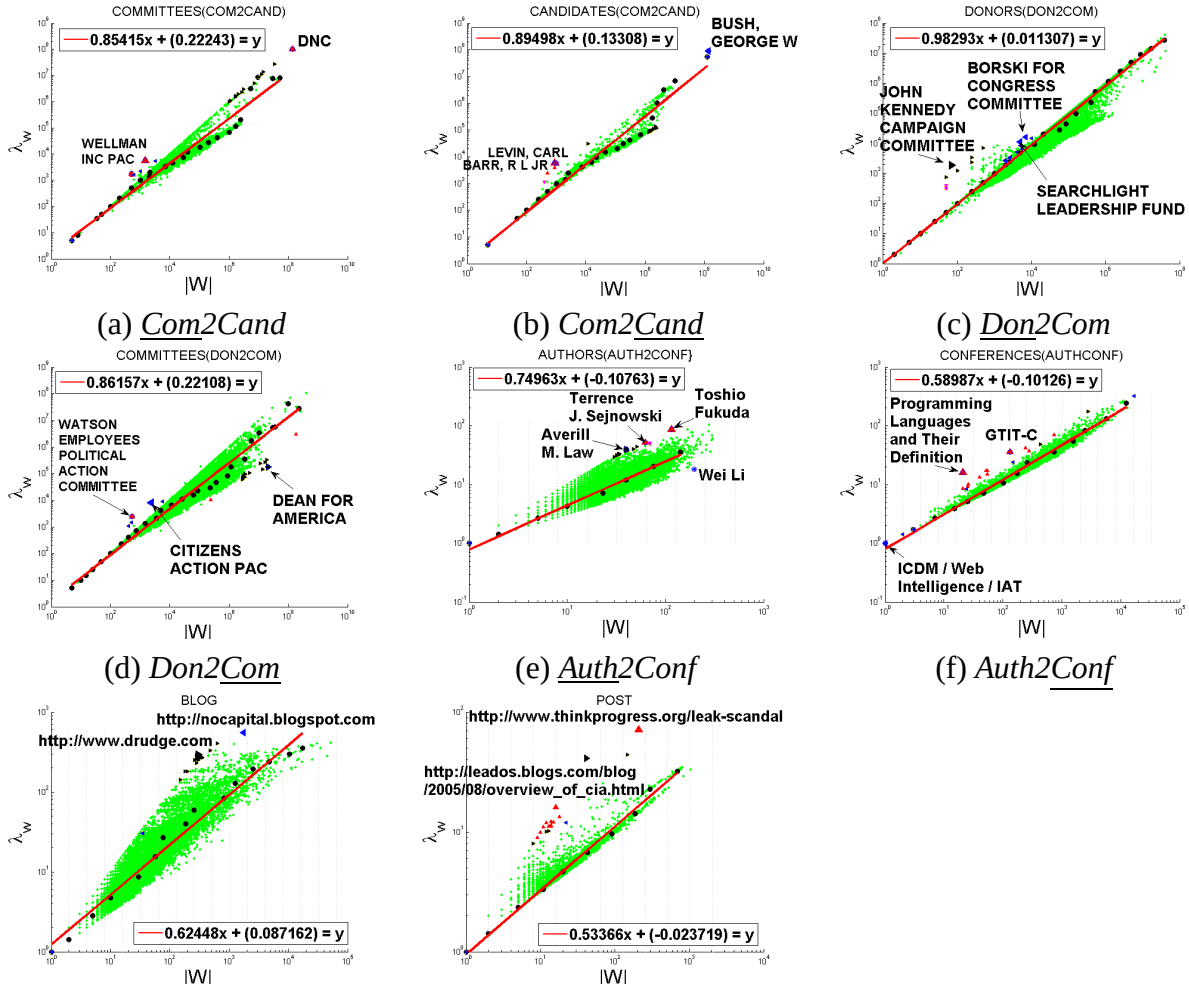
(g) *Blognet* (h) *Postnet*

Figure 4: Illustration of the Egonet Weight Power Law (*EWPL*) and the weight-edge anomaly *HeavyVicinity*. Plots show total weight vs. total count of edges in the egonet for all nodes(in log-log scales). Detected outliers include Democratic National Committee and John F. Kerry (in FEC campaign donations), and Averill M. Law in DBLP. See Section 5.2.1 for more discussions, and Fig.1 for an illustrative example from *Postnet*.

itively, however, we see that that amount was spent for an opposing advertisement against "*George W. Bush*").

In *Don2Com*(Fig.5c,d), there exists points above the slope 1 line. Further inspection shows that these points correspond to authorities having negative weighted edges due to returns. Points below the fitting line, such as "*Dean For America*" on the *Committees* plot, correspond to authorities with even weight distributions on the edges, without any particular dominant heavy link.

DominantPair flagged extremely focused authors (those publish heavily to one conference) in the *DBLP* data, shown in Fig.4(e,f). For instance, "*Toshio Fukuda*" has 115 papers in 17 conferences (at the time of data collection), with more than half (87) of his papers in one particular



(a) *Com2Cand* (b) *Com2Cand* (c) *Don2Com*

(d) *Don2Com* (e) *Auth2Conf* (f) *Auth2Conf*

(g) *Blognet* (h) *Postnet*

Figure 5: Illustration of the Egonet λ_w Power Law (*ELWPL*) and dominant heavy link anomaly *DominantPair*. Top anomalies are marked with triangles and labeled. See Section 5.2.2 for detailed discussions for each dataset and Fig.1 for an illustrative example from *Com2Cand*.

conference (ICRA). "*Alberto-Sangiovanni Vincentelli*" published 279 papers to 45 distinct conferences, with 76 of his papers in DAC. On the *Conferences* plot, "*Programming Languages and Their Definition*" has 21 papers from 6 authors, with 16 papers by one particular author ("*Hans Bekic*").

5.1 Scalability

Major computational cost of our method is in feature extraction. In particular, computing those features, such as the total number of edges and total weight, for the egonets is the bottleneck as one needs to find the induced 1-step neighborhood subgraphs for all nodes in the network.

The problem of finding the number of edges in the egonet of a given node can be reduced to the

problem of triangle counting. One straightforward *listing* method for local triangle counting is the *Node-Iterator* algorithm. *Node-Iterator* considers each one of the N nodes and examines which pairs of its neighbors are connected. Time complexity of the algorithm is $O(Nd_{max}^2)$. Approximate streaming algorithms for local triangle counting can be applied to reduce the time complexity to $O(E \log N)$ with space complexity $O(N)$ [7].

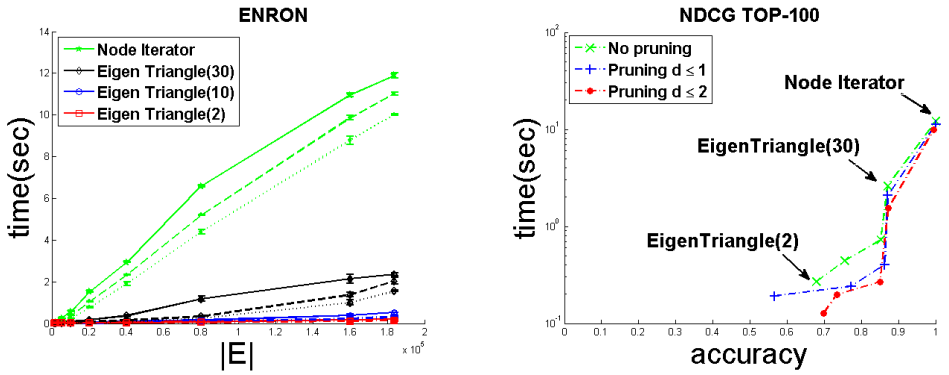


Figure 6: (a) Computation time of computing egonet features vs. number of edges in *Enron*. Effect of pruning degree-1 and degree-2 nodes on computation time of counting triangles. Solid(–): no pruning, dashed(−−): pruning $d \leq 1$, and dotted(. . .): pruning $d \leq 2$ nodes. Computation time increases linearly with increasing number of edges, whereas it decreases with pruning. (b) Time vs. accuracy. Effect of pruning on accuracy of finding top anomalies as in the original ranking before pruning. New rankings are scored using Normalized Cumulative Discounted Gain. Pruning reduces time for both *Node-Iterator* and *Eigen-Triangle* for different number of eigenvalues while keeping accuracy at as high as ~ 1 and $\sim .9$, respectively.

In our experiments, we use *Eigen-Triangle*, proposed in [35], which uses eigenvalues/vectors to approximate the number of paths of length three, i.e. local triangle counts, without performing actual counting. We compare its performance to *Node-Iterator*, which gives exact counts. To improve speed even more, we propose pruning low degree nodes. Notice that removing degree-1 nodes has no effect on the number of triangles in the graph. We also try removing degree-2 nodes; this would remove some triangles but hopefully would not change the relative number of triangles across nodes drastically and still reveal similar outliers.

In Figure 6a, we show computation time for *Node-Iterator* (green), and for *Eigen-Triangle* using 2(red), 10(blue), and 30(black) eigenvalues versus graph size in terms of number of edges for Enron($E \simeq 180K$). Solid(–), dashed(−−), and dotted(. . .) lines are for no pruning, after pruning $d \leq 1$, and $d \leq 2$ nodes, respectively. We empirically note that computation time grows linearly with increasing graph size and also reduces with pruning. (Experiments ran on a Pentium class workstation, with 16GB of RAM, running Linux Fedora Core. To account for possible variability due to system state, each run is repeated 10 times and execution time results are averaged. Error bars show the variance across repeated runs.)

Pruning low-degree nodes and using *Eigen-Triangle* reduces computation time, however, they only provide approximate answers. In order to quantify their accuracy, we compare the rank list of

outliers returned by *Node-Iterator* to the rank list of outliers returned by each approximate method. To measure how rankings changed with approximation compared to the original rankings, we use Normalized Dicounted Cumulative Gain(NDCG) which is prevailingly used in Information Retrieval for measuring the effectiveness of search engines. The premise of NDCG is that highly ranked items in the original list appearing lower in the approximated list are penalized logarithmically proportional to their positions in the approximated list.

Figure 6b shows time vs. NDCG scores for *Eigen-Triangle* using 2, 5, 10 and 30 eigenvalues, and also *Node-Iterator* for top k anomalies. For brevity, we only show ranking scores for k=100. *, +, and o symbols represent no pruning, pruning $d \leq 1$, and $d \leq 2$ nodes, respectively. We notice that while reducing computation time, pruning low degree nodes as well as using *Eigen-Triangle* keeps the accuracy at as high as~$.9$ for *Eigen-Triangle*(30), and above 0.55 for *Eigen-Triangle*(2).

6 Conclusion

This is one of the few papers that focus on anomaly detection in graph data, including weighted graphs. The major contributions are the following:

1. Proposing to work with "egonets", that is, the induced sub-graph of the node of interest and its neighbors; we give a small, carefully designed list of numerical features for egonets.
2. Discovery of new patterns that egonets follow, such as patterns in density (Obs.1: *EDPL*), weights (Obs.2: *EWPL*), principal eigenvalues (Obs.3: *ELWPL*), and ranks (Obs.4: *ERPL*). Proof of Lemma 1, linking the *ERPL* to the *EWPL*.
3. `OddBall`, a fast, hybrid, unsupervised method to detect abnormal nodes in weighted graphs. Possible approximations in feature extraction that provides speed-up, keeping accuracy at as high as~$.9$.
4. Experiments on real graphs of over 1M nodes, where `OddBall` reveals nodes that indeed have strange or extreme behavior.

Future work could generalize `OddBall` to time-evolving graphs, where the challenge is to find patterns that neighborhood sub-graphs follow and to extract features incrementally over time.

References

[1] Charu C. Aggarwal and Philip S. Yu. Outlier detection for high dimensional data. In *SIGMOD*, pages 37–46, 2001.

[2] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. RTM: Laws and a recursive generator for weighted time-evolving graphs. In *ICDM*, Pisa, Italy, December 2008.

[3] Reka Albert, Hawoong Jeong, and Albert-Laszlo Barabasi. Diameter of the world wide web. *Nature*, (401):130–131, 1999.

[4] Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. A linear method for deviation detection in large databases. In *KDD*, pages 164–169, 1996.

[5] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley and Sons, Chichester, New York, 1994.

[6] Stephen Bay, Krishna Kumaraswamy, Markus G. Anderle, Rohit Kumar, and David M. Steier. Large scale detection of irregularities in accounting data. In *ICDM*, 2006.

[7] Luca Becchetti, Paolo Boldi, Carlos Castillo, and Aristides Gionis. Efficient semi-streaming algorithms for local triangle counting in massive graphs. In *KDD*, pages 16–24, 2008.

[8] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. In *SIGMOD*, pages 93–104, 2000.

[9] A. Z. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. L. Wiener. Graph structure in the web. *Computer Networks*, 33:309–320, 2000.

[10] Deepayan Chakrabarti. Autopart: Parameter-free graph partitioning and outlier detection. In *PKDD*, pages 112–124, 2004.

[11] Deepayan Chakrabarti and Christos Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1), 2006.

[12] Duen Horng Chau, Shashank Pandit, and Christos Faloutsos. Detecting fraudulent personalities in networks of online auctioneers. *PKDD*, 2006.

[13] Amitabh Chaudhary, Alexander S. Szalay, and Andrew W. Moore. Very fast outlier detection in large multidimensional data sets. In *DMKD*, 2002.

[14] Ping Chen, Rebecca B. Buchheit, Jr, and Sue Mcneil. Web-vacuum: Web-based environment for automated assessment of civil infrastructure data. *Journal of Computing in Civil Engineering*, 19(2):137–147, 2005.

[15] Tamraparni Dasu and Theodore Johnson. *Exploratory Data Mining and Data Cleaning*. Wiley-Interscience, May 2003.

[16] William Eberle and Lawrence B. Holder. Detecting anomalies in cargo shipments using graph properties. In *ISI*, pages 728–730, 2006.

[17] William Eberle and Lawrence B. Holder. Discovering structural anomalies in graph-based data. In *ICDM Workshops*, pages 393–398, 2007.

[18] Amol Ghoting, Matthew Eric Otey, and Srinivasan Parthasarathy. Loaded: Link-based outlier and anomaly detection in evolving data sets. In *ICDM*, 2004.

[19] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: An efficient clustering algorithm for large databases. *Inf. Syst.*, 26(1):35–58, 2001.

[20] D. Hawkins. Identification of outliers. *Chapman and Hall*, 1980.

[21] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice Hall, 1998.

[22] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, 1990.

[23] Edwin M. Knorr and Raymond T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB*, pages 392–403, 1998.

[24] Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detection. In *KDD*, pages 157–166, 2005.

[25] Jure Leskovec, Mary McGlohon, Christos Faloutsos, Natalie Glance, and Matthew Hurst. Cascading behavior in large blog graphs: Patterns and a model. In *Society of Applied and Industrial Mathematics: Data Mining*, 2007.

[26] Chao Liu, Xifeng Yan, Hwanjo Yu, Jiawei Han, and Philip S. Yu. Mining behavior graphs for "backtrace" of noncrashing bugs. In *SDM*, 2005.

[27] Mary McGlohon, Leman Akoglu, and Christos Faloutsos. Weighted graphs and disconnected components: Patterns and a model. In *ACM SIG-KDD*, Las Vegas, Nev., USA, August 2008.

[28] H. D. K. Moonesinghe and Pang-Ning Tan. Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools*, 17(1), 2008.

[29] Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. In *VLDB*, pages 144–155, 1994.

[30] Caleb C. Noble and Diane J. Cook. Graph-based anomaly detection. In *KDD*, pages 631–636, 2003.

[31] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW '07*, New York, NY, USA, 2007.

[32] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B. Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *ICDE*, pages 315–, 2003.

[33] Mukund Seshadri, Sridhar Machiraju, Ashwin Sridharan, Jean Bolot, Christos Faloutsos, and Jure Leskovec. Mobile call graphs: Beyond power-law and lognormal distributions. In *ACM SIGKDD*, 2008.

[34] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graph. *ICDM*, November 27-30 2005.

[35] Charalampos Tsourakakis. Fast counting of triangles in large real networks, without counting: Algorithms and laws. In *ICDM*, 2008.

[36] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. In *SIGMOD*, pages 103–114, 1996.