Chaotic Mining: Knowledge Discovery Using the Fractal Dimension (Extended Abstract)

Daniel Barbará
George Mason University *
Information and Software Engineering Department
Fairfax, VA 22303
dbarbara@gmu.edu

1 Introduction

Nature is filled with examples of phenomena that exhibit seemingly chaotic behavior, such as air turbulence, forest fires and the like. However, under this behavior it is almost always possible to find *self-similarity*, i.e. an invariance with respect to the scale used. The structures that appear as a consequence of self-similarity are known as *fractals* [12].

Fractals have been used in numerous disciplines (for a good coverage of the topic of fractals and their applications see [14]). In the database arena, fractals have been sucessfully used to analyze R-trees [6], Quadtrees [5], model distributions of data [7] and selectivity estimation [3].

Fractal sets are characterized by their fractal dimension. In truth, there exists an infinite family of fractal dimensions. By embedding the dataset in an n-dimensional grid whose cells have sides of size r, we can compute the frequency with which data points fall into the i-th cell, p_i, and compute D_q, the generalized fractal dimension [8, 9], as shown in Equation 1.

$$D_q = \frac{1}{q-1} \frac{\log \sum_i p_i^q}{\log r} \qquad (1)$$

Among the dimensions described by Equation 1, the *Hausdorff fractal dimension* ($q = 0$), the *Information Dimension* ($\lim_{q \to 1} D_q$), and the *Correlation dimension* ($q = 2$) are widely used. The Information and Correlation dimensions are particularly useful for data mining, since the numerator of D_1 is Shannon's entropy, and D_2 meassures the probability that two points chosen at random will be within a certain distance of each other. Changes in the Information dimension mean changes in the entropy and therefore point to changes in trends. Equally, changes in the Correlation dimension mean changes in the distribution of points in the dataset.

Fast algorithms exist to compute these dimensions. (FD3, an implementation based on the ideas described in [11] can be obtained from [13] and other software repositories.)

In Section 2 we show examples of techniques that employ the fractal dimension to mine large datasets.

*This work has been supported by NSF grant IIS-9732113

2 Examples

In this section we present some examples of techniques that use the fractal dimension as a mining tool. We are currently working in developing efficient, scalable algorithms for all these methods.

2.1 Event anomalies in time series

A time series is a temporal sequence of measured values, which mark the occurence of an event such as a stock price changing, an electroencephalographic potential measurement, or a TCP connection occurrence. In many cases, the pattern of event occurrences is self-similar (e.g., traffic in a network [10]) and a deviation from this pattern may indicate the prescence of an anomalous behavior.

As an example, consider a time series that shows the occurences of *half-open* TCP connections in a network. A TCP connection is characterized as half-open [17] if one end has closed or aborted the connection without the other end knowing. Normally, these connections are caused by host crashing or by errors incured by software or by users. However, intruders use half-open connections to invade networks in an attack that is known commonly as *network spoofing* (an example of such attack can be found in [16]. This attack is characterized by the sudden appearance of numerous half-open connections within a small amount of time.

How can we use the fractal dimension to detect this anomaly? We have experimented with a time series where every δ seconds, a measurement reflects the number of half-open connections made in that period. We have observed that the pattern of half-open connections in the time series while the network is not being attacked is self-similar (exhibiting a fractal dimension close to 1). However, in data traces that contain spoofing attacks, self-similarity breaks down during the attack. This can be detected rolling a moving window of size $\Delta = k \times \delta$, where k is an integer, over the time series and computing the fractal dimension of the data covered by the window. The complexity of computing the fractal dimension is $O(n \log(n)$, where n is the number of data points in the window, determined by the scale and size of the window used. The resulting dimension shows a drastic decrease when the region of the attack is entered. We have observed that the Correlation Dimension drops from 0.98 when the window contains attack-free data to half that value when the window covers the data where the attack took place. [1] This suggest that the fractal dimension is a powerful, robust indicator of anomalies in this time series. We are currently trying to define other events that give way to different time series for network traffic, and to use the fractal dimension over those series to uncover other types of attacks. An obvious candidate, for example, is the event of connecting to the password port in the FTP service (port 21). The pattern of accesses to this port is, under normal conditions, self-similar; however, during a *password guessing* attack, the number of connection attempts to this port is noticeably bigger, disrupting the pattern and altering the fractal dimension. The fractal dimension in these cases serves as a more robust detection method than just counting connections in a period of time. Doing the latter, it is not trivial to set up thresholds to separate normal from suspicious behavior.

2.2 Self-similarity in association rules

Association rules [1] are rules of the form $X \longrightarrow Y$ where X and Y are sets of attribute-values, with $X \bigcap Y = \emptyset$ and $\|Y\| = 1$. The set X is called the antecedent of the rule while the item Y is called consequent. For example, in a market-basket data of supermarket transactions, one may find that customers who buy *milk* also buy *honey* in the same transaction, generating the rule $milk \longrightarrow honey$. There are two parameters associated with a rule: *support* and *confidence*. The rule $X \longrightarrow Y$ has *support* s in the transaction set T if $s\%$ of transactions in T contain $X \cup Y$. The rule $X \longrightarrow Y$ has *confidence* c if $c\%$ of transactions in T that contain X also contain Y.

With all their importance and practical applicability, association rules say nothing about the way the common occurrence of attribute-values occur in time. For instance, if the previous rule $milk \longrightarrow honey$ has a support of 0.7, all we know is that the out of all the transactions analyzed in the dataset, 70 % of them contain both *milk* and *honey*. The rule says nothing about the distribution of transactions that originated this support. We do not know whether all the customers bought milk and honey over mostly during a short period of time (as a response to a promotion, perhaps), or the buying of these two products together responds to a pattern that is the same, regardless of the scale choosen (self-similar). This information is indeed valuable to the supermarket: knowing that this rule is seasonal, or the reponse to a promotion leads to different decisions than knowing that the rule responds to a more "regular" pattern.

We are interested in using the fractal dimension to analyze how association rules occur in a dataset. Naively, it would be a simple task to roll a window over the data to perform a similar analysis that we suggested in Section 2.1. However, this is wasteful, since it would lead to even more passes over the data than the ones required by the implementation of the association rules mining algorithm. Therefore, we are interested in finding efficient ways of performing the fractal dimension analysis while the association rules algorithm is taking place. Consider the a-priori algorithm described in [2]. Two things must be done to analyze self-similarity while discovering itemsets with high support. First, as a k-itemset is under consideration, and we are scanning the dataset to compute its support, we should also roll a window and compute the fractal dimension of the occurrence of this rule as we go through the data. Secondly, and more difficult, if this itemset is found to have a lot of support, enough information about the fractal dimension of this rolling window should be kept to be used when processing the $k+1$ extensions of this itemset in the next iteration of the algorithm.

2.3 Analyzing patterns in datacubes

Investigating the patterns followed by many "special" values in a datacube can lead to important discoveries. For instance, an analyst can be interested in uncovering the trends of the null cells in a cube (i.e., cells for which there is no aggregate). By doing this, he or she may discover that some null cells are caused by special factors. For instance, the lack of sales of a product in a particular store during certain months may be anomalous with respect to other stores: uncovering this fact can lead to important knowledge about the store's operation.

Again, a similar technique to the ones used in Sections 2.1 and 2.2 can be used to detect anomalous patterns. A hyperdimensional window can be rolled

[1] The Information dimension suffers a similar drop.

over the datacube, computing the fractal dimension of the set of null cells contained in the window. Drastic changes in the fractal dimension should point out to anomalous trends. Probably, after an anomalous window of cells has been identified, it would be desirable to reduce the window size incrementally and rolled over the anomalous region to try to isolate the cells that are causing the change in trend. An efficient algorithm to roll these windows without excesive I/O is needed.

2.4 Incremental clustering using the fractal dimension

Incremental clustering techniques are needed to deal with large datasets. In [4], the authors describe the Extended K-Means algorithm, an incremental technique that requires only one scan of the database. At every step of the algorithm, the data tuples already processed are either retained (as outliers), reduced via compression and summarized, or discarded after updating the description of the clusters. The algorithm is an extension of the classic K-Means algorithm [15], operating over data and statistics of previously reduced data.

We plan to implement a variation of Extended K-Means, that uses fractal dimension to characterize clusters and decide point membership. Running K-Means over a sample of the dataset that fits in memory, we can obtain a first clustering. After computing the fractal dimension of each cluster, we can use the dimension to determine where a new tuple should be included, by comparing the dimension one obtains in a cluster when adding this new member(s). If the change is very substantial, this is probably not a good cluster to put the new member(s) in. We plan to keep also a set of retained tuples (outliers) and a set of representatives of a cluster to perform the next iteration with.

3 Conclusions

The fractal dimension can be a powerful parameter to uncover anomalous patterns in datasets. We have presented, by via of examples, a suite of techniques based in the use of fractal dimension that can significantly help analysts uncover valuable information from large datasets. We are currently working in implementing these algorithms.

References

[1] R. Agrawal, T. Imielinski, , and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD Conference on Management of Data*, Washington D.C., may 1993.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast Discovery of Association Rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, 1996.

[3] A. Belussi and C. Faloutsos. Estimating the Selectivity of Spatial Queries Using the 'Correlation' Fractal Dimension. In *Proceedings of the International Conference on Very Large Data Bases*, pages 299–310, September 1995.

[4] P.S. Bradley, U. Fayyad, and C. Reina. Scaling Clustering Algorithms to Large Databases. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, New York City*, August 1998.

[5] C. Faloutsos and V. Gaede. Analysis of the Z-ordering Method Using the hausdorff Fractal Dimension. In *Proceedings of the International Conference on Very Large Data Bases*, pages 40–50, September 1996.

[6] C. Faloutsos and I. Kamel. Relaxing the Uniformity and Independence Assumptions, Using the Concept of Fractal Dimensions. *Journal of Computer and System Sciences*, 55(2):229–240, 1997.

[7] C. Faloutsos, Y. Matias, and A. Silberschatz. Modeling Skewed Distributions Using Multifractals and the '80-20 law'. In *Proceedings of the International Conference on Very Large Data Bases*, pages 307–317, September 1996.

[8] P. Grassberger. Generalized Dimensions of Strange Attractors. *Physics Letters*, 97A:227–230, 1983.

[9] P. Grassberger and I. Procaccia. Characterization of Strange Attractors. *Physical Review Letters*, 50(5):346–349, 1983.

[10] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the self-similar nature of ethernet traffic. *IEEE Transactions on Networking*, 2(1):1–15, February 1994.

[11] L.S. Liebovitch and T. Toth. A Fast Algorithm to Determine Fractal Dimensions by Box Countig. *Physics Letters*, 141A(8), 1989.

[12] B.B. Mandelbrot. *The Fractal Geometry of Nature.* W.H. Freeman, New York, 1983.

[13] John Sarraille and P. DiFalco. FD3. http://tori.postech.ac.kr/softwares/.

[14] M. Schroeder. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise.* W.H. Freeman, New York, 1991.

[15] S.Z. Selim and M.A. Ismail. K-Means-Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1), 1984.

[16] T. Shimomura and J. Markoff. *Take-Down; The Pursue and Capture of Kevin Mitnick.* Hyperion Books, February 1996.

[17] W.R. Stevens. *TCP/IP Illustrated, Volume 1.* Addison-Wesley, 1994.