SHARD: A Framework for Sequential, Hierarchical Anomaly Ranking and Detection

Jason Robinson[1], Margaret Lonergan[1], Lisa Singh[1],
Allison Candido[1], and Mehmet Sayal[2]

[1] Georgetown University, Washington, DC 20057, USA
[2] Hewlett Packard, Palo Alto, CA 94304, USA

Abstract. This work explores unsupervised anomaly detection within sequential, hierarchical data. We present a flexible framework for detecting, ranking and analyzing anomalies. The framework 1) allows users to incorporate complex, multidimensional, hierarchical data into the anomaly detection process; 2) uses an ensemble method that can incorporate multiple unsupervised anomaly detection algorithms and configurations; 3) identifies anomalies from combinations of categorical, numeric and temporal data at different conceptual resolutions of hierarchical data; 4) supports a set of anomaly ranking schemes; and 5) uses an interactive tree hierarchy visualization to highlight anomalous regions and relationships. Using both synthetic and real world data, we show that standard anomaly detection algorithms, when plugged into our framework, maintain a high anomaly detection accuracy and identify both micro-level, detailed anomalies and macro-level global anomalies in the data.

Keywords: Anomaly detection framework, multi-resolution anomalies, ensemble method

1 Introduction

Anomaly detection has many applications, including fraud detection, outbreak identification, and data scrubbing [13] [4]. Each of these domains contains its own semantic relationships, many of which can be modeled as hierarchical. In this paper, we present a framework that allows users to identify anomalies across different levels of these hierarchical structures. For example, in fraud detection, users may be interested in detecting fraudulent behavior across different time granularities (weeks, month, years) or across different locations (neighborhood, city, state). In this case, both time and location are different examples of semantic hierarchies that can be used to identify recurring or aggregated anomalies. Figure 1 shows an example of a sequential, time based hierarchy that we will refer to as an *anomaly tree*. Each level of the anomaly tree represents a different granularity of time. By viewing these different semantic groups of data hierarchically, users can better understand how anomalies propagate through different sequential, hierarchical relationships associated with their applications. Are anomalies scattered or recurring? Are some days, months, or years more anomalous than others?

In this work, we propose $SHARD$, a flexible framework for **S**equential, **H**ierarchical, **A**nomaly, **R**anking, and **D**etection that supports incorporation of hierarchical semantics across numeric and categorical data into unsupervised, anomaly detection and ranking. This work makes the following contributions. First, we present system and design

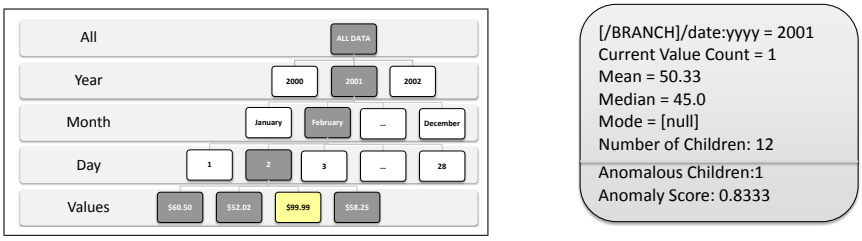

Fig. 1. Anomaly tree example and individual node statistics

considerations for developing a general framework for hierarchical anomaly detection. These considerations lead to the decoupling of data formats, outputs, and the definition of 'anomalous' for a given use case. The second contribution is the framework itself, which allows single or multiple anomaly detectors to work together. Most importantly it allows domain experts to drive the anomaly detection process by scripting meaningful, hierarchical relationships between the attributes. Finally, we present experiments on synthetic and real world data sets that show similar performance of detailed, micro-level anomaly detection when compared to the baseline detector performance without the framework; the experiments also demonstrate high-order macro-level anomalies that would completely escape the expert's view without the framework.

The remainder of this paper is organized as follows. Section 2 presents related literature. Section 3 presents background concepts. Our framework is presented in section 4, followed by experimental results in section 5, and the conclusions in section 6.

2 Related Literature

A large body of literature on anomaly detection exists. For a detailed survey of anomaly detection techniques, we refer you to [4] and [13].

Anomaly detection frameworks: A few anomaly detection frameworks have been proposed in the literature. For example, Chandola [3] proposes a Reference Based Analysis (RBA) framework for analyzing anomalies with numeric and categorical data in sequences and time series. While RBA offers summary visualizations, it does not offer the multi-resolution evaluations, the interactive visualizations, or the plugin detection and ranking algorithms that our framework does. Nemani *et al.* [12] propose a framework for detecting anomalies in spatial-temporal data. This framework supports plugin detection algorithms; yet, it does not appear to support visualization of multi-granular time series, nor is it clear how customizable other aspects of this framework are.

Anomaly detection algorithms: A number of approaches for anomaly detection of time series data exist [5], [8], [10]. Antunes and Oliveira [5] transform the time series into forms that can use standard approaches for anomaly detection. Keogh, Lonardi, and Chiu [10] evaluate the frequency of substrings in a time series and compare the resulting distribution to a baseline time series. Li and Han [11] explore anomaly detection in multidimensional time series data, identifying the top-k outliers for each detection method and iteratively pruning these sets until a uniform set of anomalies is discovered. All of these sequential anomaly detection algorithms focus on single resolution

anomaly detection. Instead, this work focuses on a framework that supports integration of many algorithms across multiple resolutions.

Joslyn and Hogan [9] explore similarity metrics in directed acyclic graphs and other hierarchical structures. Their work can be utilized to visualize and find anomalies in ontologies. While the ideas concerning semantic hierarchies that we present are implicit in Joslyn and Hogan's work, their focus is entirely on similarity metrics in these tree structures and not on the full implementation of an anomaly detection framework.

3 Hierarchical Anomalies

Suppose we are given a data set D, containing a set of attributes or features, $F = \{F_1, F_2, \ldots, F_m\}$, where m is the number of features in D. Each feature contains an ordered list of n values, $F_i = [v_1, v_2, \ldots, v_n]$. We define an anomaly, A, as a data point or set of data points that deviate or behave differently than the majority of *comparison data*, where the comparison data represents values for one or more features in D. We purposely define an anomaly broadly since the type of deviation of interest can vary depending on the data type (numeric, categorical, etc.) and/or the domain characteristics.

Even though our framework can handle any data that can be represented sequentially and hierarchically, including natural language (document, sentences, words, syllables, letters) and genetic sequences (DNA, genes, proteins), for ease of exposition and ubiquity of data, we focus on time series data and time anomaly trees. In this case, data values exists for each feature in the data set at n time points. We also define a set of semantic resolutons $r = \{r_1 ... r_h\}$, where each resolution represents a different semantic grouping for data in D. The semantic groupings for our example in figure 1 are day, month, and year, $r = \{day, month, year, all\}$. These semantic groupings can then be used as the basis for creating a time anomaly tree T of height h, where $h = 4$ for our example. The resolutions tell us the granularity of data associated with each node in a particular level of the tree. The leaf nodes contain statistics about data values at resolution r_1, the day resolution in our example. The parent nodes of the leaf nodes contain statistics about the data values at resolution r_2, e.g. the month resolution, and so on. Given this anomaly tree, we define a hierarchical anomaly $A(n_l)$ to be a node n at level l that deviates significantly from other nodes (or a subset of other nodes) at level l in the anomaly tree, where deviation is measured by one or more detectors selected by the user and significance is algorithm specific.

For example, in a stock data domain, a single company can be considered anomalous if it has an unlikely, sudden surge and subsequent drop in price, if it has an unlikely surge in price that is maintained for some sustained duration, e.g. month, before dropping back to normal, if daily behavior differs drastically from other companies', or if the company manifests a combination of these unusual behaviors. The specific type of behavior identified depends on the detectors and rankers specified by the user.

4 Anomaly Detection Framework

Our high level algorithm for anomaly tree construction and annotation is presented as Algorithm 1. The input to the algorithm is the data (D), an ontology template that

specifies the semantic relations of interest (τ), the anomaly detectors of interest (A), and an anomaly ranker (R). Using this information, the framework builds an anomaly tree by assigning data values to the nodes and updating the node summary statistics according to the ontology template, runs different anomaly detectors on the nodes of this tree to obtain a set of anomaly scores for each node, and ranks the anomalies in the tree by computing a score based on criteria such as the level of agreement between the anomaly detectors and the anomaly scores of the child nodes. The resulting tree is then used for an interactive tree visualization that can be analyzed by the user. The remainder of this section describes the framework and different design decisions.

Algorithm 1 Anomaly tree construction and annotation

INPUT: Template τ, Anomaly Detectors A, Ranker R, Data D
OUTPUT: T

function T = BUILD_TREE(τ, D)
function IDENTIFY_ANOMALIES(T, A)
function RANK_ANOMALIES(T, R)
return T

4.1 Ontology Template

The ontological tree template not only decides the hierarchy of where and how feature values are organized and propagated, but also determines how the detectors evaluate nodes. Specific considerations are 1) the range of nodes that maintain summary statistics for the detectors to analyze, 2) normalizing or scaling of multivariate combinations, and 3) sorting of temporal or ordinal features. Table 1 shows an example ontology template and the resulting anomaly tree. The XML template describes an application that attempts to find three different semantic hierarchies based on time, industry, and employee education.

4.2 Anomaly Tree Structure

The anomaly tree T generated by the ontology template consists of multiple node types.

Definition 1. *The **leaf nodes** at the lowest level of the tree contain data values. Data from these nodes are aggregated and propagate information to the remaining levels of the tree. **Semantic grouping nodes** are non-leaf nodes that are associated with a feature and group children nodes according to the feature values. **Branching nodes** create a branch of nodes to be evaluated for anomalies. These nodes determine how the child values are evaluated and propagated through T. The propagation of leaf node values stops at the branching node.*

Each node type handles individual data values differently. Semantic grouping nodes split on every new value of the attribute specified in the ontology template. Branching nodes are not associated with a value. Instead they store summary statistics of all descendant nodes and tell the detectors whether or not to search for anomalies in a particular branch. The branch creation process creates a root node and a set of children nodes, where each child corresponds to a branching node based on attribute values specified in the ontology template. For example, in the tree path industry/company/[PRICE]/Price/yyyy/

```
<DataTreeTemplate>
   <Node attribute="industry">
      <Node attribute="company">
         <Node attribute="edu" branch="EDU[c]"
               propagateValues="True" >
            <Node attribute="employee">
               <Leaf attribute="edu" />
            </Node>
         </Node>
      <Node attribute="date" step="2"
            branch="PRICE"
            propagateValues="True" >
         <Node attribute="date" step="0">
               <Node attribute="date"
                                 step="1" >
                  <Leaf attribute="price"/>
               </Node>
         </Node>
      </Node>
   </Node>
   <Node attribute="edu" branch="EDU_PR"
         propagateValues="True">
      <Node attribute="date" step="2" >
         <Leaf attribute="price"/>
      </Node>
   </Node>
</DataTreeTemplate>
```

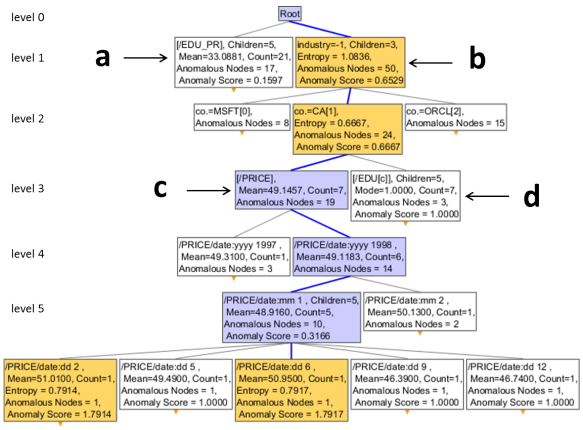

Table 1. XML template and anomaly tree for XML template. Nodes a, c and d are examples of branching Nodes. Node b is a semantic grouping node, as are all nodes below c and d. Node d also specifies the data propagation to be categorical.

mm/dd/price[3], all nodes are grouping nodes except for [PRICE] and the leaf node price data. The leaf nodes propagate their values upward to the top branching node, which means that every parent node is a summary of all of its child nodes. The XML example has two leaf attribute values, price and education that anomalies will be calculated for.

The branch EDU[c] creates a branching node that maintains summary statistics (e.g. $mode$) of the categorical datatype education for each employee in the semantic grouping node $company$, so that we can determine the most frequent level of education per company. Likewise, the parent semantic grouping node $industry$ allows the researcher to also evaluate levels of education across industries. Branching node [EDU_PR] aggregates prices by the average levels of education across all companies.

Table 1 also shows portions of the anomaly tree for the specified XML template. In this example, there is only one industry, technology, under which there are three nodes, one for each of the companies.[4] The arrows at the bottom of nodes indicate nodes that can be expanded to show their children. As the figure illustrates, the anomaly statistics are populated throughout the tree and data statistics from the leaf nodes under a branching node are aggregated as they are pushed up to the branching node, populating the intermediary nodes along the way. Each intermediary node maintains summary statistics of its children nodes. The month level node for the price attribute, for example, maintains the average price for all the children day nodes. Other statistics are also calculated, including median, mode, standard deviation, and entropy.

4.3 Baseline Anomaly Detectors

The anomaly detectors use the anomaly tree, T, to determine the degree of anomalousness of each node in T. This is accomplished by running each user specified anomaly

[3] The XML template in table 1 uses the keyword 'step' to identify which time steps to split on.
[4] See http://cs.georgetown.edu/∼singh/SHARD/ for larger figures, data sets, and source code.

detection algorithm, e.g. statistical significance or entropy, for each element in the tree. Along with the basic detectors, SHARD includes an ensemble detector that combines the detection results of the individual detectors using a weighted voting algorithm, where the weights are prespecified by the user. Once the anomaly scores are computed by the different detectors, the tree nodes are annotated with this additional information. This is also illustrated in Table 1.

In order to identify an anomaly, a data value must be compared to other data values. When evaluating a particular node in T, we use neighboring nodes as comparison data. However, how these nodes are used differs depending on the particular anomaly detection algorithm. For example, table 1 shows the current node under consideration to be day 6 of month 1 (January) of year 1998 of CA, Inc. The options for comparison data for this example include: 1) all immediate sister nodes, all nodes in January for this year and company; 2) all prices for all months under the same company; 3) all prices for all months and companies; 4) all the January 6ths' for the current year across all companies; and 5) the averages of the previous days or months. The SHARD framework includes three parameterized defaults: 1) all local siblings (sister) nodes; 2) all nodes at the same tree height for the same attribute; and 3) previous nodes at the same tree height for the same attribute. Other options can be specified at configuration time and new options are straightforward to integrated into framework.

4.4 Ranking Anomalies

Once all of the detectors have evaluated the nodes in T, the algorithm then runs a user specified ranking method to assign an overall anomaly score to each node. The ranking procedure can compute the anomaly score based on any of the following criteria: 1) the anomaly scores provided by different detectors for a particular node; 2) the percentage of detectors that found a particular node anomalous; 3) the priority of the detectors that found the node to be anomalous; 4) the percentage of child nodes that were found to be anomalous; 5) the importance of the level of granularity in which the anomalous node occurs; and 6) whether anomalies occur in other parallel branches at the same granularity. Our intuition is that the level of anomalousness depends on the domain priorities, objectives and definitions of comparison data. Therefore, we incorporate a tunable ranker that can be adjusted to these considerations. Ranking based on the percentage of anomalous children is the default ranker in SHARD, although we also provide other ranking procedures that combine different subsets of the mentioned factors.

4.5 Anomaly Tree Visualization

SHARD uses the SpaceTree [14] hierarchical visualization application to highlight the most anomalous nodes based on a color heat map. SpaceTree reads in XML and displays an interactive tree of variable depth and width. This interactive software enables users to expand the entire tree or focus on subtrees of different branches of the full tree while hiding other subtrees. Doing this helps the user see where anomalies occur across multiple resolutions. Because our framework is customizable, any amount of detail can be displayed for each node including ranking scores, statistical summaries, individual detector results, and raw data. This interactive visualization supports both an overview

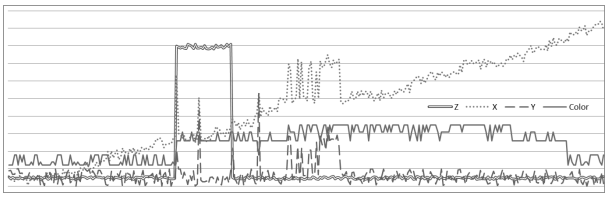

Fig. 2. One year of synthetic time series data

and a detailed view, allowing for a more comprehensive analysis of the anomalies. Most of the tree images in this paper were generated using SpaceTree.

5 Empirical Evaluation

In this section, we evaluate our framework on synthetic and real world data sets. Our evaluation of the SHARD framework focuses on detection accuracy and anomalies discovered. Specifically, we compare the accuracy of the detectors outside our framework with the same detectors within the SHARD framework and show that the overall accuracy is generally maintained, while also offering bigger picture insights. We also discuss these insights at different levels of the anomaly tree and demonstrate the flexibility of our framework.

We experimented with four standard anomaly detection algorithms in our framework: 1) the Shewhart algorithm [1], which flags anomalies that are x standard deviations away from the mean; 2) the Cumulative Sum (cusum) algorithm, which tracks the mean of all previous elements and compares the values to the current element; 3) entropy (applied to anomalies as described in [7]); and 4) a thresholding version of Bruenig *et. al*'s [2] Local Outlier Factor (LOF).

The ranking algorithm used in all of the experiments is *RankerA*. This ranker first evaluates the children nodes. If at least half are anomalous, the current (parent) node is also considered anomalous. Otherwise, the sum of all anomaly scores, one from each detector, of a node is divided by the number of children nodes.

5.1 Synthetic Data Experiments

For this analysis, we generated three time series with a numeric data value for each day over a six year period, and one categorical times series. Figure 2 shows each of the numeric time series for a one year period. As illustrated in the figure, each time series has different properties and anomalies. Time series X increases in overall magnitude over time with burst anomalies for 200 random days, one random month of the year (this includes several of the random anomalous days), and one random year (this includes approximately 1/3 of its days being anomalous). Time series Y is similar except that the "normal" comparison values across all 6 years remain relatively steady. Like X, it contains randomly anomalous days, months and a year- most of which coincide with the anomalies in time series X. Time series Z is mostly independent of the other two time series and illustrates a plateau anomaly that starts and ends with anomalies found in X and Y. It contains the same anomalous month each year in which all values during

Detector	Attribute - Path	Precision	Recall
Shewhart	x - yyyy/mm/dd	75.3%	11.6%
	x - leaf	100.0%	8.9%
	y - yyyy/mm/dd	93.3%	12.5%
	y - leaf	100.0%	54.5%
	z - yyyy/mm	83.3%	45.5%
	z - leaf	3.3%	50.0%
	x,y,z - yyyy/dd	100.0%	2.2%
OVERALL		52.9%	12.7%
Entropy	x - yyyy	50.0%	100.0%
	x - yyyy/mm	12.5%	50.0%
	x - yyyy/mm/dd	29.6%	86.2%
	x - leaf	21.6%	63.0%
	y - yyyy	100.0%	100.0%
	y - yyyy/mm	60.0%	100.0%
	y - yyyy/mm/dd	29.1%	85.7%
	y - leaf	100.0%	100.0%
	z - yyyy/mm	83.3%	45.5%
	z - yyyy/mm/dd	0.4%	25.0%
	z - leaf	3.3%	50.0%
	color - yyyy	25.0%	100.0%
	color - yyyy/mm	6.7%	25%
	color - leaf	30.2%	95.0%
	x,y,z - yyyy	50.0%	100.0%
	x,y,z - yyyy/mm	100.0%	52.9%
	x,y,z - yyyy/mm/dd	33.9%	88.0%
OVERALL		28.3%	82.2%
LOF(1)	x - yyyy/mm/dd	92.0%	41.1%
	y - yyyy/mm/dd	100.0%	29.9%
LOF(3)	x,y,z - yyyy/mm/dd	98.1%	46.4%
OVERALL		95.6%	7.9%

Fig. 3. Single detectors

Detector	Parameters	Attribute	Precision	Recall
Shewhart	thresh=2	x	100.0%	8.9%
	"	y	100.0%	54.5%
	"	z	3.8%	50.0%
	"	color	n/a	n/a
	"	x,y,z	100.0%	10.3%
OVERALL			52.6%	24.2%
Entropy	thresh=1.2	x	21.6%	63.0%
	"	y	100.0%	100.0%
	"	z	3.8%	50.0%
	"	color	1.3%	47.8%
	"	x,y,z	21.8%	62.8%
OVERALL			20.5%	74.3%
LOF	k=15, dim=1	x	9.0%	0.5%
	"	y	100.0%	5.4%
	"	z	0.0%	0.0%
	"	color	n/a	n/a
	k=15,dim=3	x,y,z	85.0%	15.2%
OVERALL			74.6%	6.7%

(a) Baseline detectors

Detector	Attribute - Path	Precision	Recall
Shewhart Entropy LOF(1) LOF(3)	x - yyyy/mm/d	88.2%	43.3%
	x - leaf	100.0%	8.9%
	y - yyyy/mm/d	97.1%	30.4%
	y - leaf	100.0%	100.0%
	z - yyyy/mm	83.3%	45.4%
	z - leaf	3.3%	50.0%
	x,y,z - yyyy/mm/d	99.1%	46.4%
OVERALL		68.4%	25.6%

(b) Ensemble detectors

Fig. 4. Baseline and ensemble detectors

this month are consistent for this month, but still much higher than the normal day value for the rest of the year. At the individual day level, the only anomalies are the first day of this month when the values increase and the first of the following month when the values decrease back to normal. We also include a categorical attribute, $Color$, that is dependent on the season in the times series (during months *11,12, 1, 2, 3* {*blue, green, purple*}; *4, 5, 10* {yellow, orange}; and *6, 7, 8, 9*{red, orange, yellow}). An anomalous instance is an out-of-season color that corresponds with the Y anomalies' time points.

Our ontology template for this data set consists of 5 branches underneath the root. The first three simply aggregate each of the continuous variables by year, month and day independently:

[DATE-X]/yyyy/mm/dd/x, [DATE-Y]/yyyy/mm/dd/y, [DATE-Z]/yyyy/mm/dd/z

The fourth branch groups all three variables under each unique date:

[DATE-XYZ]/yyyy/mm/dd/x,y,z

Here, the time series are evaluated together, in the context of each other. In other words, the most anomalous time periods are when all three time series have anomalous behavior during the same time period. Note that there are parameters in the XML to normalize or scale multiple values under a single node. In this run, the configuration was set to Normalize. The final branch, [COLOR][c]/yyyy/mm/color organizes the categorical colors by month and year to capture anomalies in the context of different seasons. These various branches show the flexibility of the framework for handling different feature combinations that the user wants to investigate.

Figure 4(a) shows the scores of the baseline algorithms outside of our framework. The algorithms process each attribute individually and flag individual values as being anomalous, but give no indication of anomalous months or years. Figure 3 shows the results of the baseline algorithms within our framework. The overall scores are comparable with the record level scores outside of our framework in figure 4(a); however,

Detector	Attribute - Path	Precision	Recall
Shewhart	mm/dd/hh	21.7%	49.4%
	mm/dd/hh/c	25.0%	58.6%
	hh	100.0%	9.1%
	hh/Day/c	24.9%	43.2%
	hh/Day/c/id/c	25.0%	56.5%
OVERALL		24.7%	51.9%

Detectors	Attribute - Path	Precision	Recall
Shewhart Entropy LOF(1)	mm/dd/hh/c	72.2%	4.5%
	Day	50.0%	4.2%
	Day/c	62.8%	4.9%
	Day/c/id/c	60.7%	5.5%
OVERALL		63.9%	4.6%

Detector	Attribute - Path	Precision	Recall
Entropy	mm/dd/hh/c	72.2%	4.5%
	hh/Day	14.3%	58.3%
	hh/Day/c/id/c	60.1%	5.8%
OVERALL		39.5%	4.1%

Detector	Attribute - Path	Precision	Recall
LOF	hh/Day	50.0%	16.7%
	hh/Day/c	61.8%	4.9%
OVERALL		59.5%	1.3%

Table 2. Anomaly detectors on the CalIt2 dataset, (dd = day of month; Day = day of week)

a richer picture is gained using our framework: Shewhart now correctly identifies z's anomalous months with much higher accuracy, entropy performs well at nearly all resolutions of the anomaly tree, and LOF's recall is higher for most variables. Finally, figure 4(b) shows the results of the ensemble of these detectors. While the overall accuracy and precision is lower than the single detectors in the framework, the interior nodes of the tree have similar or better precision and accuracy results, demonstrating a potential benefit of a diverse set of detectors for hierarchical anomaly detection.

5.2 Event Attendance Data Results

We now consider an event data set, the CalIt2 dataset [6], for detecting anomalous events. This data set contains two observation time series (people flowing in and people flowing out of the building) over 15 weeks from July to November. There are 48 time points per day. The 'normal' behavior of this data set is a periodic, light flow of people going in and out of this building. When a conference is occurring, the flow increases for what is considered normal at that day and time, and an anomaly occurs.

Using the SHARD framework we specified two parallel branches in the ontology template, which offers two different views of the data. The first is Month/Day/Hour/Count - the intutive hierarchy. The second branch is Hour/DayOfWeek/Count/id/Count. This branch first establishes normal data behavior of the 24 hours of the day across the entire dataset, and then sub-aggregates the data by the day of the week and then the counts. So, it might establish that the average count for 9:00 am is 3.5 people, and the average for 9:00 am/Wednesday is 5.0 people. The next groupings id/Count, then establish counts based on individual records.

Inside the SHARD framework with this XML configuration, Shewhart with a threshold of 1 scores 24.7% precision, 51.9% recall on the anomaly tree nodes; Entropy with a threshold of 7.5 scores 39.5% precision, 4.1% recall; LOF where k=5 scores 59% precision and 1.3% recall; and these three detectors in an ensemble configuration score 63.9% precision and 4.6% recall. Outside of the SHARD framework Shewhard and Entropy perform comparably on the flat data (pr= 24.8%, re=56.3%, and pr=55.7%, re=5.4%, respectively), but LOF scores 0% precision and recall.

We offer a few observations. First, the 0 score of LOF outside of our system is probably due to at least k records with high counts that are not known events. As these

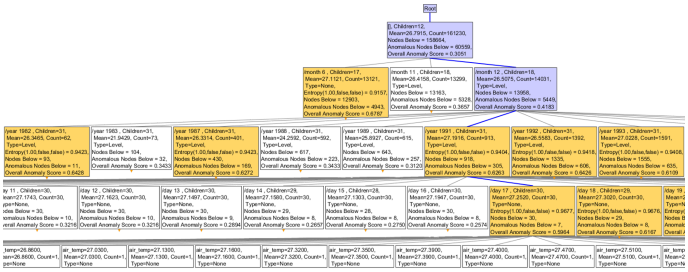

Fig. 5. El Nino anomaly tree: inverted month-year hierarchies. Anomalous nodes shaded orange.

points are considered normal comparison data, no points are flagged anomalous when the comparison data consists of all records. In our framework this happens less because these normal high-count records are dispersed throughout different parts of the anomaly tree. Second, the ensemble run of these three methods produced a higher precision level than any of these three algorithms independently. Third, the SHARD framework produced insight into many different levels of the anomaly tree. Specifically, investigating the SpaceTree nodes that were flagged anomalous, we determined: November is anomalous because it has no events but very high counts, August is anomalous because it has more events than the other months, all Saturdays are anomalous because they do not have any events, one Sunday is anomalous because it is the only Sunday with an event, and three days are anomalous because they are the only days with multiple events.

5.3 Climatology Data Results

Here we use a data set collected by the Pacific Marine Environmental Laboratory to study the El Nino and La Nina phenomena [6]. This data set contains climatology data from 1980-1998, during which there were 6 El Ninos (**1982**, 1987, **1991**, 1992, 1994, **1997**) and 1 La Nina (**1988**). The years in bold were considered very strong. The most anomalous months with unusually high temperatures are typically December of that year and January of the following year. There were 178,080 total readings of date, location, trade winds, humidity and air and sea surface readings.

Using the SHARD framework, we create an XML template that contains a typical, sequential date hierarchy year/month/day/{attribute} structure for each attribute. Using Entropy, threshold=1, the framework flags the appropriate El Nino and La Nina years with 87.5% precision and 58.3% recall using the ocean surface temperature; 88.9% and 66.7%, respectively, with the air temperature readings. Because we do not have ground truth weather information to accurately label all anomalous months and days, the precision and recall cannot be reported for the other levels of the anomaly tree.

We pause to mention that this data set contains many missing values since not every buoy was equipped to measure all of these attributes. Our framework can handle missing values by creating tree nodes only for values that are present and then searching for local anomalies within the tree.

Because of the flexibility of our XML templating, we also considered an alternative XML template that inverts months with years, so that the hierarchy is month/year/day as shown in figure 5. This means that for the month of December we have all year nodes

Industry	Years
Application Software	1999, 2000
Asset Management	2006 - 2008
Beverages - Brewers	2006 - 2008
Investment - Brokerage - Nat.	2000, 2006, 2007
Major Airlines	1998-2001, 2006, 2007
Regional Banks	2004-2007
Regional Airlines	1999, 2001, 2006

Table 3. Anomalous years in stock data set

as children and under each year node all December day measurements. This gives the researcher a very easy way to learn during which years December was most anomalous. Using this inverse technique, if we examine December, we find 85.7% precision and 77.7% recall at tagging the appropriate years. More interestingly, though, the highest ranked nodes correspond very well to the 'strong' El Nino years.

5.4 Stock Data Results

In these experiments, we analyze the NASDAQ daily stock quotes from 1998-2009 of 34 companies in the Technology, Financial, Services and Consumer Goods sectors. There is 1 date attribute, 7 numeric attributes and 5 categorical attributes for 14,805 records. We chose these years and industries because much happened in this decade: there was the dot.com bubble, followed by a correction year, 9/11, and another correction year following the real estate bubble. With the stock data we decided to study the most anomalous years by industry with the XML template configured as Industry/Year/Company Size/Company Name/Month/Day/Closing Price. We again used a default Entropy detector with a threshold of 1. A brief summary of these results can be found in table 3. Although we found the correlations between anomalies in Asset Management and those in Beverages - Brewers unexpected, the rest of the results seem easily interpretable, Application Software's dot.com boom and correction are rightly noted, the airlines show up in 2001, and many financial anomalies start to show up in 2004-2008. These results are consistent with expectations.

5.5 Discussion

The experimental results demonstrate the utility of having a hierarchical anomaly detection framework. Our synthetic and event attendance detection results indicate that the ensemble method has fewer false positives than the individual detection methods and a higher accuracy than any of the individual methods. We believe this results because the ensemble method is able to capture a more robust image of the data, whereas the individual algorithms are more suited to detect a particular type of anomaly.

Our results also show that the existence of anomalies at one granularity is not indicative of anomalies in other granularities. Figure 5 depicts a feature with many anomalous leaf nodes, but the parents of these nodes are not anomalous as indicated by 'Anomalous Nodes Below'. This is consistent with our understanding of point and contextual anomalies, and that one does not imply the other. Higher granularities are more descriptive of contextual anomalies, and not simply single point anomalies.

Using the SpaceTree application, we were also able to visualize our results in a meaningful way. The user is able to access relevant statistics about each node, as well as quickly see where anomalies are occurring. This is important in our work as mentally visualizing anomalies at multiple granularities is not an intuitive task.

6 Conclusions and Future Work

This work introduces SHARD, a framework that supports analysis of complex, multi-dimensional, hierarchical anomalies. Our framework is robust and allows for easy customization for different applications, as well as easy extensions for adding additional anomaly detectors and rankers. Using our prototype system, we illustrate both the flexibility and utility of this framework on both synthetic and real world data sets. Future work includes expanding the detectors in the framework, allowing for streaming analysis, demonstrating other semantic hierarchies that are not time based, and reducing the number of user specified parameters. Finally, many of the hierarchical aggregates mentioned are examples of cuboids. Extending our tree framework to a cube framework is another promising direction.

Acknowledgments: This work is supported by the FODAVA program at the National Science Foundation grant number #CCF0937070.

References

1. G.A. Barnard. Control charts and stochastic processes. *Journal of the Royal Statistical Society*, B21:239–271, 1959.
2. Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: identifying density-based local outliers. *SIGMOD Record*, 29:93–104, May 2000.
3. V. Chandola. *Anomaly detection for symbolic sequences and time series data*. PhD thesis, University of Minnesota, 2009.
4. Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computer Surveys*, 41(3):1–58, 2009.
5. AL Oliveira CM Antunes. Temporal data mining: An overview. In *KDD Workshop on Temporal Data Mining*, 2001.
6. A. Frank and A. Asuncion. UCI machine learning repository, 2010.
7. Zengyou He, Shengchun Deng, and Xiaofei Xu. An optimization model for outlier detection in categorical data. In *ICIC, Part I, LNCS 3644*, page 400 âĂŞ 409. Springer-Verlag, 2005.
8. Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22:85–126, 2004.
9. C. Joslyn and E. Hogan. Order metrics for semantic knowledge systems. *Hybrid Artificial Intelligence Systems*, pages 399–409, 2010.
10. Eamonn Keogh, Stefano Lonardi, and Bill Chiu. Finding surprising patterns in a time series database in linear time and space. In *ACM KDD*, pages 550–556. ACM, 2002.
11. Xiaolei Li and Jiawei Han. Mining approximate top-k subspace anomalies in multi-dimensional time-series data. In *VLDB*, pages 447–458. VLDB Endowment, 2007.
12. R. Nemani, H. Hashimoto, P. Votava, and F. et al Melton. Monitoring and forecasting ecosystem dynamics using the terrestrial observation and prediction system (tops). *Remote Sensing of Environment*, 113(7):1497–1509, 2009.

13. A. Patcha and J.M. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, 2007.
14. Catherine Plaisant, Jesse Grosjean, and Benjamin B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. *IEEE Symposium on Information Visualization*, 0:57, 2002.