Hilighter: Automatically Building Robust Signatures of Performance behaviors for Small and Large-Scale Systems

Peter Bodik, Moises Goldszmidt, Armando Fox
RAD Lab, EECS Department, UC Berkeley
Microsoft Research, Silicon Valley

By,
Shuvodeep Chatterjee, s0933652
Manas Rai, s0975444
Introduction to the Problem

• Interactive network services experience performance problems for a wide variety of reasons.

• Diagnosing these problems is complicated due to various factors
  – Large scale deployment of these services,
  – their complex interactions,
  – large number of metrics collected in each component of the system.
Background

• Statistical analysis techniques could successfully be used to construct compact signatures of distinct operational problems in Internet server systems.

• They are used to index past problems and identify particular operational problems as new or recurrent.

• New statistical technique used for constructing signatures yields encouraging results (Logistic Regression with L1 Regularization)

• This approach was validated on data from an Internet service testbed and also from a production enterprise system comprising hundreds of servers in several data centers.
The First Solution

• Cohen et. al. proposed a solution to this problem.

• Proposed an approach to alleviate the diagnosis problem by automatically building a compact signature of each failure mode.

• These signatures are then used to:
  – a) localize the problem (sometimes establishing a path to the root cause)
  – b) detect recurring instances of the same problem
  – c) serve as indexes for annotations regarding resolution.

• The applications b) and c) were addressed using a similarity search over the signatures.

• Uses a bayesian network classifier

• Reduced problem identification to information retrieval.

• The search method considers changes to only one feature at a time in a greedy fashion and so with large no. of features in a typical problem, only a small part of the feature space is explored.
Reason for selection of Technique

• Logistic Regression with L1 regularization is preferred for three reasons:

  1. Works well for cases where the number of features is an order of magnitude larger than the number of samples.
  
  2. Works for problems with over 50000 samples.
  
  3. Model works well for different sets of data.
Logistic Regression

• Signatures are of 2 types: Normal and Anomalous
• So we need a binary classifier
• Logistic Regression is used:

\[ p_j \equiv P(Y_j = 1 | M, m) = \frac{\exp(\sum_i \beta_i m_i)}{1 + \exp(\sum_i \beta_i m_i)} \]

where \( \beta \)s are the parameters, \( j \) is the sample and \( i \) is the feature/metric.

• For Fitting maximize the likelihood function

\[ L(\beta) = \prod_j p_j^{y_j} (1 - p_j)^{1 - y_j} \]
L1 Regularization

- Because of a large number of features and less samples, Regression will usually overfit the data.
- To avoid this, use L1 Regularisation
- Regularization reduces the complexity of the model
- This is done by driving most of the parameters i.e $\beta$s to zero.
- In L1 regularization the regularization parameter is
  \[ R(\beta) = \sum |\beta_i| \]
- Forming the Lagrangian for the L1 regularizer, we get
  \[ R(\beta) = \sum |\beta_i| \leq \lambda \]
- Most of the betas become zero, so feature selection becomes easy.
Signatures

• The automatic construction of signatures is based on classification with feature selection.

• Finds subsets of measured system metrics (CPU, memory utilization, etc.)

• The signature of a particular instance of a problem is then extracted

• By finding whether each of the selected features contributes to classifying the instance as anomalous or as normal.

• Described the benefits of this approach on a transactional system consisting of tens of geographically distributed servers.
Signature Construction

- The Signature captures the state of the system at a given time.

- A small number of metrics that predict both normal and abnormal performance with high accuracy are selected.

- HiLighter transforms the measured metrics using standard preprocessing:
  - a) removes the constant metrics,
  - b) augments each metric by computing a gradient and standard deviation with respect to one or two time periods in the past,
  - c) normalizes the metrics to zero mean and variance of one.
Models are evaluated using Balanced Accuracy:
  - Averages the probability of correctly classifying the normal with the probability of detecting an anomaly.

λ - The constraint on L1 Regularization upper bounds the number of metrics which form the signature.

For optimal classification, HiLighter used 5 fold cross validation along with Binary Search over the values of λ.

HiLghter fits the final model using metrics selected by L1.

It adds L2 constraint to stabilize for co-related features.

Finally reports
  - Balanced Accuracy,
  - Confusion Matrix incl. False Positives & False Negatives
  - Balanced Accuracy of using each metric alone as a classifier
Case Study 1 :
High variability in response time during application testing

• Internet application on multiple data centers.

• Metrics measured every 30 seconds such as: workloads, response times, OS level, dotNET Runtime, app. specific metrics.

• Performance Anomaly: large variance in 99th percentile of latency.

• These values range from 50-1200 ms.

• Threshold selected was 400 ms.

• Values above the threshold are abnormal.
Case Study 1:
High variability in response time during application testing

- Feature Extraction resulted in Dataset with 86 data points and 2008 features.
- Hilighter selected only 2 metrics (both dotNET runtime):
  - Number of Induced Garbage Collection
  - Change in number of allocated bytes
- Final Cross-validated balanced accuracy of 0.94 was obtained.
Case Study 2 & 3:
Configuration Issue/Overloading in a Production Datacenter

• Focuses on a performance issue in the production version of a large-scale Internet application.
• Application is deployed across six data-centers with hundreds of servers, each serving one of five roles in the overall application.
• The “application core” servers—(Role 1 servers)—contain the main application logic
• Each server measures the average latency of requests it processed.
• **Case Study 2**: The performance problem is triggered by a deployment of a bad configuration file to all Role 1 servers
• **Case Study 3**: One data center overloaded – redirecting traffic to other datacenters.
• Significant processing delays.
Case Study 2 & 3:
Configuration Issue/Overloading in a Production Datacenter

- Feature Extraction led to dataset with ~500 metrics and 100 datapoints per server and time interval.
- Threshold supplied by service operator.
- HiLighter ran on 4 groups of Role 1 servers.

Case Study 2:
- HiLighter selected 6-9 metrics of which 5 were consistent across different servers.
- Crossvalidated Balanced Accuracy ranged from 0.92 - 0.95.
- Observed that more metrics selected increased balanced accuracy.
  - Eg. 0.83 with 2 metrics to 0.92 with 6 metrics.

Case Study 3:
- HiLighter selected 4-5 metrics. Of which 3 were consistent across different servers.
- Model accuracy recorded between 0.95 & 0.99.

- Models generated identical metrics for both cases
Case Study 2: Results

- After adjusting the data for missing values, Metrics selected in different groups of servers were almost identical.
Case Study 3: Results

Table 3. Results for case study 3 (overloaded datacenter) for servers in role 1 in datacenter 5; each cell shows balanced accuracy of a model trained on the group of servers specified in the row header and evaluated on the group specified in the column header. Entries along the diagonal represent the cross-validation results.
Signature Evolution

• Label each metric as abnormal if $m_i \beta_i > 0$, normal otherwise.
• Construct signature for each data point and time interval.
• One signature marks anomaly – all metrics are abnormal.
• Plot changes in signature over time – observe when metrics become abnormal.
Signature Evolution: Case Study 2

- Signature C (all 6 metrics abnormal) marks anomaly
- Spotting Signature B (5/6 abnormal) allows prediction of anomaly.
Signature Evolution: Case Study 3

**Figure 2.** As in Figure 1, the bottom plot shows the latency on the 71 role-one servers. The plots on the top show the state of each of the selected five metrics on these servers. Dark gray means that the value of the particular metric was abnormal with respect to the HILighter model.

- Metric 1 abnormality starts before the performance issue – can be used as a predictor
- Metrics 2, 3 & 5 correspond well to the duration of the anomaly
Case Study 2: Anomaly Localization

Running HiLighter on Server Roles 2 to 5 (apart from 1) produced the above data.

To identify abnormal servers check whether balanced accuracy is high.

Table 2. Each cell shows the cross-validated balanced accuracy for case study 2 (configuration issue) for servers in all roles in all datacenters. This might help localize the source of the problem. Note: not all datacenters contain servers of all roles.
Conclusion

• It is a part of the diagnostic tool Artemis for finding bugs and performance problems in large clusters of machines.

• Encourage the addition of new features (based on the measured metrics)

• Will help engineers add domain knowledge without being constrained by the characteristics of the model building process.

• L1 regularization produced robust models in scenarios:
  – with large numbers of features and only few datapoints,
  – in cases of multi-datacenter applications with hundreds of machines & datasets
Remarks

- In case a metric becomes abnormal long before the anomaly starts, the Application might report a false negative unless it is the only anomalous metric.
- The paper does not compare the results of its approach with previous work (Cohen et al.)
- It is mentioned that all major classifiers were tried including Naïve Bayes and Support Vector Machines, but results were not good.
- However no evidence is given.