Real-time Detection of Performance Anomalies for Cloud Services

Olumuyiwa Ibidunmoye
Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden
Email: muyi@cs.umu.se

Thijs Metsch
Intel Labs Europe
Collinstown Industrial Park
Leixlip, Ireland
thijs.metsch@intel.com

Erik Elmroth
Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden
Email: elmroth@cs.umu.se

I. Introduction

Service performance degradation and downtimes are a common on the Internet today. Many on-line services (e.g. Amazon.com, Spotify, and Netflix, etc.) report huge loss in revenue and traffic per episode. This is perhaps due to the correlation between performance and end-users's satisfaction.

The high-cost of performance anomalies in large-scale cloud services drives the need for timely detection of such anomalous behaviours. Service providers are, as a result, very concerned about detecting unwanted behaviour quickly and in real-time before they impact users or violate service-level objectives [1]. Existing approaches for detecting application performance anomalies are based on hard thresholds or application instrumentation [2]. A limitation of thresholds is that they may become obsolete workload changes or require assumption about the underlying statistical distribution of the data. Dynamic behaviour of cloud services calls for more robust approaches. Hence, we ask the following question; '*how can we continuously detect deviations in sequential measurements of a service performance metric?*' In the following sections, we introduce and evaluate two techniques to address this question.

A. BAD: Behaviour-based Anomaly Detection

This technique exploits the knowledge of the characteristic *probability density function* (PDF), of normal performance measurements (i.e. baseline). BAD estimates a model of the unknown PDF of a given baseline using Kernel Density Estimation (KDE) [3]. The standard KDE algorithm is known to produce spurious density estimates due to the use of global bandwidths [4]. We develop an adaptive KDE (AKDE) that incorporates local bandwidths to account for local variations in the data as follows:

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h\lambda_i}K_h(\frac{x-x_i}{h\lambda_i}) \tag{1}$$

Eq. (1) estimates the unknown density function $\hat{f}(X)$ of a baseline series $X : \mathbb{R}^{n\times 1}$ by aggregating a set of scaled Gaussian kernel functions $K_h(\cdot) = \frac{1}{h\sqrt{2\pi}}\exp(-\frac{(\cdot)^2}{2h^2})$ centered at each data point with smoothing controlled by a *global* and local bandwidth parameters h and $h\lambda_i$ respectively. The individual local density is computed using the square root rule following the procedure in [4]. The KDE algorithm has a time complexity of $\mathcal{O}(n^2)$, hence we implemented AKDE using k-dimensional trees which readily offer $\mathcal{O}(n\ log\ n)$ k-nearest neighbour computations.

Given the AKDE model, $\theta_{\mathcal{B}} = \hat{f}_{\mathcal{B}}(X)$, of a baseline $\mathcal{B}$, and a new observation x_0. We compute the likelihood of $\mathcal{B}$ taking on the value of x_0 given $\theta_{\mathcal{B}}$ as $p(x_0|\theta_{\mathcal{B}})$. For a suitably chosen parameter, ϵ, x_0 is classified anomalous if $p(x_0|\theta_{\mathcal{B}}) < \epsilon$. This is based on the premise that normal observations lie in dense regions while anomalies are rare and thus lie in sparse regions of the PDF.

B. PAD: Prediction-based anomaly detection

PAD exploits the temporal dependency in successive observations to detect abrupt changes in performance measurements based on Holt-Winters forecasting [5] and Exponentially Weighted Moving Average Control Charts (EWMA) [6]. Given a time-series $X = \{x_1, \cdots, x_{t-1}, x_t, x_{t+1}\cdots\}$, Holt-Winters decomposes X into a *level* component (the intercept of a signal), ℓ_t, a linear *trend* (the long term direction or slope), b_t, and a *seasonal* (the systematic, calendar related movement) component, s_t using the following expressions [5];

$$\ell_t = \ell_{t-1} + b_{t-1} + \alpha e_t \tag{2}$$

$$b_t = b_{t-1} + \alpha\beta e_t \tag{3}$$

$$s_t = s_{t-m} + \gamma e_t \tag{4}$$

where α, β and γ are smoothing parameters for the components respectively ($0 < \alpha, \beta, \gamma \leq 1$). The one-step-ahead forecast at time t using previous forecast components at time $t-1$ is thus $\hat{x}_{t|t-1} = \ell_{t-1} + b_{t-1} + s_{t-m}$. The corresponding forecast error or residual, e_t, is computed as $e_t = x_t - \hat{x}_{t|t-1}$, where x_t is the observed metric value at time t and $\hat{x}_{t|t-1}$ is the predicted value using the most recent smoothed components from time $t-1$. The forecast errors serve as the basis for anomaly detection. First, the residuals are smoothed by a time controlled parameter λ $(0 < \lambda \leq 1)$ according to Eq. 5.

$$e'_t = \lambda e_t + (1 - \lambda)e'_{t-1} \tag{5}$$

The smoothed residuals are then plotted on a control chart with control limits defined as follows [6]:

$$(UCL, LCL) = \mu_t \pm L\sigma_t \sqrt{\frac{\lambda}{(2 - \lambda)}\left[1 - (1 - \lambda)^{2t}\right]} \tag{6}$$

While λ determines the rate of smoothing, L is a multiple of the standard deviation and controls how sensitive the chart is to shifts around the center line. The center line of the chart, CL is the mean, μ_t of past smoothed errors with standard deviation σ_t. Parameters μ_t, σ_t are computed adaptively using Welford's iterative algorithm[1] using observations up to time step t. Out-of-control observations whose residuals violate the control limits, i.e. $e'_t > UCL$ or $e'_t < LCL$, are flagged as anomalies.

II. Evaluation and Results

To evaluate and compare the performance of both techniques we tested them with time-series from the `A1Benchmark` of the Yahoo! Webscope anomaly detection dataset [7]. The benchmark consists of labeled time-series representing metrics of various Yahoo services. Each time-series is replayed sequentially to emulate real life scenario. The baseline for the BAD algorithm is based on windows of K sequential observations ($K = 100$ in our setup). That is, first K observations act as baseline to detect observations in the next window. Anomalous points are excluded from each baseline before model estimation to avoid noise. The detection threshold is set as $\epsilon = 0.001$ for all dataset. The global bandwidth parameter, h, is derived through grid search cross-validation while the k-dimensional tree is based on modules in `scikit-learn`[2]. For PAD, the same windowing method is used to estimate parameter α, β [3] but are optimized to minimize the mean absolute percentage error (MAPE) of the forecasts. Also, the control chart parameters are set to standardized values $\lambda = 0.2$ and $L = 3$ [6].

Fig. 1 shows the spread of the false positive (or alarm) rate (FPR), the proportion of normal samples that are incorrectly identified as anomalous, when the methods are tested on 52 non-seasonal time-series from the benchmark. The average FPR for PAD is 8.7% (±7.5%) and 4.4% (±7.3%) for BAD. Further, we included more time-series from the same benchmark with diverse characteristics (64 in total). PAD produced lower FPR of 9.3% (±7.5%) compared to 16.5% (±26.9%) for BAD.

[1]https://en.wikipedia.org/wiki/Algorithms_for_calculating_variance
[2]http://scikit-learn.org/stable/
[3]Eq. (4) is ignored since we used non-seasonal time-series.

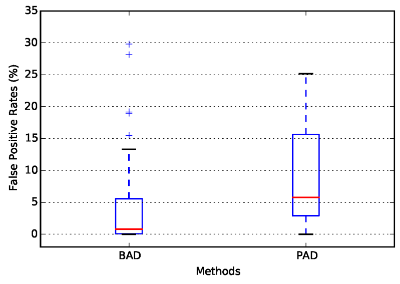

Fig. 1. Comparison of the False alarm rates of both methods on 52 time-series of the Yahoo! Webscope dataset.

III. Discussion and Conclusion

Though BAD seems to be less spread than PAD and hence fewer false alarms, it contains more outliers than PAD. BAD performed better on stationary time-series as the outliers in Fig. 1 correspond to non-stationary time-series with sharp variation in trend. The forecasting technique used in PAD does not exclude anomalies in its forecast thereby affecting the movement and sensitivity of the control limits. In general, PAD performed better on the more diverse dataset because it captures shifts in trend better than BAD.

In conclusion, we have proposed and evaluated two schemes for detecting anomalies in real-time service performance metric measurements. Future work will extend PAD with forecasting that is robust to anomalies and to include time-series with (seasonal patterns) as well as evaluate scalability of both methods when deployed for multiple services in a virtualized testbed.

Acknowledgment

Supported by the Swedish Research Council (VR) for Cloud Control (C0590801) and the EU's 7th Framework Programme CACTOS (610711).

References

[1] T. Metsch, O. Ibidunmoye, V. Bayon-Molino, J. Butler, F. Hernández-Rodriguez, and E. Elmroth, "Apex Lake: A Framework for Enabling Smart Orchestration," in *Proceedings of the Industrial Track of the 16th International Middleware Conference*. ACM, 2015, pp. 1:1–1:7. [Online]. Available: http://doi.acm.org/10.1145/2830013.2830016

[2] O. Ibidunmoye, F. Hernández-Rodriguez, and E. Elmroth, "Performance Anomaly Detection and Bottleneck Identification," *ACM Comput. Surv.*, vol. 48, no. 1, pp. 4:1–4:35, July 2015. [Online]. Available: http://doi.acm.org/10.1145/2791120

[3] C. C. Aggarwal, *Outlier Analysis*. Springer Science & Business Media, 2013.

[4] F. J. G. Gisbert, "Weighted Samples, Kernel Density Estimators and Convergence," *Empirical Economics*, vol. 28, no. 2, pp. 335–351, 2003.

[5] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2014, accessed: 2016-02-10.

[6] D. C. Montgomery, *Introduction to Statistical Quality Control*. John Wiley & Sons, 2007.

[7] Yahoo! Webscope, "S5 - A Labeled Anomaly Detection Dataset, version 1.0." [Online]. Available: https://webscope.sandbox.yahoo.com/catalog.php?datatype=s