INFORMATION THEORETIC MULTIVARIATE CHANGE DETECTION FOR MULTISENSORY INFORMATION PROCESSING IN INTERNET OF THINGS

Lev Faivishevsky

Intel
lev.faivishevsky@intel.com

ABSTRACT

Internet of Things (IoT) is one of the main technological trends in the recent years. It allows machine-to-machine communication over the internet. Almost each device may transmit information from its sensors over the web to enable centralized insights derivation in an appropriate cloud architecture.

In this paper we review analytical aspects of the sensory information processing. We emphasize the importance of multisensory approach, in which the joint distribution of all sensors values of a device is used to derive insights out of the stream of sensory data. We introduce a novel information theoretic multivariate change detection method based on k-nearest neighbor (kNN) estimation. The algorithm is designed and implemented to satisfy the requirements of IoT for fast online parallel multisensory information processing. We provide a numerical evidence of the validity of the proposed method on simulated and real world data.

Index Terms— Internet of Things, Change detection

1. INTRODUCTION

Change detection is an important statistical method for detecting changes in a system. The method monitors data streams from sensors embedded in a device. If the values significantly change over a period of time the method outputs an alert. This is a widely needed capability in IoT where a vast amount of devices should be monitored in the unsupervised way.

In this paper we introduce the novel method of Information Theoretic Multivariate Change Detection (ITMCD) for multisensory information processing in IoT. The method was developed and implemented in an IoT cloud computing platform. The method alerts about a change in a device behavior by monitoring sensor values samples in a sliding time window of a fixed length in an online manner. If the multivariate joint distribution of sensors values in the beginning of the time window differs significantly from the joint distribution of the sensors values in the end of the window then the method detects a change. To measure the difference between these multivariate distributions we apply nonparametric estimators of relative entropy based on k-nearest neighbor calculations [2].

The rest of the paper is organized as follows. In section 2 we review challenges of information processing in the IoT domain. Section 3 describes the proposed algorithm ITMCD in detail. Section 4 is devoted to conducted numerical experiments.

2. INFORMATION PROCESSING IN IOT

2.1. IoT Cloud Analytics Platform Characteristics

The Internet of Things domain serves a large variety of verticals with significantly different types of data and different peculiar characteristics. For example, the stationary sensor signals are frequent in the manufacturing, whereas daily periods of activity characterize human generated data in healthcare and wearable industries. Therefore an analytical method applicable in an IoT cloud cannot rely on a specific data pattern and should be adaptive to a new data shape. An additional practical consequence of this variety is the lack of the supervision information available for the algorithm setup because it is impossible to develop a universal flow for label information treatment. The setup stage therefore is limited to collecting device sensor values for a certain period of time and deriving general insights of their distribution.

Real time insights are of special value in many important use cases, including medical and security applications. This raises importance of algorithms that may run in an online manner, providing most updated insights about devices. This requirement combined with Big Data nature of large scale computations emphasizes the need for easy parallelizable algorithms which run in a real time with small latency. For example, if an online algorithm operates on a sliding time window of samples then an inherent latency is induced by the length of the time window. Hence algorithms employing shorter time windows are beneficial.

The cloud platform should provide valuable information to a user, reporting about true events occurring in an environment of a device. The false alarm rate of an algorithm is the crucial parameter influencing the user experience and the added value of the whole analytics platform. The false alarm rate should be tunable to adapt to a specific user capacity to react to an alert. In particular, the stability of the false alarm during the running time of an algorithm is a crucial aspect of the algorithm suitability for deployment in a cloud analytics platform.

Typically an IoT cloud platform serves not a single instance of a device, but a set of similar devices. As an example, the platform may serve a fleet of manufacturing tools in a semiconductor fabrication plant performing a same operation. As another example, the platform may collect and process information from cars of a specific model, produced by the same manufacturer. In these cases the cloud computing platform may benefit from aggregating information transmitted by a subset of devices. Statistical insights derived from this information may then be applied to the whole family of devices, thus leveraging the Big Data nature of cloud computing in IoT. As a consequence, a data processing algorithm in IoT analytics platform benefits from being extendable to a multidevice mode.

2.2. Multisensory IoT information processing

In many use cases in IoT a device is equipped with several sensors, each of them is able to capture a specific characteristics of environment. For instance, a manufacturing tool may contain a temperature sensor, a pressure sensor, a humidity sensor, whereas a wearable may be equipped with an accelerometer and a GPS sensor. In security domain, multivariate behavioral and biomechanical aspects of a specific user activity session, such as typing dynamics may be recorded and transmitted for an online authentication in a cloud. A real world phenomenon in a device environment usually manifests itself in a variety of physical aspects.

In all these situations the ability to gain insights from multisensory information by analyzing joint distribution of device sensors is beneficial. First, the multisensory treatment allows to amplify a signal from a real phenomenon by taking into account multiple sources of information. Small changes in individual sensor values may be treated as insignificant, however processing such individual sensors as a combination may reveal a significant event. Second, there are changes in the device behavior that cannot be detected in individual sensor levels. For example, a change in a correlation pattern between two sensors of a device may witness a significant change in the system behavior however it cannot be traced inspecting individual sensors only. An IoT information processing algorithm will therefore benefit from being able to process a multisensory output.

2.3. Change detection aspects for IoT

Change detection methods aim to detect consistent changes in a distribution of random variables over a certain time period [3]. As we discussed above due to high variability of possible data patterns no prior parametric form can be assumed for sensor values distribution. It means that change detection algorithms should be applied in two sample mode. In this model distributions of sensor values before and after a suspected change are unknown and a distance between them should be measured in a nonparametric way. Then a change is detected if the distance is above an appropriate threshold.

A well-known algorithm for change detection is Kolmogorov Smirnov Test [4] which employs estimation of empirical distributions of the sensor values before and after the change. The KS uses a maximal absolute difference between curves of empirical distributions as a statistic and assumes the continuity of these distributions. Under the null hypothesis that the both distributions are equal this statistic has a known Kolmogorov distribution regardless of the underlying sensor values distributions. Using this Kolmogorov distribution a desired false alarm rate may be easily tuned. However the KS may be applied only to one-dimensional random variables, hence it cannot be used in multisensory information processing directly.

A number of recent multivariate change detection algorithms employ One Class Support Vector Machines (OCSVM) for multivariate probability distribution estimation. For example, Generalized Kolmogorov Smirnov Test [5] applies OCSVM for the recovery of level sets of a multivariate distribution and then utilizes one-dimensional KS. The online kernel change detection algorithm [3] includes learning two OCSVMs followed by distance measure. The common drawback of these algorithms with regard to IoT is the need to perform OCSVM model learning as a condition for the change detection. The OCSVM model learning is computationally expensive since it involves solving a quadratic optimization problem. Even more important, it requires a large number of data points for the model training, that is an order of magnitude bigger than a number of sensors in device. From the point of view of IoT these algorithm pose a significant computational load on the cloud platform and do not allow fast online change detection due to the large number of time points required for training.

The algorithm of Maximum Mean Discrepancy [6] involves a kernel evaluation on each pair of samples for considered time window. This would pose a hard computational load on the cloud as well. In addition, the performance of the algorithm is affected by a choice of a specific kernel, which can negatively impact the robustness of the method with regard to IoT. Finally, the bootstrapping based [7] change detection leads to complex kdq-tree computations, which are not appropriate for practical implementations in the real-time parallel cloud architecture.

3. ITMCD ALGORITHM

A change detection method in IoT runs over a stream of sensors values $X^t = X_1^t, ..., X_d^t$, where d is a number of sensors in a device and t is a time index. We assume that all sensor values are sampled equidistantly. To detect a consistent change in the behavior of the device at a reference time t_0 a time window of sensor values around this time moment $X^{t-p}, ..., X^{t_0}, ..., X^{t_f}$ is processed. We denote by p the amount of d-dimensional samples prior to the reference time, and f is the amount of samples after the reference time. A change is detected if the distribution of points in the *Past* $P \sim Prob(X_t|t \in [-p, ..., -1])$ is significantly different from the distribution of points in the *Future* $F \sim Prob(X_t|t \in [1, ..., f])$:

$$D(P||F) > T \tag{1}$$

where D is a distance between distributions and T is a threshold, that defines both detection rate and false alarm rate of the method. As a consequence, a change detection method is defined by a choice of a distance D and a way to determine a threshold T. As discussed above, cloud analytics platform requirements do not allow to assume any parametric form of distributions, so the distance estimation should be performed in a nonparametric way:

$$D(P||F) \approx \hat{D}(X^{t-p}, ..., X^{t-1}||X^{t_1}, ..., X^{t_f}) \tag{2}$$

We proposed to use the relative entropy as a measure of dissimilarity between distributions. This information theoretic quantity also known as Kullback Leibler divergence is widely used as a distance between distributions.

$$D(P||F) = \int P(x) \log \frac{P(x)}{F(x)} \tag{3}$$

This metric may be estimated in a nonparametric manner [8]. The k-th Nearest Neighbor (kNN) technique provides a convenient and computationally fast way for the estimation. Denote an Euclidean distance g for a d-dimensional vector X^i from the Past to its nearest neighbor X^j in the Past as

$$\rho_p(i) = \min_{j \neq i | i,j<0} g(X^i, X^j) \tag{4}$$

We also define the distance of X^i to its nearest neighbor X^l in the Future as

$$\nu_f(i) = \min_{l=1,...,f} g(X^i, X^l) \tag{5}$$

Then the estimator of [2] is given by

$$\hat{D}_{p,f} = \frac{d}{p} \sum_{i=1}^{p} \log \frac{\nu_f(i)}{\rho_p(i)} + \log \frac{f}{p-1} \tag{6}$$

The authors [2] established asymptotic unbiasedness and mean-square consistency of the estimator (6). The same proofs could be applied to obtain k-nearest neighbor version of the estimator:

$$\hat{D}^k_{p,f} = \frac{d}{p}\sum_{i=1}^{p}\log\frac{v^k_f(i)}{\rho^k_f(i)} + \log\frac{f}{p-1} \quad (7)$$

The relative entropy is not symmetrical with respect to distributions P and F, so we apply symmetrization to get a final score function S for our change detection method:

$$S = \hat{D}(X^{t-p}..X^{t-1}||X^{t_1}..X^{t_f}) + \hat{D}(X^{t_1}..X^{t_f}||X^{t-p}..X^{t-1}) \quad (8)$$

Calculating the score S requires finding the k-th nearest neighbor for each point in the time window. The computational complexity of this computation is $O(n\log n)$ for a time window of length n [9]. After the k-th nearest neighbors are found, the rest of computations is performed in linear time. The overall computational complexity of score S calculation is therefore $O((p+f)\log(p+f))$.

After we defined the score to be computed, we introduce a way to compute a threshold T that ensures a desired false alarm rate α. We utilize the algorithm setup time to collect R reference time points $X^1, ..., X^R$. These reference time points represent a regular device behavior therefore we may calibrate a false alarm rate α on them. We apply a sliding time window score S computation to the data, that results in values of scores $S^s, s \in [1, ..., R-p-f]$. We sort these values in descending order to obtain a monotonic decreasing sequence $\hat{S}^1 \geq \hat{S}^2 \geq ... \geq \hat{S}^{R-p-f}$. Then the following choice of threshold T ensures the desired false alarm rate:

$$T = \hat{S}^{[\alpha(R-p-f)]} \quad (9)$$

where $[\alpha(R-p-f)]$ is the closest integer to $\alpha(R-p-f)$. As a last remark, the sensor values $X_1, ..., X_d$ may have different dynamic ranges, therefore a contribution of a sensor with the highest nominal values may dominate in the score computation. To equalize the contributions of sensors a preprocessing stage may be required. We linearly scale each sensor to bring its values to zero mean and unit variance:

$$\hat{X}_i = k_i * X_i + b_i \quad (10)$$

The constant multiplicative k_i and b_i additive factors are calibrated during the algorithm setup, and applied during the algorithm run time. The algorithm, which we dub Information Theoretic Multivariate Change Detection (ITMCD), is summarized in Figure 1.

The resulting algorithm satisfies all the requirements of a change detection algorithm for an IoT cloud analytics platform. This multivariate method does not assume any assumptions on data pattern, runs in a purely unsupervised mode and allows a fast online parallel implementation. It is implemented in parallel Map-Reduce architecture and deployed in an IoT Cloud Analytics Platform.

The ITMCD algorithm is readily extendable to the multidevice mode. Suppose we have a family of devices Θ, such that a multisensory data stream of device $\theta \in \Theta$ is given by $X^t_\theta \in R^d$. Data streams of a subset of devices $\overline{\Theta} \subset \Theta$ are observed during the setup stage and a threshold $T(\theta)$ (9) is calibrated for each device $\theta \in \overline{\Theta}$.

To detect a change in a device $\hat{\theta} \in \Theta$ with a data stream $X^t_{\hat{\theta}}$ not observed in the setup stage ($\hat{\theta} \notin \overline{\Theta}$) we need to compute the corresponding threshold $T(\hat{\theta})$. Similarity of devices inside family Θ allows to replace the unknown threshold by its expectation. Assuming that the subset $\overline{\Theta}$ is representative this expectation may be estimated by averaging thresholds calculated in the setup stage:

$$T(\hat{\theta}) \approx \int T(\theta)d\Theta \approx \int T(\theta)d\overline{\Theta} \approx \frac{1}{|\overline{\Theta}|}\sum T(\theta) \quad (11)$$

Setup.
Input: Sensor values $X^1, ..., X^R \in \Re^d$, false alarm rate α, k, past and future time window lengths p and f

Output: Threshold T
Method:

- Compute score S (8) for each sliding time window of length $p+f+1$ in reference data
- Sort score values in a decreasing order
- Determine a threshold value T according to (9)

Run time.
Input: T, k, a time window of sensor values $X^{t-p}, ..., X^{t_0}, ..., X^{t_f} \in \Re^d$

Output: Decision whether a change occurred at time t_0
Method:

- Compute score S (8) for the input values
- Alert about a change if $S > T$

Fig. 1. The ITMCD algorithm

Table 1. Performance on 1D Gaussian data

Change	α	KS F	KS E	ITMCD F	ITMCD E
$\mu+=1$	0.1%	0.04%	13.1%	**0.09%**	**13.9%**
$\mu+=1$	1%	0.5%	32.6%	**0.6%**	**34.3%**
$\mu+=2$	0.1%	0.04%	87.1%	**0.06%**	**92.1%**
$\mu+=2$	1%	0.6%	94.0%	**0.8%**	**96.0%**
$\sigma+=1$	1%	0.7%	1.5%	**1.2%**	**6.3%**
$\sigma+=2$	1%	0.4%	2.5%	**1.1%**	**13.2%**
$\sigma+=3$	1%	0.6%	3.6%	**0.9%**	**26.4%**

4. EXPERIMENTS

4.1. Simulated data

We performed a number of comparative experiments with the ITMCD algorithm. First, we evaluated its performance on simulated data in comparison with the well established Kolmogorov Smirnov (KS) test. Since the Kolmogorov Smirnov Test is limited to one-dimensional inputs, we simulated a number of Gaussian distributed sequences with zero mean and unit variance. At some time point the parameters of the distribution were changed either in terms of mean μ or in terms of standard deviation σ. The task was to detect the change with a predefined false alarm rate α. A better change detection method should have a higher detection rate (E) and keep the false alarm rate (F) closer to α in the run time. We fixed the future window size $f = 10$ and used a larger past time window size $p = 50$ for small $\alpha = 0.1\%$, whereas a smaller past time window size $p = 30$ was used for a larger $\alpha = 1\%$. The kNN parameter was set to $k = 8$. The results are summarized in Table 1.

We observed that larger window sizes lead as expected to better performance of both KS and ITMCD. The detection rate of ITMCD is generally higher than that of KS, especially with regard to detection of standard deviation changes. Also ITMCD exhibits more stable false alarm rate, which is an important advantage in IoT.

Our next numerical simulations were dedicated to the multivariate case. The task was to detect a change of a correlation coefficient of 2D Gaussian data. We compare ITMCD with MMD. To ensure

Table 2. Performance in a keystroke security application

α	w	MMD F	MMD E	ITMCD F	ITMCD E
0.5%	2	NA	NA	**0.6%**	**40.2%**
1%	2	NA	NA	**0.7%**	**46.7%**
2%	2	NA	NA	**1.0%**	**58.1%**
3%	2	NA	NA	**1.8%**	**65.4%**
4%	2	NA	NA	**2.7%**	**70.9%**
5%	2	NA	NA	**3.0%**	**72.9%**
10%	2	NA	NA	**8.6%**	**84.2%**
0.5%	3	NA	NA	**1.5%**	**61.7%**
1%	3	1.6%	0	**1.8%**	**65.5%**
2%	3	1.6%	0.1%	**2.5%**	**71.3%**
3%	3	1.6%	0.2%	**3.0%**	**72.2%**
4%	3	1.6%	1.3%	**4.3%**	**74.5%**
5%	3	1.6%	3.1%	**5.6%**	**76.8%**
10%	3	1.9%	59.3%	**9.5%**	**82.7%**
0.5%	5	NA	NA	**7.0%**	**88.1%**
1%	5	**1.6%**	45.7%	2.4%	**91.1%**
2%	5	**1.7%**	81.0%	3.1%	**92.7%**
3%	5	1.7%	86.8%	**3.6%**	**96.6%**
4%	5	1.8%	91.3%	**6.1%**	**97.7%**
5%	5	1.9%	92.2%	**6.2%**	**97.7%**
10%	5	2.0%	92.8%	**10.8%**	**98.5%**

a fair comparison we used windows of sizes 120, where after 110 points a change of correlation coefficient ρ from 0 to 0.9 might occur. ITMCD used the first 100 points as a reference setup data, then it was applied in run time to remaining 20 points with both the past window size p and the future window size f equal to 10. MMD used all 120 points as an input, and aimed to detect a change between first 110 points and remaining 10 points. We used a range of predefined false alarms and built Receiver Operating Curves (ROC) for both methods. The results of 1000 repetitions are summarized in Figure 2. ITMCD demonstrated a better ROC curve in the whole range of tested detection and false alarm rates.

4.2. Real world data security application

We demonstrate a security application of the proposed method. We used keystroke dynamics data [10] to distinguish people by their typing rhythms. A digital fingerprint would tie a person to a computer-based crime in the same manner that a physical fingerprint ties a person to the scene of a physical crime.

In the context of a typical security application a user registers in a site and types a password. Entering the site next times the user is required to type the password again. If the users credentials were stolen then a hacker would attempt to enter the site with them. During these attempts the password would be entered in a different rhythm. Therefore a change detection method applied to credentials keystroke rhythm may recognize a malicious access to the user account. The method treats a keyboard as a device and its keys as sensors. The sensor values are key press times and key release times for each symbol. This way entering a password of 10 symbols results is processed as a 21-dimensional sensor output of a device.

A challenge of a security application is that a user may be attacked soon after the sign up to a site. Therefore the change detection method cannot assume a long history of reference data for a specific user. From the other hand, if the credentials are stolen, then the attack should be detected as soon as possible. This means that a change detection should be used with the smallest possible time window. In addition, the algorithm setup cannot be applied to a specific user.

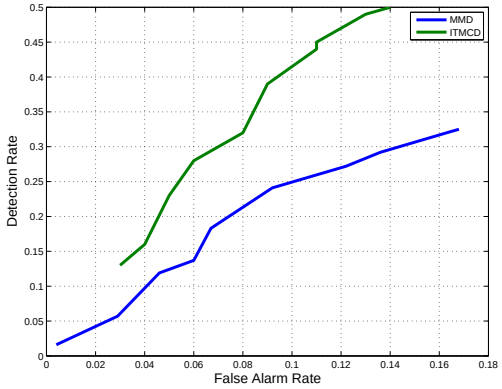

Fig. 2. ROC of correlation change detection for 2D Gaussian data

In this case the ITMCD method was applied in the multidevice mode. ITMCD learns only a threshold for an allowed dissimilarity between keystroke dynamics of a specific user. The method does not learn a specific pattern of keystroke times for each user in the setup time, such adaptation occurs automatically in the run time. Therefore the threshold of dissimilarity between the keystroke times may be learned once per a sensitivity α and window sizes p and f by averaging appropriate thresholds learned for a set of users. Then this threshold (11) may be applied in a run time for other users.

In the data set we had 400 keystroke times sequences for each of 50 users. We divided the users into two subsets. The first subset of 40 users was used for the threshold calibration procedure. The other subset of 10 users were used for emulating an attack in the following way. A user performed a p accesses for a site, then another user enters the site f times. The algorithm is applied to the resulting window of $p + f$ 21D keystroke timings in order to detect a change with a predefined false alarm α. Here we used the symmetrical window sizes $w = p = f$. The kNN parameter was set to $k = 1$. The ITMCD was compared with the MMD method and the results are summarized in Table 2.

Almost for the whole range of tested parameters the performance of ITMCD is significantly better. ITMCD is able to produce a reliable results even for the most challenging short time window cases. From the other hand the MMD method often collapses for small time windows and does not return meaningful results. ITMCD demonstrated an adequate performance for this security application.

5. CONCLUSIONS

In this paper we discussed the analytical aspects of IoT. We introduced a novel information theoretic multivariate change detection algorithm which allows fast online parallel implementation. The algorithm is designed and implemented to satisfy the requirements of the IoT cloud analytics platform.

We provided a theoretical background of the method and performed numerous simulated experiments, followed by a real world data application to the keystroke dynamics security. The algorithm demonstrated a state-of-the-art performance both in terms of higher detection rate and more stable false alarm rate.

6. REFERENCES

[1] P. Middleton, P. Kjeldsen, and J. Tully, "Forecast: the internet of things, worldwide," *Gartner*, 2013.

[2] Q. Wang, S. R. Kulkarni, and S. Verdu, "A nearest-neighbor approach to estimating divergence between continuous random vectors," *IEEE Int. Symp. Information Theory, Seattle, WA*, 2006.

[3] F. Desobry, M. Davy, and C. Doncarli, "An online kernel change detection algorithm," *Trans. Sig. Proc.*, vol. 53, no. 8, pp. 2961–2974, Aug. 2005.

[4] A. Kolmogorov, "Sulla determinazione empirica di una legge di distribuzione," *IG. Ist. Ital. Attuari*, vol. 4, no. 2, pp. 83–91, 1933.

[5] A. Glazer, M. Lindenbaum, and S. Markovitch, "Learning high-density regions for a generalized kolmogorov-smirnov test in high-dimensional data," in *Advances in Neural Information Processing Systems 25*, pp. 728–736. Curran Associates, Inc., 2012.

[6] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *Advances in Neural Information Processing Systems 19*, pp. 513–520. MIT Press, 2007.

[7] Tamraparni Dasu, Shankar Krishnan, Suresh Venkatasubramanian, and Ke Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," *Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*, 2006.

[8] K. Sricharan, R. Raich, and A. O. Hero, "Estimation of nonlinear functionals of densities with confidence," *IEEE Trans. on Inform. Theory*, vol. 58, no. 7, pp. 4135–4159, 2012.

[9] PravinM. Vaidya, "An o(n log n) algorithm for the all-nearest-neighbors problem," *Discrete and Computational Geometry*, vol. 4, no. 1, pp. 101–115, 1989.

[10] K. S. Killourhy and R. A. Maxion, "Comparing anomaly detectors for keystroke dynamics," *IEEE Computer Society Press, Los Alamitos, California*, pp. 125–134, 2009.