ADE: An Ensemble Approach for Early Anomaly Detection

Teodora Sandra Buda[1], Haytham Assem[1], Lei Xu[1]
[1]Cognitive Computing Group, Innovation Exchange, IBM Ireland
{tbuda, haythama, lxu}@ie.ibm.com

***Abstract*—Proactive anomaly detection refers to anticipating anomalies or abnormal patterns within a dataset in a timely manner. Discovering anomalies such as failures or degradations before their occurrence can lead to great benefits such as the ability to avoid the anomaly happening by applying some corrective measures in advance (e.g., allocating more resources for a nearly saturated system in a data centre). In this paper we address the proactive anomaly detection problem through machine learning and in particular ensemble learning. We propose an early Anomaly Detection Ensemble approach, ADE, which combines results of state-of-the-art anomaly detection techniques in order to provide more accurate results than each single technique. Moreover, we utilise a a weighted anomaly window as ground truth for training the model, which prioritises early detection in order to discover anomalies in a timely manner. Various strategies are explored for generating ground truth windows. Results show that ADE shows improvements of at least 10% in earliest detection score compared to each individual technique across all datasets considered. The technique proposed detected anomalies in advance up to ∼16h before they actually occurred.**

I. Introduction

Anomaly detection refers to the problem of finding patterns in data that do not conform to expected behaviour. These non-conforming patterns are often referred to as anomalies, outliers, discordant observations, exceptions, aberrations, surprises, peculiarities or contaminants in different application domains [1]. The ability to discover anomalies within a dataset can have a significant impact in variety of application areas, such as: fraud detection for banking and financial industries, intrusion detection for discovering security threats, health related problems, performance degradation detection, traffic congestion detection and so on. For instance, a failure within a data centre can be considered an anomaly. In spite of processing and storage capabilities of computer system steadily increasing across the years, Figure 1 illustrates that failures are still common in data centres both from hardware and software perspectives, resulting in a discomfort in user experience and naturally in tremendous revenue losses for organisations[1].

Furthermore, *proactive* anomaly detection refers to the problem of detecting in advance an anomaly that will happen in the near future. For instance, a proactive detection of a failure can be of great benefit, leading to proactive fault tolerance [3], such that when a physical machine is suspected of failing in the near future, its VMs can be proactively moved to safer locations thus avoiding potential downtime [4]. Even more, a proactive

[1]http://www.evolven.com/blog/downtime-outages-and-failures-understanding-their-true-costs.html

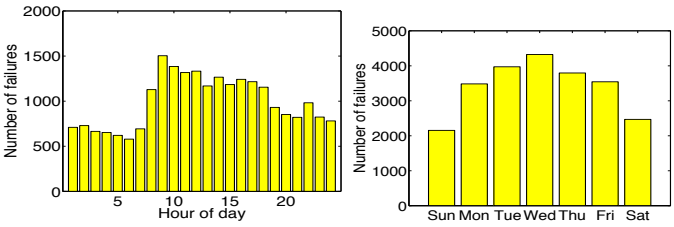


Fig. 1: Number of failures across 22 high-performance computing systems at Los Alamos National Lab (LANL) [2].

detection is only beneficial if the it discovers the anomaly with sufficient time in advance for corrective actions. For instance, in the case of VM proactive migration it is important to discover a potential future failure as soon as possible as the process is typically lengthy and time-consuming. Another important use case is detecting potential network congestions that will impact the traffic flow. For instance, the impact of congestions in the transportation systems is critical [5], especially considering their exigent requirements. Therefore, it is of utmost importance to proactively detect an anomaly as early as possible in order to allow mitigating actions to be taken in advance to reduce their potential impact in the future.

Moreover, machine learning practitioners are often faced with the extremely difficult challenge of choosing between various available machine learning techniques. Similarly in anomaly detection, plenty of techniques have been proposed that are specialised for either a type of dataset or application domain, or type of anomaly to discover (e.g., point, contextual or collective anomalies). Although this enables them to have a wide repertoire of tools available, as the complexity of systems and size of collected data are constantly increasing as well, selecting and tuning techniques manually becomes infeasible. It is a very costly, time-consuming trial-and-error practice that ideally should be avoided.

In this paper, we address the challenges above, and propose ADE, *an Anomaly Detection Ensemble approach*, a proactive anomaly detection ensemble approach that prioritises early discovery of anomalies. The approach learns in time the accuracy of each technique on that particular dataset type. For any incoming data measuring the same metrics as the training dataset, ADE will combine the results from multiple existing anomaly detection techniques based on the prior learning in order to detect an anomaly. The approach generates a weighted

anomaly window and utilizes it as ground truth for training the model. The ground truth window generation is a core contribution of our approach as it prioritises an earlier detection within that window for instance by applying higher weights closer to the beginning of the window. Several strategies are explored for generating the weighted window. Further, the approach uses as input features the scores from multiple existing anomaly detection techniques. Thus, there is no longer need to choose a technique from multiple available ones, as the model learns in time the accuracy of each technique based on the dataset considered and applies accordingly a weight to each of their scores, each contributing to the computation of the overall result. Moreover, an extensive set of experiments are conducted on real-life datasets provided by the Numenta Benchmark [6]. Besides precision and recall, we also evaluate the proposed and compared techniques from an early detection perspective through the proposed ed metric. Results show that our proposed approach outperforms the compared anomaly detection techniques in terms of early anomaly detection, with an average of $\sim$ 7 hours early detection prior to the actual anomaly across all datasets considered.

Assumptions: (1) The training dataset is a labelled dataset as ADE follows a supervised learning approach. As such a labeled dataset is required for training, where an instance is recorded as anomalous or not. (2) As the model employs multiple techniques for obtaining the final score, all individual models need to be trained on the labeled dataset. (3) The ensemble learning is based on gathering and learning on the results of other techniques. As such its scalability in training the model will depend on the scalability of the employed techniques; although this is only for the training phase. Naturally, applying the model for scoring on a new incoming dataset should not be impacted by the training duration.

II. Related Work

Most common existing techniques deployed in real systems employ threshold based methods, which can be categorised into parametric (such as [7]) and non parametric ones (such as [8]). Parametric methods make assumptions of the underlying distribution of the dataset and determine threshold values based on the distribution. However, this assumption is many times unrealistic and violated in practice. Moreover, non parametric methods avoid making such assumptions but determine the threshold in an ad-hoc manner. However, both approaches are generally non realistic, do not adapt to varying patterns in incoming datasets, and often require significant efforts in tuning the threshold value.

Recent efforts revealed systems that employ more than one existing anomaly detection techniques, however they utilise a rather simplistic approach such as a voting mechanism for determining the result (e.g., Skyline [9] declares a metric as anomalous when a consensus of 6 techniques detecting it is reached). Meanwhile, there is another trend in the area, which tries to tackle the big data problem and increase the scalability of solutions. For example, [10] presented a distributed anomaly detection solution based on Hadoop and Weka that removes the centralised failure point and enables real-time analysis for big data;[11] developed a light-weight statistics method based on performance and failure Key Performance Indicators (KPI) to detect abnormal behaviours in cells.It is computationally less expensive compared with machine learning base solution, which enables this solution to be deployed on large netowrks or environment with low processing power.

As there is a wide spread of use of the following anomaly detection techniques in the literature and industry, ADE will utilize their results for training the ensemble model:

1) IBM SPSS Anomaly Detection module[2], referred further as simply SPSS, which employs a clustering based approach for detecting anomalies in a dataset.
2) NuPIC (Numenta Platform for Intelligent Computing), referred further as simply Numenta, is an open source project based on a theory of neocortex called Hierarchical Temporal Memory (HTM). The HTM technique is well suited for online real-time applications and has proven useful in applications such as monitoring server data, geospatial tracking, stock trading and social media.
3) Etsy Skyline [9], which employs several simplistic anomaly detection techniques and calculates an anomaly score through a voting mechanism. Some of the detectors in Etsy Skyline are: deviation from moving average, deviation from a least squares estimate, and deviation from a histogram of past values.
4) AnomalyDetection[3] is an R package for detecting anomalies developed by Twitter, introduced first in [12]. We will refer to this technique further as simply Twitter, which is based on a combination of statistical techniques to robustly detect outliers. Based on the Generalized ESD test, they are combined with robust statistical metrics and piecewise approximation to detect long term trends.

These can be expanded to additional anomaly detection techniques, given that they are scalable and able to be applied on batch and streaming data. Moreover we also compare our results with each of these individual approaches in Section IV.

III. ADE: Early Anomaly Detection Ensemble

A. Preliminaries

This section introduces the notations used throughout the paper, along with their definitions. Let d be a dataset with n features, also referred to as fields: $d = \{f_0, f_1, ..., f_n\}$. We denote by $\|d\|$ the total number of rows (i.e., records) in the dataset d. Let the first field be the index sequence of the dataset, which represents an arithmetic progression with a common difference of 1 (i.e., 0, 1, 2, ...). Moreover, as in this paper we focus on time series datasets, let the second feature be the timestamp field of the dataset. The timestamp field represents a chronological sequence of timestamps, denoted by: $\langle t_0, t_1, ..., t_m \rangle$. Let i denote the index (i.e., position) of the timestamp t_i in dataset d: $pos_d(t_i) = i$.

[2]https://www.ibm.com/support/knowledgecenter/SS3RA7_15.0.0/com.ibm.spss.modeler.help/anomalydetectionnode_general.htm

[3]https://github.com/twitter/AnomalyDetection

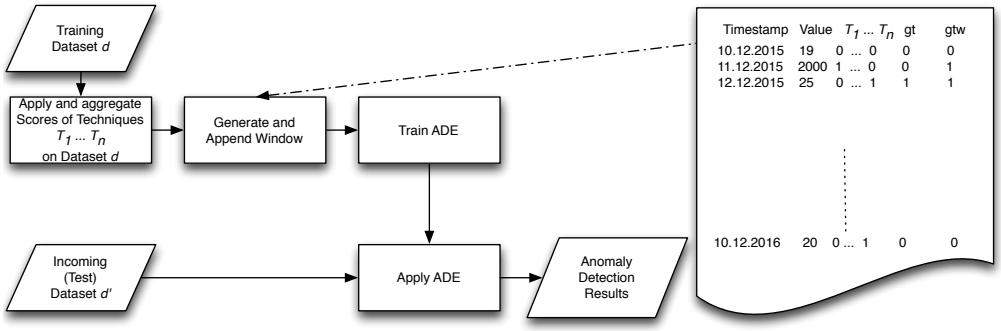


Fig. 2: Anomaly Detection Ensemble approach.

The set of anomalies within a dataset d is denoted by A_d. Anomalies are also referred to as ground truth, their associated field in the dataset as ground truth label, and their index is given by $pos_d(a)$. The ground truth label filed is denoted by $gt = \langle gt_0, gt_1, ..., gt_m \rangle$, where:

$$gt_i = \begin{cases} 0 & \text{if } t_i \notin A_d; \\ 1 & \text{if } t_i \in A_d; \end{cases} \quad ,\forall i \in [1, m] \wedge m = \|d\| \qquad (1)$$

Moreover, we denote by $\|A_d\|$ the total number of anomalies within dataset d. From [6] we adopt the definition of an anomaly window within a dataset d, denoted by aw_d and representing a range of data points centred around a true anomaly. Furthermore, we denote by $\|aw_d\|$ the length of the anomaly window within dataset d, which similarly to [6] is 10% of the size of the dataset, divided by the number of true anomalies contained: $\|aw_d\| = \frac{10\% \cdot \|d\|}{\|A_d\|}$.

The anomaly detection techniques used as input for the ensemble model are denoted by T_j, where $j \in \{1, 2, 3, 4\}$, which correspond to the ones enumerated in Section II. The associated field of T_j represents the anomaly detection labels produced by the technique. We denote by T_j^i the label produced by T_j at index i, where $T_j^i \in \{0, 1\}$, 1 when an anomaly is discovered by T_j at index i, and 0 otherwise . Moreover, $T_j(a)$ returns the earliest timestamp at which anomaly a is discovered by T_j, within its window $aw_d(a)$.

B. Approach

The approach of the ADE is presented in Figure 2 and involves the following steps:

1) Data preparation (presented in Section III-B1).
2) Anomaly window generation to be used as ground truth (presented in Section III-B2).
3) Training the ensemble model using the ground truth window generated field from the prior step.
4) Applying and gathering the results of applying the model on a new incoming or test dataset.

For the ensemble model, we used the $XGBboost$ library, which is an optimized distributed gradient boosting library from the R programming language[4]. The library provides a parallel tree boosting (also known as GBDT, GBM) that is known for being efficient and accurate. For the actual window field used as ground truth for training, a window generation algorithm is used, following one of the strategies described in Section III-B2. It is important to mention that for training the ensemble model we used the scores produced by the techniques T_j, which in the actual implementation are the four described in detail in Section II.

1) Data Preparation:: Given a labeled dataset *d*, the data preparation phase involves three steps: (1) Applying existing anomaly detection techniques, T_j; this will produce (2) The scores of each technique on the given dataset, which will then be (3) Aggregated in a single file for training purposes. This can be observed in the left upper boxes of Figure 2.

2) Anomaly Window Generation:: We devised different strategies for generating anomaly windows fields, referred further as ground truth windows for the ensemble model, in order to investigate their impact on the early detection of anomalies. Some strategies focus on giving higher weights closer to the actual anomaly for improved precision and recall. Others focus on giving higher weights closer to the beginning of the window for earlier detection. In particular, the strategies and associated fields devised are described below:

1) *Ground truth window (gtw)*: This field takes the value of 1 for the entire anomaly window, $aw_d(a)$, and 0 otherwise. Through this approach we investigate whether giving the maximum weight both close to the beginning of the interval and close to the actual anomaly will improve the results.
2) *Earliest detection (gtw_{ed})*: The objective of this approach is to prioritise early detection by training ADE to give preference to the scores given by the technique detecting earliest the anomaly. The field will give a higher probability to the earliest detection by any technique, decreasing until reaching the actual anomaly. The probabilities can be seen as weights, where a higher weight is assigned to the earliest discovery, decreasing until reaching the end of the anomaly window. This associated field is defined as:

$$gtw_{ed}^i = \begin{cases} p & \text{if } t_i \in aw_d \wedge i > pos(a), \forall a \in A_d; \\ 0 & \text{if } t_i \notin aw_d; \end{cases} \qquad (2)$$

[4]https://github.com/dmlc/xgboost

$$\forall 1 \in [1, m] \wedge m = \|d\| \wedge pos(a) = min(pos_d(T_j(a)))$$

where $pos(a)$, and $min(pos_d(T_j(a)))$ represent the index of the earliest detection of a by any of the evaluated techniques T_j. Moreover, the probability p belongs to a decreasing sequence of numbers between 1 and 0.5, such that $gtw_{ed}^{pos(a)} = max(p)$ and subsequently its value decreases until reaching the end of the interval aw_d. The reason for choosing 0.5 is to assign more weight within the window. Experiments with a larger range down to 0.1 were also performed; however 0.5 for the lower limit produced better results.

3) *Before earliest detection ($gtw_{bf_{ed}}$)*: This approach verifies whether the scores given by the techniques before detecting the anomaly could be used for ADE to detect the anomaly before it is observed by the techniques. Thus, we start with higher weights before the anomaly was detected by any technique. Similarly to the approach above it uses probabilities between 1 and 0.5, with the exception that $pos(a)$ is 10% of aw_d earlier than $min(pos_d(T_j(a)))$.
4) *Ground truth label (gtw_{gtl})*: The objective of this approach is to prioritise a more accurate detection by training ADE to give preference to the scores given by the techniques detecting the anomaly closest to the ground truth labels (i.e., $pos_d(a), \forall a \in A_d$. In this case, $pos(a) = pos_d(a)$.
5) *Before ground truth label ($gtw_{bf_{gtl}}$)*: The objective of this approach is to prioritise a both accurate and early detection by training ADE to give preference to the scores given by the techniques detecting the anomaly earlier but closest to the ground truth labels (i.e., $pos_d(a), \forall a \in A_d$. In this case, $pos(a)$ is 10% of aw_d earlier than $pos_d(a)$.
6) *Discrete (gtw_{discr})*: This approach prioritises a more accurate detection by assigning $p = 1$ only when when the anomaly occurs, i.e., $i = pos_d(a)$, and 0 otherwise.
7) *Discrete probabilistic ($gtw_{discr_{prob}}$)*: This approach prioritises a correct detection by assigning $p = 1$ only when the anomaly occurs, i.e., $i = pos_d(a)$, and 0.5 otherwise.

IV. Evaluation

A. Datasets used

1) Numenta Anomaly Benchmark: The NAB benchmark[5] provides a set of real-world time-series datasets, denoted by D (58 files). These datasets are labeled, i.e., contain a field that can be 1 or 0 depending on whether the record is an anomaly or not, respectively. The NAB benchmark also compares Numenta with Twitter and Skyline. In our evaluation we also compare against these three techniques, and in addition to SPSS.

2) Splitting the data: The datasets measuring the same metric, such as *cpu*, *rds-cpu*, *network*, are merged into one dataset for each metric in order to enlarge the training dataset available for the ensemble model. Moreover, in order to ensure that the training and testing data is sufficient, for this evaluation we considered only datasets spreading across more than 1 file, as generally each file contains 1 or 2 true anomalies, with only a few containing more. We split the data into training and testing sets, as typically done in the community, using 80% of the data for training and 20% for testing.

[5]https://github.com/numenta/NAB

B. Metrics

In this section, we present the metrics used for the evaluation of ADE. We follow a similar approach to the NAB benchmark for evaluating each technique by considering the early detection aspect; however, the NAB score proposed in [6] combines the precision, recall and the early detection of the techniques into one single score. In contrast to this, we evaluate all techniques from each of these metrics separately in order to get a more thorough evaluation.

1) Early Detection (absolute time and rank): Firstly, we compare the techniques across all anomalies discovered in terms of absolute time and index of detection.

2) Early Detection (ed): The ed score evaluates how early an anomaly a was detected relative to the anomaly window. The ed score is between 0 and 1, where 1 represents that the anomaly a was discovered at the beginning of the interval and 0 at the end. The ed score is relative to the time interval, i.e., a 10% increase in ed means that a technique detected an anomaly 10% of the time interval earlier. For instance, considering an anomaly window of 200 indexes (most common), a 5min step between two consecutive indexes and an ed of 10% more, leads to 100min earlier detection. As datasets might use different time steps between t_i and t_{i+1}, the ed uses the indexes of the timestamp for calculating the score (i.e., index of t_b is b, and index of t_e is e).

The ed score of a technique T_j for an anomaly a within a window $aw_d(a) = [t_b, t_e]$ and d is defined as:

$$ed_a(T_j) = \begin{cases} 1 - \frac{pos_d(T_j(a)) - b}{e - b} & \text{if } a \in A_d^{T_j}; \\ 0 & \text{if } a \notin A_d^{T_j}; \end{cases} \quad (3)$$

where $A_d^{T_j}$ represents the anomalies discovered by T_j in dataset d, $T_j(a)$ represents the time of the first detection by technique T_j of the anomaly a in dataset d, and $pos_d(T_j(a))$ represents the index of that timestamp. Subsequent detections of anomaly a by T_j within the window $aw_d(a)$ are ignored as this metric only evaluates the anomalies' earliest detection.

The average early detection score of T_j represents an average of the ed score for all anomalies across all datasets.

3) Precision and recall:: Similarly to NAB approach, if T_j discovers an anomaly $a \in A_d$ within its window, $aw_d(a)$, but not exactly at $pos_d(a)$ then it is considered as a true anomaly discovered, both for precision and recall.

C. Results

1) Early Detection: We begin by presenting for each technique the detection time and rank of all anomalies across all datasets used for testing purposes. A filtered table of these results, showing only the top performers of ADE is presented in Table I. Moreover, as ADE_{window} and ADE_{ed} outperform

Dataset folder	Anomaly: Date (Index)	Numenta: Detection date (Index) Rank	Skyline: Detection date (Index) - Rank	SPSS: Detection date (Index) - Rank	Twitter: Detection date (Index) - Rank	XGB_window: Detection date (Index) - Rank	XGB_earliest: Detection date (Index) Rank
realAWSCloudwatch	26/02/2014 22:05 (3547)	26/02/2014 22:10 (3548) - 3rd	**26/02/2014 22:05 (3547) - 2nd**	*26/02/2014 14:25 (3455) - 1st*	**26/02/2014 22:05 (3547) - 2nd**	26/02/2014 22:10 (3548) - 3rd	26/02/2014 22:15 (3549) - 4th
realAWSCloudwatch	27/02/2014 17:15 (3777)	**27/02/2014 17:20 (3778) - 2nd**	NA	*27/02/2014 08:55 (3677) - 1st*	NA	27/02/2014 17:45 (3783) - 3rd	NA
realAWSCloudwatch	19/02/2014 19:10 (5528)	*19/02/2014 20:10 (5540) - 1st*	NA	20/02/2014 03:10 (5624) - 4th	20/02/2014 03:10 (5624) - 4th	**19/02/2014 20:15 (5541) - 2nd**	**19/02/2014 20:15 (5541) - 2nd**
realTweets	03/03/2015 21:07 (1433)	03/03/2015 21:12 (1434) - 4th	**03/03/2015 05:02 (1240) - 2nd**	04/03/2015 00:37 (1475) 6th	03/03/2015 21:07 (1433) - 3rd	*03/03/2015 04:52 (1238) - 1st*	NA
realTweets	09/03/2015 17:32 (3118)	NA	**09/03/2015 15:57 (3099) - 2nd**	09/03/2015 19:57 (3147) - 3rd	NA	*09/03/2015 06:47 (2989) - 1st*	NA
realTweets	16/03/2015 02:57 (4959)	**15/03/2015 16:17 (4831) - 2nd**	*15/03/2015 16:12 (4830) - 1st*	NA	*15/03/2015 16:12 (4830) - 1st*	*15/03/2015 16:12 (4830) - 1st*	*15/03/2015 16:12 (4830) - 1st*
realTweets	31/03/2015 03:27 (9285)	30/03/2015 18:02 (9172) - 3rd	**30/03/2015 17:57 (9171) - 2nd**	30/03/2015 18:52 (9182) 5th	**30/03/2015 17:57 (9171) - 2nd**	*30/03/2015 11:57 (9099) - 1st*	30/03/2015 18:07 (9173) - 4th
realTweets	05/03/2015 19:47 (17895)	05/03/2015 19:52 (17896) 5th	*05/03/2015 18:17 (17877) - 1st*	05/03/2015 19:47 (17895) - 3rd	**05/03/2015 18:22 (17878) - 2nd**	05/03/2015 19:47 (17895) - 3rd	05/03/2015 19:47 (17895) - 3rd
realTweets	11/03/2015 20:57 (19637)	11/03/2015 21:02 (19638) - 3rd	**11/03/2015 20:57 (19637) - 2nd**	**11/03/2015 20:57 (19637) - 2nd**	**11/03/2015 20:57 (19637) - 2nd**	*11/03/2015 09:22 (19498) - 1st*	**11/03/2015 20:57 (19637) - 2nd**
realTweets	01/04/2015 21:57 (25697)	NA	*01/04/2015 18:27 (25655) - 1st*	NA	*01/04/2015 18:27 (25655) - 1st*	NA	NA
realTweets	08/04/2015 04:52 (27508)	**07/04/2015 23:57 (27449) - 2nd**	**07/04/2015 23:57 (27449) - 2nd**	*07/04/2015 23:52 (27448) - 1st*	*07/04/2015 23:52 (27448) - 1st*	**07/04/2015 23:57 (27449) - 2nd**	**07/04/2015 23:57 (27449) - 2nd**
realKnownCause	17/07/2014 09:50 (1287)	*17/07/2014 16:30 (1330) - 1st*	NA	NA	NA	NA	NA
realAdExchange	14/07/2011 13:00 (325)	*14/07/2011 14:00 (326) - 1st*	NA	NA	NA	NA	NA
artificialWithAnomaly	11/04/2014 00:00 (2880)	**11/04/2014 00:05 (2881) - 2nd**	NA	NA	*10/04/2014 15:30 (2778) - 1st*	**11/04/2014 00:05 (2881) - 2nd**	*10/04/2014 15:30 (2778) - 1st*

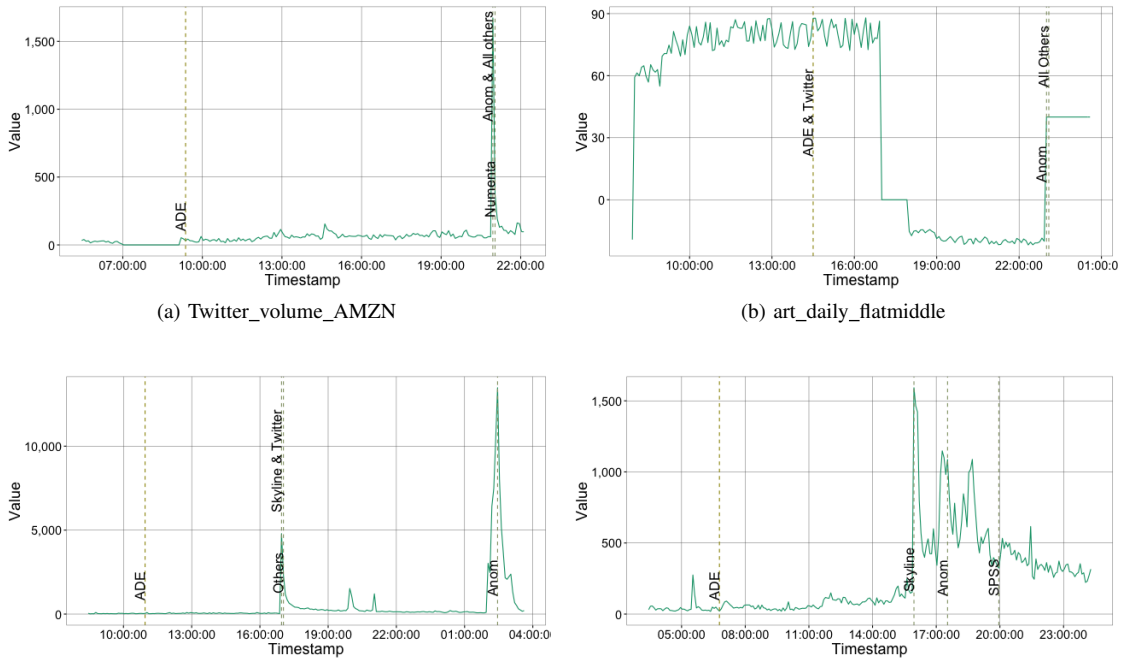
TABLE I: Earliest detection for all anomalies that have been detected by at least two technique across all testing datasets. First ranked detections are illustrated in green and italic font. Second ranked detections are illustrated in orange and bold font.

the other ADE techniques in terms of earliest detection, we only present the results of these techniques in the table. As it can be observed ADE_{window} is ranked 1^{st} in 5 out of 14 cases, in other 3 cases as 2^{nd} and other 3 as 3^{rd}. Twitter anomaly detector is the 2^{nd} performer, with 4 discoveries ranked as 1^{st} and 3 as 2^{nd}, and 1 as 3^{rd}. More interesting is to observe the difference between the detection indexes. For instance, even though ADE_{window} is ranked 2^{nd} for the 3^{rd} anomaly, the difference between its detection index and Numenta's, which is ranked 1^{st}, is of just 1 position.

We expand on four of these cases in Figure 3, in particular to show how much in advance ADE discovered the anomaly before other techniques. We observe in Figure 3a that only ADE_{window} managed to detect the anomaly earlier than it actually occurred. In particular, it detected the anomaly $\sim$ 12h earlier. ADE_{window} used as ground truth the entire anomaly window, which starts much before the anomaly actually occurred (in this case 3h), and using the prior learning on the techniques' scores for previous anomalies, it was able to detect it in advance, although each individual technique did not label it as anomaly. Moreover, all others detected the anomaly at exactly the time it occurred, except for Numenta, which detected the anomaly 5min later. Figure 3b shows that only ADE_{ed}, and Twitter managed to detect the anomaly earlier than it occurred, by 8h and a half. All others that managed to detect it, ADE_{window}, Numenta and a few other ADE approaches had done it only 5min after it occurred. Furthermore in Figure 3c we observe that all techniques managed to detect the anomaly earlier than it occurred by $\sim$ 9h before. However, ADE_{window} detected the anomaly $\sim$ 15h before it occurred. Figure 3d presents how Skyline and ADE were the only ones that detected the anomaly before it had occurred; ADE by $\sim$ 11h, while Skyline $\sim$ 1h and a half in advance. Only SPSS managed to detect this anomaly as well; however, $\sim$ 2h and a half after.

The average ed scores are presented in Figure 4a. We observe that ADE_{window} outperforms the other techniques by at least 10% (between ADE_{window} and Numenta). Moreover, we observe that other ADE strategies are typically outperformed by the compared techniques in terms of ed score, with a difference between 2% and 9% for ADE_{ed}, which follows ADE_{window} for this metric; although, we observed before that across different anomalies ADE_{ed} had discovered the anomaly earlier than the other compared techniques and ADE strategies. $ADE_{discr_{prob}}$ is not far behind ADE_{ed}, with 1% difference.

2) Precision and recall: Furthermore, in terms of precision, we observe in Figure 4b that, as expected, the techniques with a strategy revolving around the ground truth label outperform the others, i.e., $ADE_{bf_{gtl}}$ and ADE_{gtl}. $ADE_{bf_{gtl}}$ shows better results ranging from 3% to 23% more precision than the compared techniques. As the precision measures the true anomalies discovered over the total number of anomalies discovered, this shows a lower rate of false positives for $ADE_{bf_{gtl}}$ and ADE_{gtl} than the other techniques. In terms of recall, we observe in Figure 4c that Numenta outperforms all the other techniques, and in particular ADE_{window} by



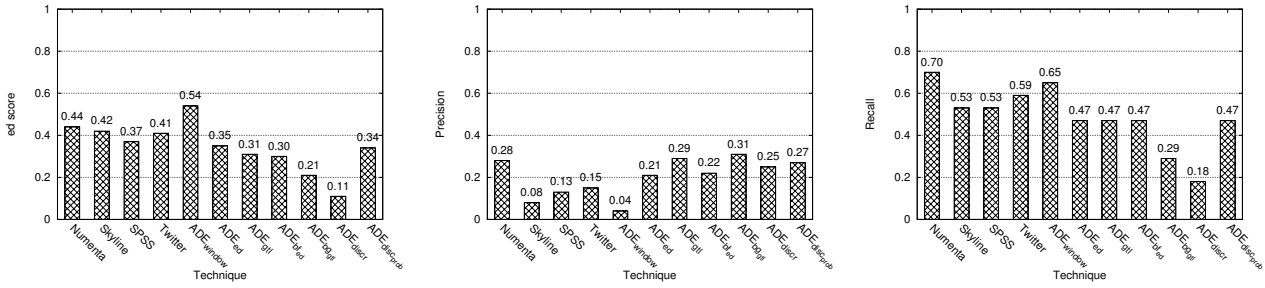
(a) Twitter_volume_AMZN

(b) art_daily_flatmiddle

(c) Twitter_volume_AAPL

(d) Twitter_volume_AAPL

Fig. 3: Early detection of ADE compared to other techniques on different datasets.



(a) Average ed scores.

(b) Precision.

(c) Recall.

Fig. 4: Precision, recall and early detection results for all techniques.

5%. This reflects that ADE_{window} discovers slightly less anomalies than Numenta. However, ADE_{window} outperforms all other techniques with 6% to 12%, except for Numenta.

To summarize, depending on the context, different ADE strategies can be employed to maximize the results for a particular concern: earliest detection, precision or recall. For ed score, ADE_{window} outperforms all others, while for precision, $ADE_{bf_{gtl}}$ shows best results. Numenta shows better results in terms of recall, while ADE_{window} outperforms the rest of the compared techniques.

V. Conclusion

In this paper, we proposed an anomaly detection ensemble approach ADE that prioritises early discovery of anomalies. Given a dataset, the approach computes the results from multiple existing anomaly detection techniques and learns their accuracy on the dataset. For any incoming data measuring the same metric as the analysed dataset, ADE will combine results from the techniques employed in order to detect an anomaly based on prior learning. Seven different window generation strategies have been explored to maximize the results of different evaluation metrics such as early detection or precision, recall. In order to minimize the number of false positives, strategies that revolve around the ground truth label show better results in precision. Moreover, strategies that employ a weighted window starting from an earlier index such as ADE_{ed}, and in particular ADE_{window} show the best results in earliest detection. Results show that ADE outperforms the compared techniques in early detection across the datasets considered, with up to ∼16h earlier and on average ∼6h earlier than the anomaly actually occurred.

VI. Acknowledgments

This work was partly supported by the EC project CogNet, 671625 (H2020-ICT-2014-2, Research and Innovation action).

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[2] B. Schroeder and G. Gibson, "A large-scale study of failures in high-performance computing systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 7, no. 4, pp. 337–350, 2010.

[3] B. Nicolae and F. Cappello, "A hybrid local storage transfer scheme for live migration of i/o intensive workloads," in *Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing*. ACM, 2012, pp. 85–96.

[4] A. B. Nagarajan, F. Mueller, C. Engelmann, and S. L. Scott, "Proactive fault tolerance for hpc with xen virtualization," in *Proceedings of the 21st annual international conference on Supercomputing*. ACM, 2007, pp. 23–32.

[5] X. Ma, H. Yu, Y. Wang, and Y. Wang, "Large-scale transportation network congestion evolution prediction using deep learning theory," *PloS one*, vol. 10, no. 3, p. e0119044, 2015.

[6] A. Lavin and S. Ahmad, "Evaluating real-time anomaly detection algorithms – the numenta anomaly benchmark," in *14th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2015, pp. 38–44.

[7] Y. Wang, E. W. Ma, T. W. Chow, and K.-L. Tsui, "A two-step parametric method for failure prediction in hard disk drives," *IEEE Transactions on industrial informatics*, vol. 10, no. 1, pp. 419–430, 2014.

[8] M. Bertini, A. Del Bimbo, and L. Seidenari, "Multi-scale and real-time non-parametric approach for anomaly detection and localization," *Computer Vision and Image Understanding*, vol. 116, no. 3, pp. 320–329, 2012.

[9] "A. Stanway. (2013) etsy/skyline," https://github.com/etsy/skyline.

[10] B. Cui and S. He, "Anomaly detection model based on hadoop platform and weka interface," in *10th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, July 2016, pp. 84–89.

[11] L. Bodrog, M. Kajo, S. Kocsis, and B. Schultz, "A robust algorithm for anomaly detection in mobile networks," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sept 2016, pp. 1–6.

[12] A. Kejariwal, "Introducing practical and robust anomaly detection in a time series," *Twitter Engineering Blog. Web*, vol. 15, 2015.