Anomaly Detection in Data Center IT Infrastructure using Natural Language Processing and Time Series Solutions

Elisabetta Ronchieri[1,2]
L. Giommi[1], D. C. Duma[1], A. Costantini[1], D. Salomoni[1], F. Pacinelli[2]

[1]INFN CNAF, [2]Univ. Bologna

ISGC 2023, March 19-24, 2023, Taipei, March 22, 2023

Table of Contents

INFN CNAF Data Center in Bologna, Italy

- It focuses on technological development and knowledge transfer to heterogeneous experiments, national and European projects, and industry (whenever possible).

Image provided by Pier Paolo Ricci, INFN CNAF.

Facts

- over 1700 active users
- over 50 experiments
- over 65,000 CPU cores
- 41 PB of disk
- 98 PB of tapes

INFN CNAF Data Center Pillars

Pillars already supported

- ✓ Connectivity:
  to cloud and infrastructure for data transmission
- ✓ Security:
  for data and privacy protection
- ✓ ...

New Pillar

- Intelligence
  - through infrastructure and algorithms in order to extract value from service and machine data and convert them into useful information
  - perform predictive maintenance

Table of Contents

Anomaly Detection through Logs and Monitoring Metrics

Facts

- large amount of data
- over 1,000 different services running
- data flowing 24/7

Goal

- perform predictive maintenance

How?

- define intelligence based on logs and monitoring metrics in order to detect anomalies and properly intervene

Table of Contents

Data Sources

Time range for this study: 6th June, 2020 - 21st July, 2021

Log Files

- 40 kinds of log files
- 1143 distinct machines

Monitoring Metrics

- 18 metrics
- 3 categories of monitoring metrics: io-stat, load average, and memory

Log Files and Monitoring Metrics

Log Files

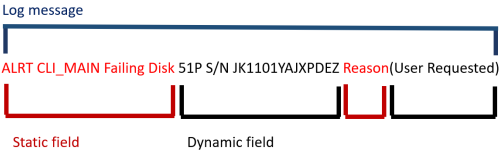
- Logs are created by Linux system services and contain semi-structured texts.
- They are used to keep records of occurring events, analyze and debug system failures.
- The services may be highly verbose [Dec+20b].
- On a single machine, there are several different log files.
- Each log file corresponds to a distinct service.
- The log files have different formats.

Monitoring Metrics

- Each machine usually runs different services and we can collect information on:
  1. **memory** statistics, representing memory usage;
  2. the **load** the machine has been under, averaged over multiple time frames (i.e., **load_avg**)ç
  3. the **central processing unit** (CPU) statistics and **input/output** (I/O) statistics (i.e., **io_stat**).

An Example of the ALRT Service Log File

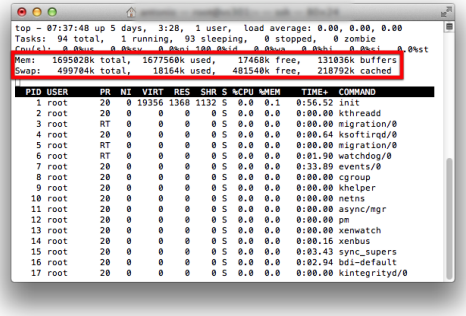
date	time	timestamp	hostname	ip	process_name	msg
2020/08/11	11:57:17	1597139837.0	ddn-04-a.cnaf.infn.it	*	ALRT	ALRT CLI_MAIN Failing Disk 51P S/N JK1101YAJXPDEZ Reason(User Requested)
2020/07/08	14:36:39	1594211799.0	ddn-04-a.cnaf.infn.it	*	ALRT	ALRT AVR_MON Left Power Supply Failure
2020/07/08	14:36:39	1594211799.0	ddn-04-a.cnaf.infn.it	*	ALRT	ALRT AVR_MON Left Side AC Line Low
2020/07/27	16:24:23	1595859863.0	ddn-04-a.cnaf.infn.it	*	ALRT	ALRT CLI_MAIN Failing Disk 19G S/N JK1101YAJSDYXV Reason(User Requested)
2020/07/25	12:24:08	1595672648.0	ddn-04-a.cnaf.infn.it	*	ALRT	ALRT DC_REC Failing Disk 42P S/N JK1101YAKAB69V Reason(IO Timeout)



- **date**, **time** and **timestamp** express when the instance has been registered in the local time zone.
- **hostname** and **ip** columns provide information on the machine and network on which the event occurred, respectively.
- **process_name** is the name of the service (log file).
- **msg** reports the textual message.

An Example of the memory.used Monitoring Metric

name	tags	time	domain	duration	metric	value
memory.used	host=api-int.cnsa.cr.cnaf.infn.it	1,62643E+18	cnsa.cr.cnaf.infn.it	0.22133	metrics-memory	1,37128E+09
memory.used	host=api-int.cnsa.cr.cnaf.infn.it	1,62644E+18	cnsa.cr.cnaf.infn.it	0.23058	metrics-memory	1,39689E+16
memory.used	host=api-int.cnsa.cr.cnaf.infn.it	1,62644E+18	cnsa.cr.cnaf.infn.it	0.22633	metrics-memory	1,46820E+16
memory.used	host=api-int.cnsa.cr.cnaf.infn.it	1,62644E+18	cnsa.cr.cnaf.infn.it	0.21558	metrics-memory	1,49160E+16
memory.used	host=api-int.cnsa.cr.cnaf.infn.it	1,62645E+18	cnsa.cr.cnaf.infn.it	0.21825	metrics-memory	1,49583E+16



- **time** expresses when the metric has been registered in the local time zone.
- **tags** and **domain** provide information about the machine values are collected.
- **name** and **metric** provide the metric name and category.
- **value** contains the metric value.

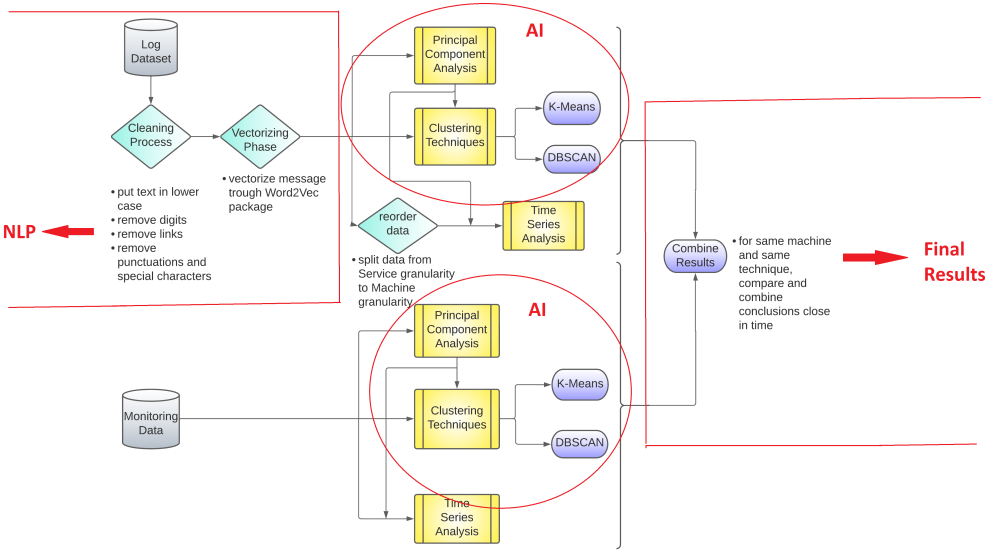
Table of Contents

Main Challenges

Working with

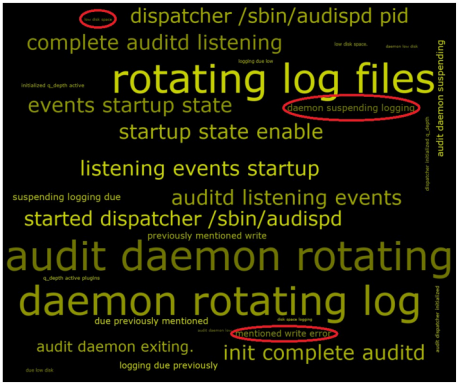
- Different types of data (textual and numerical);
- Thousands of machines to be analysed;
- A completely unsupervised task.

Methodology

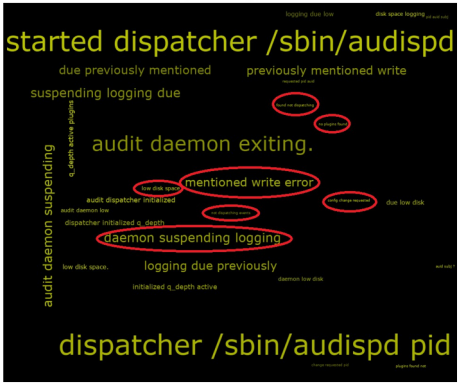


Anomaly Dictionary

30 most frequent trigrams in auditd

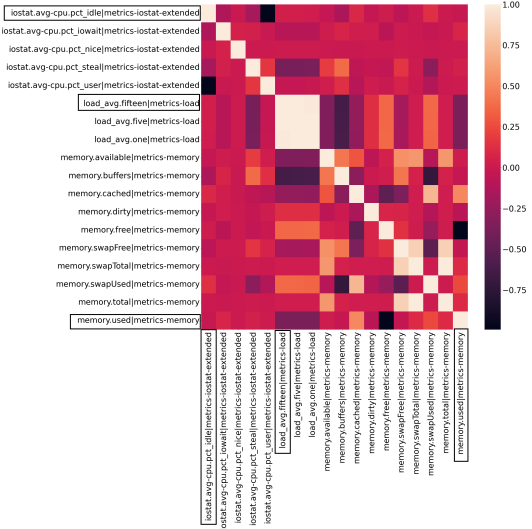


30 least frequent trigrams in auditd



Approach: The red ⬭ highlights terms that may be likely anomalies [VRC22].

Correlation between Monitoring Metrics



- 18 monitoring metrics are correlated.
- Heatmap on the left shows how each pair of metrics is correlated.
- Metrics are both positively and negatively correlated.
- Selected one metric per class:

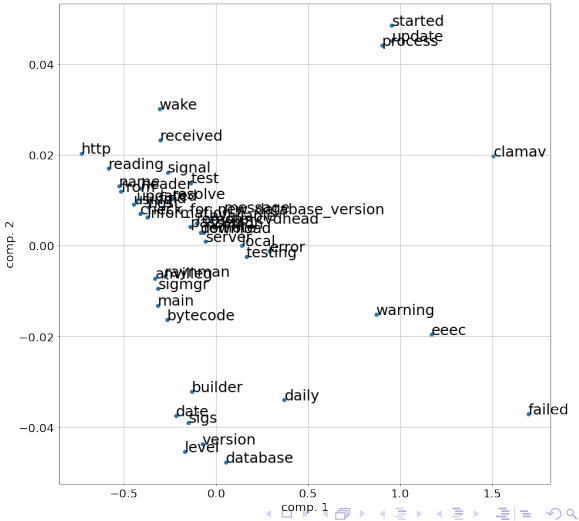
  iostat.avg-cpu.pct_idle,
  load_avg.fifteen,
  memory.usage

Principal Component Analysis (PCA)

PCA

- It is used to perform dimensionality reduction.
- PCA with reconstruction error RE allows us to recognize an anomalous observation.
- Approach: RE is larger for uncommon terms, so less observed observations.

Image shows a dimensional reduction to 2 components.



Clustering Algorithms

Density-Based Spatial Clustering of Applications with Noise

- DBSCAN builds clusters from the highly populated area of observations, close at most an *epsilon* value from each other.
- *epsilon* is computed performing parameter tuning
  - for log data, on overall distances between word-vectors
  - for monitoring metrics, by means of an elbow curve
- Approach: non-anomalous observations are expected to be concentrated closer to each other.

K-means

- K-means is a simple approach to partition data into K distinct, non-overlapping clusters.
- The only parameter is K that is computed performing parameter tuning.
- The observations have been partitioned among K clusters and then considered as potential anomalies provided belonging to the least populated cluster.
- Approach: expected large distances between anomalies and non-anomalies.

Anomaly Score (AS) - for log data

- For the entire message, AS is computed by doing the average value of the labels corresponding to the words that are part of the message.
- As this work is unsupervised, this process allows us to give a continuous measure of anomalies rather than a purely binary value, i.e. anomalous-non anomalous.
  - The binary value 0 or 1 for word represents respectively non-anomalous and anomalous terms.
- In the following, the words in red are the words identified as anomalous terms.

daily database available for update (local version: 26052, remote version: 26053)

'daily', 'database', 'available', 'for', 'update', 'local', 'version', 'remote', 'version'

$$1 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 = 2/9 = 0.222$$

Time Series Analysis - Meam and Variance Outlier Detection

For log data

- Messages have been vectorized and word-vectors from each message have been averaged to get the vector belonging to the message.
- The difference between consecutive vector-messages has been calculated (measured in terms of Euclidean distance between vectors), thus obtaining a time-series of the differences between messages.

For monitoring data

- metrics values have been standardized as the orders of magnitudes of the various metrics are really different from each other.

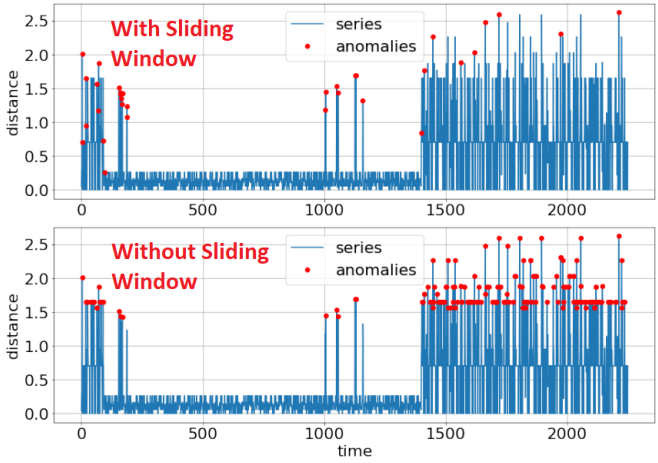
Approach: an interval of non-anomalous values is defined, including all the observations whose values are at most s standard deviations σ away from the value of mean μ:

threshold $= \mu + s\sigma$, $s \in [1, 3]$.

The sliding windows process is also considered to take overlapping subsets and detect anomalies.

An example of Mean and Variance Outlier Detection

A window size of 20 observations and tolerance of 2 standard deviations on a specific machine, i.e. **cloud-ctrl01**.



Validation Criteria

For log data

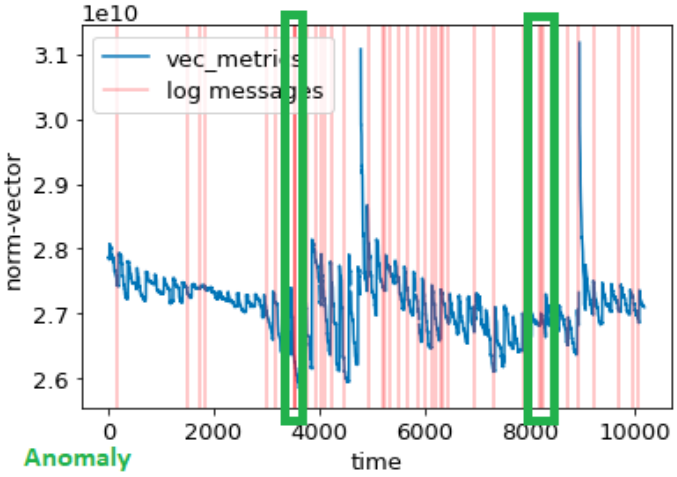
1. the anomaly score AS for each message has been considered, particularly the most recurring words in log messages with high/low AS;
2. a similarity percentage of more than 20%/25% has been considered as an alarming sign.

For monitoring data

1. the Mann-Whitney test has been used to check for significant differences between anomalous and non-anomalous classified observations for every metric.

Table of Contents

An Example of Log and Monitoring Occurrences for a Given Machine



Examples of Combining Results

Service	Technique	Parameter	% of correspondence
auditd	DBSCAN	eps = 0.1	50%
	K-Means	n_clusters = 3	97.5%
	PCA	thresh. = 0.85%	20%
smartd	DBSCAN	eps = 0.1	55%
	K-Means	n_clusters = 3	53%
	PCA	thresh. = 0.85%	55%

- Correspondence has been verified by checking logs and monitoring data occurring at less than 900 (15 minutes) seconds of difference.
- Anomalous and non-anomalous events are, at least to some extent, consistent in time.

Table of Contents

Discussions and Future Works

Discussions

- For log data, DBSCAN, K-means and Principal Component Analysis were very effective.
- For monitoring data, time series analysis provided interesting results, while treating messages as time-series observations did not show very meaningful results.
- We have got over 50% of correspondence between anomalous found in log files and monitoring metrics.

Future Works

- To analyse in more detail the anomalies' nature in order to perform a more precise identification of the root causes of anomalies.
- To develop an advanced model, e.g. a deep learning one, that uses both types of data with respect to timestamp, machine name and network info.

Table of Contents

Q&A

- Elisabetta Ronchieri, elisabetta.ronchieri@cnaf.infn.it
- Thanks to Diego Michelotto and Antonio Falabella, INFN CNAF, for their support in collecting monitoring metrics.

References I

[Dec+20a] Leticia Decker et al. "Real-Time Anomaly Detection in Data Centers for Log-based Predictive Maintenance using an Evolving Fuzzy-Rule-Based Approach". In: (Apr. 2020). arXiv: 2004.13527 [cs.AI].

[Dec+20b] Leticia Decker et al. "Real-time anomaly detection in data centers for log-based predictive maintenance using an evolving fuzzy-rule-based approach". In: *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE. 2020, pp. 1–8.

[Gee21] GeeksforGeeks. *iostat command in Linux with examples*. www.geeksforgeeks.org/iostat-command-in-linux-with-examples/". Online; accessed 08 June 2022. Dec. 2021.

[K] Mathangi K. *Load average: What is it, and what's the best load average for your Linux servers?* https://www.site24x7.com/blog/load-average-what-is-it-and-whats-the-best-load-average-for-your-linux-servers. Online; accessed 12 July 2022.

References II

[Kap] Vladimir Kaplarevic. *How to Check Memory Usage in Linux, 5 Simple Commands*. https://phoenixnap.com/kb/linux-commands-check-memory-usage. Online; accessed 12 July 2022.

[KBR16] Tom Kenter, Alexey Borisov, and Maarten de Rijke. "Siamese CBOW: Optimizing Word Embeddings for Sentence Representations". In: (2016).

[VRC22] L. Viola, E. Ronchieri, and C. Cavallaro. "Combining Log Files and Monitoring Data to Detect Anomaly Patterns in a Data Center". In: *Computers* 11.117 (2022). DOI: https://doi.org/10.3390/computers11080117.

Table of Contents

A Subset of Monitoring Metrics

Category	Metric	Description
load_avg [K]	one	load average value of the machine every minute
	five	load average value of the machine every five minutes
	fifteen	load average value of the machine every fifteen minutes
memory [Kap]	available	estimation of how much memory is available for starting new applications, without swapping
	buffers	memory reserved by the operating system to allocate as buffers when the process needs them
	cached	recently used files stored in RAM
	dirty	memory waiting to be written back to disk
	free	unused memory
	total	total installed memory on the device
	used	memory currently in use by running processes
iostat [Gee21]	pct_idle	the percentage of time that the CPU or CPUs were idle and the system did not have an outstanding disk I/O request.
	pct_iowait	the percentage of the time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request
	pct_nice	the percentage of CPU utilization that occurred while executing at the user level with a nice priority
	pct_user	the percentage of CPU being utilization that while executing at the user level

Useful Parameters

Parameters

- **Dim. reduction**: if applicable, whether or not PCA was applied on data to reduce their dimension
- **N. comps**: if applicable, (whether dimensionality reduction was applied), the number of principal components obtained.
- **N. clusters**: if applicable, the number of clusters defined for partitioning.
- **Window size**: if applicable, the length of the sliding windows (measured in a number of observations per partition).
- **Epsilon**: Estimating the Epsilon was very problematic as, potentially, every machine has its own optimal epsilon, whose computation is pretty tedious. The percentages refer to the quantile of the series obtained from the distance of all the observations (words) between each other.
- **Tolerance**: Tolerance has actually been different yet depending on the techniques in which it was adopted.

N.A.: Not Applicable, so that such parameter could not, or simply was not, specified in performing a certain algorithm.

Hyperparameters Tuning with Grid Search Analysis Model

Algorithm	**Dim. reduction**	**N. comps**	**N. clusters**	**Window size**	**Epsilon**	**Tolerance**
DBSCAN	TRUE, FALSE	2, 3, 10, 20	*variable*	*N.A.*	0.05%, 0.1%, 0.25%, 0.5%	*N.A.*
K-Means	TRUE, FALSE	2, 3, 10, 20	2, 3	*N.A.*	*N.A.*	*N.A.*
PCA Decomposition & Reconstruction	*implicit*	2, 3, 10, 20, 30	*N.A.*	*N.A.*	*N.A.*	0.75, 0.85, 0.95 (measured as the quantiles of the error vector)
Time-Series Outlier Detection	*N.A.*	*N.A.*	*N.A.*	0, 10, 30, 60 (0 means no window partition)	*N.A.*	2, 3 (the k number of σ away from μ in the outlier detection alg.)