Google Cloud Platform

Log Analysis

for CERN's file transfer service

Authors: Giovanni Marchetti, Google	Version: 0.2	Date: 12/3/2021
Reviewers: Panos Paparrigopoulos, CERN Federica Legger, CERN Alessandro Di Girolamo, CERN Grazia Frontoso, Google Alex Schroeder, Google	0.2	

Summary

Anomaly detection techniques designed for data streams over dynamic graphs have shown promising results in identifying patterns in time and space within CERN's transfer service logs. Topic modelling over the content of the logs has found several clusters and most relevant terms therein.
The combination of the two techniques can be a powerful tool for CERN to detect, prioritize and diagnose problems in their transfer service. Further research is warranted to perfect the approach.

Problem Definition

Given a set of more than 69 million log entries from CERN's file transfer service, we were asked to identify patterns that may indicate problems and to analyze the text of such entries. The end goal of this proof-of-concept study is to help the IT support team identify critical events and prioritize resource allocation.

Data Exploration

An initial exploration of the dataset highlighted that most errors occurred from 6th to 11th October 2019. Their distribution was not uniform across categories, with a majority attributable to two of those:

- Communication_error_on_send
- No_such_file_or_directory

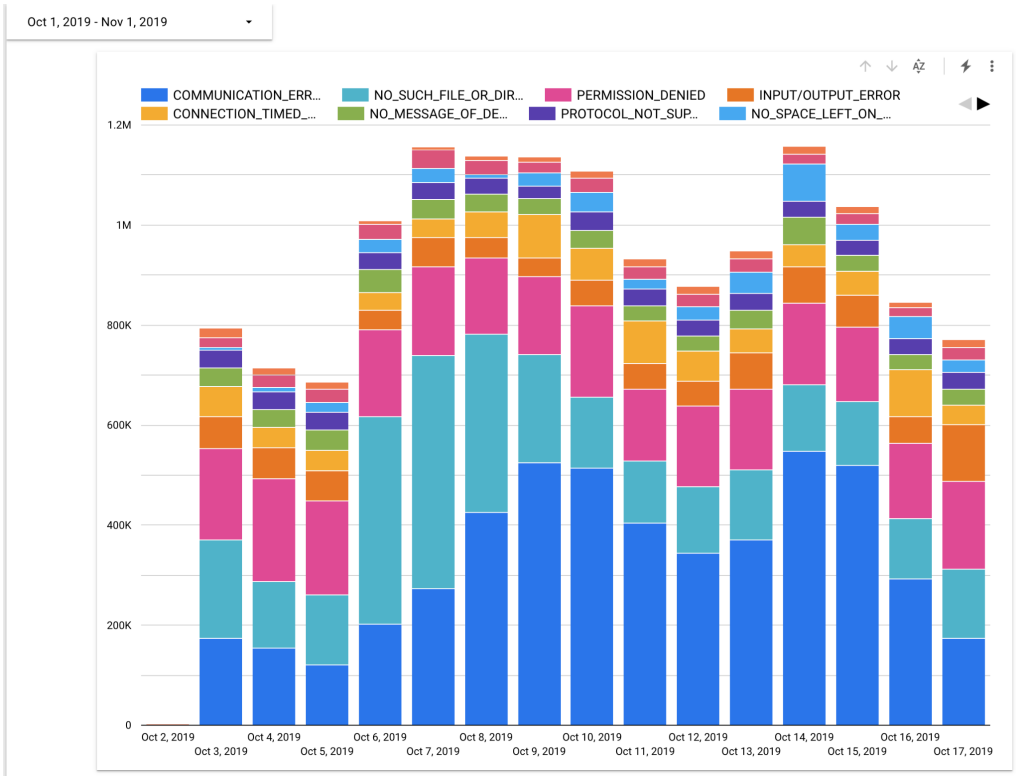


Figure 1: Quantity and type of errors over time

While it is possible for a transfer job to contain multiple tasks that fail or succeed for a variety of reasons, most failed for only one of the possible types of error.

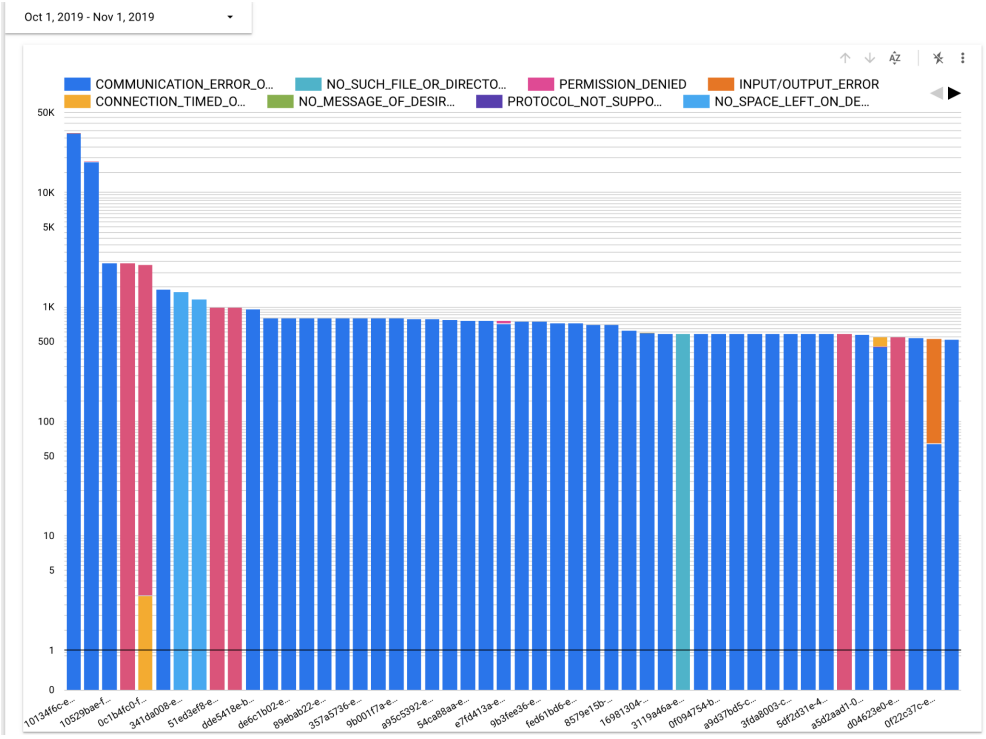


Figure 2: Count of tasks and failure types per job id

Error distribution not only varied over time, but also over the interconnections between nodes. A few nodes were responsible for most failures, and the connection pattern was dynamic.

									dst / Record Count	
src	srm-cms.gri...	gridftp.swt...	dtn.ilifu.ac.za	gridftp.hep...	t2cmcondo...	tbn18.nikhe...	uct2-dc1.uc...	fal-pygrid-3...	griddev03.s...	bohr3226.ti...
bohr3226.tier...	-	5,739	737,095	-	6,911	19,902	3,490	10,940	55,722	136
tbn18.nikhef.nl	-	12,891	-	-	14,466	-	6,133	14,429	893	14,515
eoscmsftp.ce...	38,806	-	-	37,524	-	-	-	-	-	-
dcsrm.usatla...	-	63,813	-	-	44,551	8,058	19,459	14,912	-	4,844
uct2-dc1.uchi...	-	4,764	-	-	3,487	7,157	45	6,938	-	28,582
eosatlassftp....	-	39,750	-	-	65,132	10,828	33,056	11,091	-	1,908
ccsrm.in2p3.fr	32,366	43,079	-	23,902	31,446	2,875	5,364	4,988	-	1,177
golias100.far...	-	5,196	-	-	1,397	18,766	1,973	10,772	61,104	10,434
sdrm.t1.grid.k...	-	14,670	-	-	8,203	16,549	1,018	10,025	874	9,462
storm.ifca.es	13,081	-	-	5,582	-	-	-	-	-	-

Oct 1, 2019 - Nov 1, 2019

Figure 3: Count of errors over connection pairs

										Start_Hour / Record Count	
Top 10 - dat...	Top 10 - data....	201...	Oct 10, 201...	Oct 10, 201...	Oct 10, 201...	Oct 10, 201...	Oct 10, 201...	Oct 10, 201...	Oct 10, 201...	Oct 10, 201...	Oct 1(
bohr3226.tier...	dtn.ilifu.ac.za	4,106	3,450	3,511	4,215	4,636	3,411	3,155	3,782	4,600	
	griddev03.sla...	2	-	-	-	-	-	-	-	-	
	serv02.hep.p...	183	163	143	171	207	155	171	210	195	
	tbn18.nikhef.nl	50	55	51	49	43	211	20	7	25	
	fal-pygrid-30.l...	32	38	34	29	27	25	14	26	62	
	f-dpm000.gri...	27	32	26	28	25	398	3	2	5	
	ftp1.ndgf.org	26	29	26	28	23	395	3	-	-	
	sdrm.t1.grid.k...	25	28	27	28	25	201	3	-	-	
	dcache-atlas-...	26	29	26	26	26	323	3	-	-	
	xrootd.echo.s...	23	29	29	26	21	202	-	-	-	

Figure 4: Variation over time for a given connection pair

Anomaly Detection

Given the observed changes in error distribution across time, connection graph and content (as represented by the error categories), we investigated graph anomaly detection algorithms as a possible way to identify patterns in the logs.
Given the quantity of data, the algorithm needed to be scalable and memory-efficient.
MIDAS[1] (MIcrocluster-based Detector of Anomalies in Streams) seemed a good fit:

- It finds anomalies in dynamic graphs (such as those generated by file transfers, but also intrusions)
- It detects micro-clusters (sudden “burst” of connections between nodes, such as those that may occur with multiple retrials, but also denials of service)
- Memory usage is constant and independent of graph size
- Update time in streaming scenarios is also constant

[1] See https://arxiv.org/pdf/1911.04464.pdf

The algorithm returns a floating-point score, which we can use to highlight the most anomalous connections on a graph in a given time window.

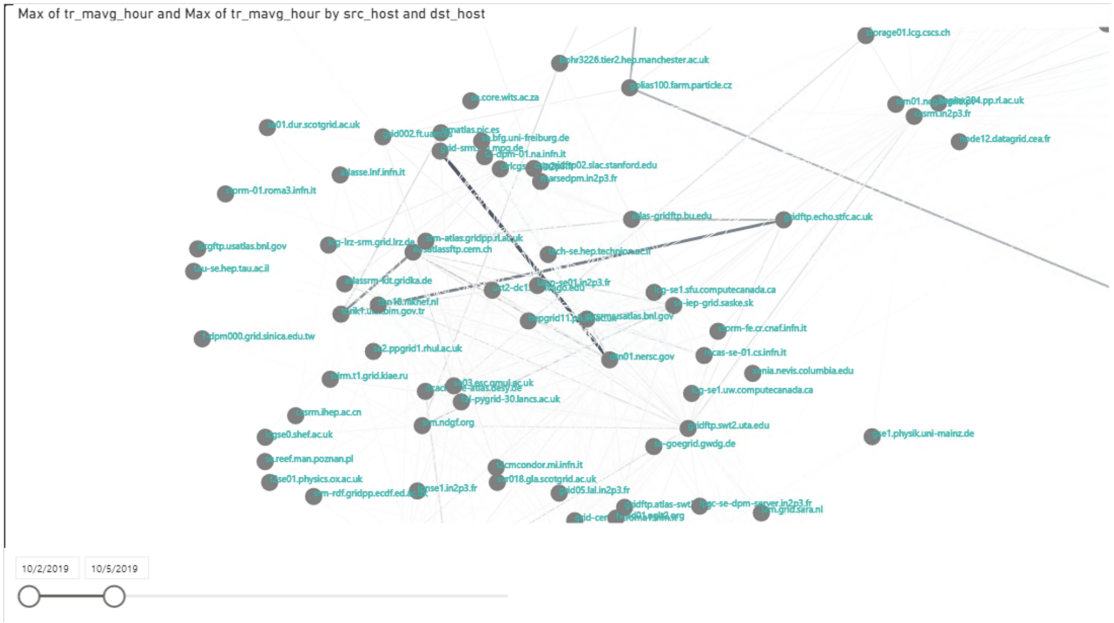


Figure 5: Connection graph between nodes. The darker the link, the higher the anomaly score.

Given the size of the graph, a useful visualization may be a chord diagram, which captures the status at a given time slice. A sequence of such diagrams shows the evolution of the system.

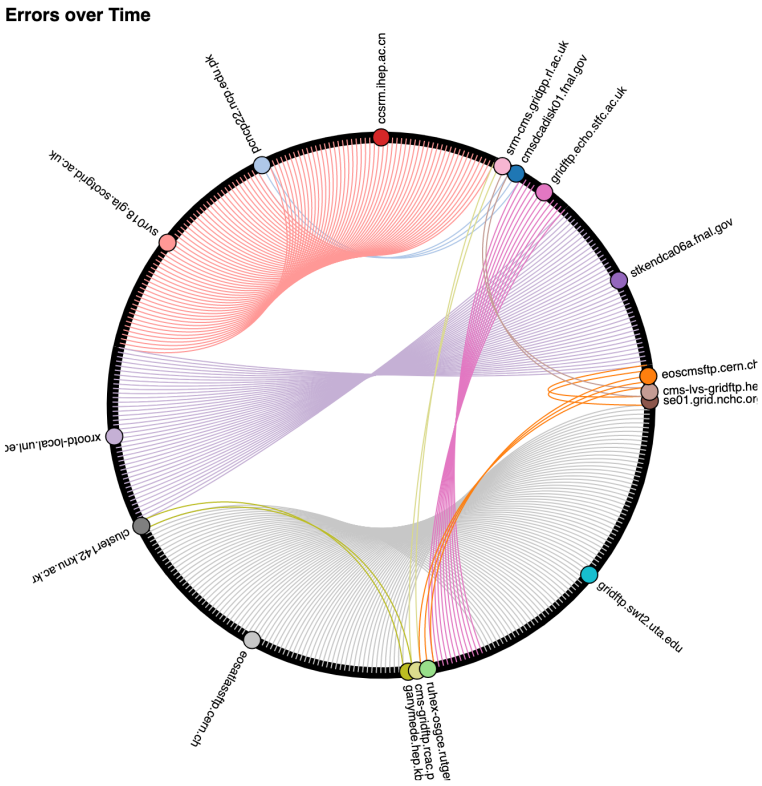


Figure 6: Chord diagram of number of errors between nodes at a given time

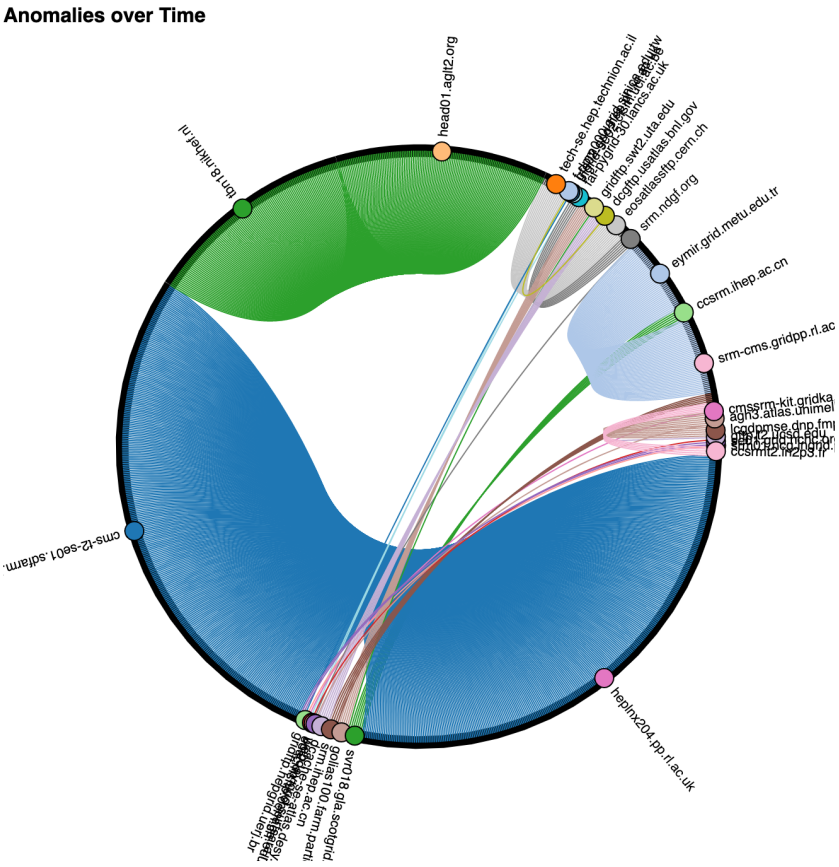


Figure 7: Chord diagram of anomaly scores at a given time

It is important to note that while the quantity of errors contributes to the anomaly score, it is not the only factor. Unusual structures in time (e.g. sudden bursts) and space (e.g. infrequent node pairs) also play a role. Besides, the algorithm is unsupervised. It will find anomalies in the data set. If the data set contains only errors, the anomalies may be in their quantity or frequency. For it to be most useful, it is necessary to provide a complete set of logs, where most events are regular and errors are the anomalies.

Text Analysis

The content of the error message is useful in identifying patterns and clusters of anomalies. To include it as a feature in our detection algorithm, it is necessary to encode it. For that purpose, we opted for Google's *universal sentence encoder*[2].
The model is trained on sentences, phrases and paragraphs rather than just words. The training corpus comes from several sources in order to accommodate a variety of tasks in the field of natural language understanding. While not specifically trained on computer logs, the vocabulary is large enough to include phrases that are likely to appear there.

[2] See https://arxiv.org/abs/1803.11175

The encoder accepts variable-length text as input and returns a 512-element floating-point vector, also known as an embedding.

Topic Modelling

Topic modelling refers to techniques for extracting topics from documents. Documents with similar topics will be clustered together in the space defined by the embedding vectors. The center of the cluster will represent the topic for that group of documents.
Clustering algorithms do not work well in high dimensions; hence, we want to project those embedding vectors into a lower-dimensional space, while preserving as much of the information structure as possible.
UMAP[3] (Uniform Manifold Approximation and Projection) is a good method to do just that. In our experiments, we chose a five-dimensional space for computation and a two-dimensional one for visualization convenience.
HDBSCAN[4] was then used for clustering in five dimensions. The algorithm employs a density-based approach, i.e. it looks for regions in a given space with higher density than their surroundings. It makes few assumptions on the shape and number of such regions, so it is particularly effective with noisy data, such as those produced by natural language processing. On a random sample of 100,000 errors, the analysis identifies 28 topic clusters of 100 elements or more.

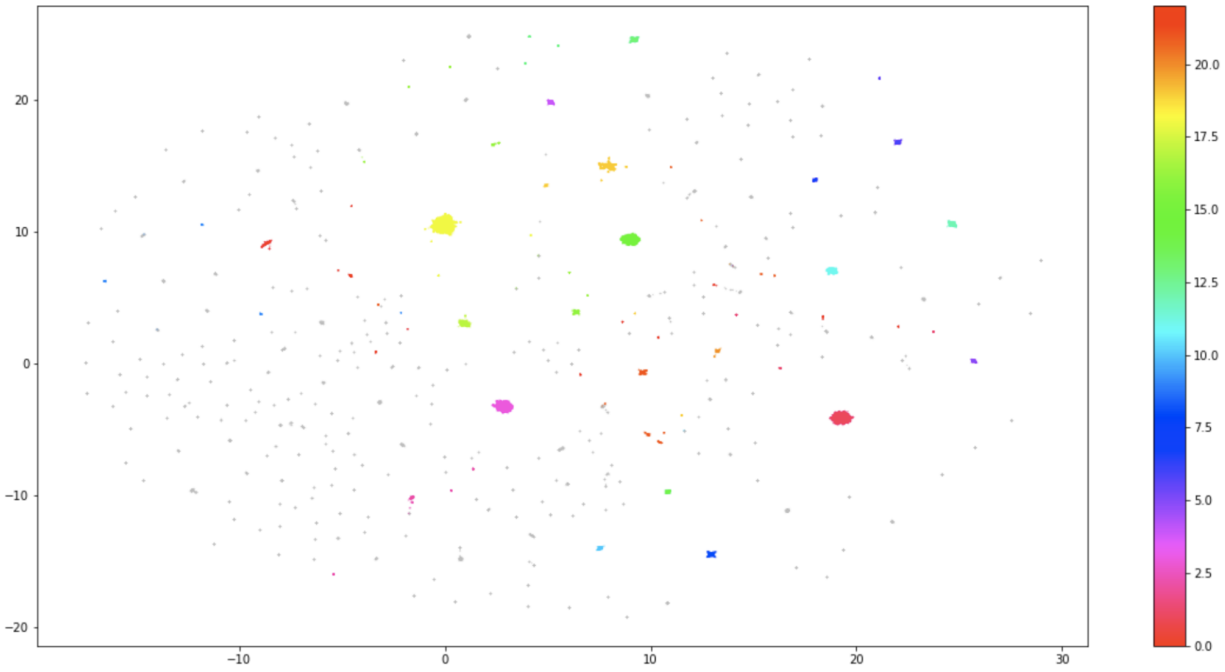


Figure 8: Topic clusters

[3] See https://arxiv.org/abs/1802.03426
[4] See https://link.springer.com/chapter/10.1007%2F978-3-642-37456-2_14

For each cluster, we derived a set of words that is representative of that topic. In order to do so, we scored their relevance by applying a variant of the TF-IDF (term-frequency, inverse document frequency) algorithm.
For instance, for topic 7 (the third error cluster by size) we obtained a set:

	Topic	Size
0	-1	2148
20	19	916
8	7	638
25	24	513
16	15	505
11	10	325
26	25	318
19	18	296
12	11	287
13	12	273

[('directory', 0.17291778055209286),
('500', 0.15570131031862802),
('file', 0.1286784223961737),
('command', 0.1066864092846095),
('open', 0.10410957212360271),
('end', 0.08719723353418019),
('failed', 0.07795190609928998),
('globus_ftp_client', 0.06742707786890055),
('responded', 0.04539232049655455),
('transfer', 0.040571536293723606)]

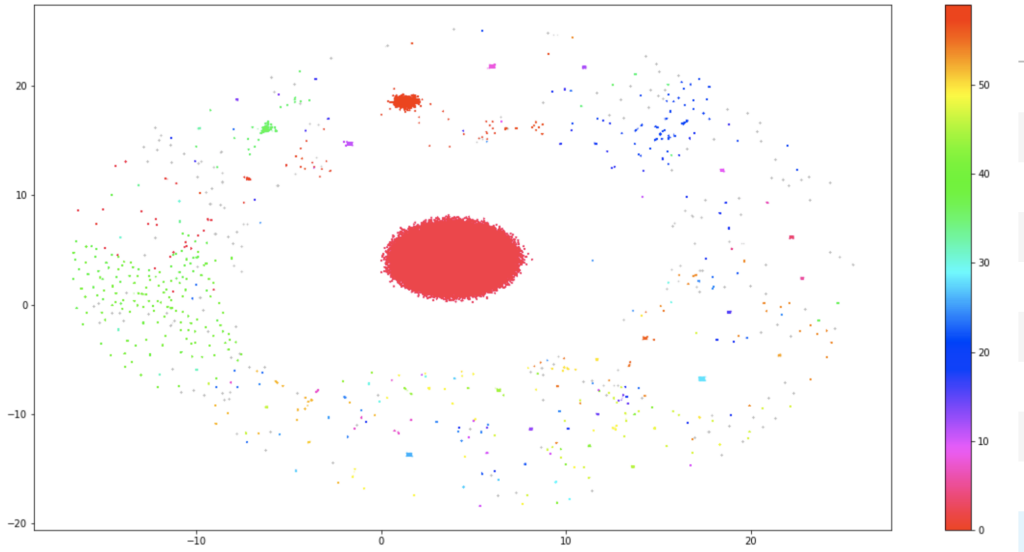
corresponding to messages such as:

'TRANSFER globus_ftp_client: the server responded with an error 500 500-Command failed. : System error in open: No such file or directory 500-A system call failed: No such file or directory 500 End. '

The topic information can be used to analyze and classify errors. It may also help identify probable cause (using the relevance score), although further research is warranted to validate this hypothesis.

Extension to complete logs

The clustering approach can be extended to the full logs. In that case, as expected, the largest cluster (topic 0) will contain successful transfers, as shown in figure 9. Keeping the minimum size at 100, the algorithm identifies 60 clusters. The content of the top ones corresponds to what we found for the error-only example; for instance, the 3rd error cluster (topic 21 in figure 9) still contains messages like “no such file or directory”.



	Topic	Size
1	0	78821
0	-1	2371
57	56	1970
22	21	1272
24	23	1091
28	27	1047
14	13	714
54	53	526
2	1	509
7	6	480

Figure 9: The largest cluster represents successful transfers, errors are outliers

Text features in anomaly detection

Anomalies in traffic over a connection graph are not limited to the dynamic aspect of the graph itself; the information passed over such connections is also relevant. We must consider not only the number, timing and location of links between nodes, but also the messages. Other metadata such as user, file size etc… may play a role too.
To detect anomalies on such multi-aspect data, we used the MSTREAM[5] algorithm, derived from MIDAS, because:

- It works with both categorical and numerical features.
- It consumes a constant amount of memory and a constant amount of time for each record, thus it scales well with the quantity of data we have.
- It captures correlation among multiple features.

MSTREAM returns a floating point anomaly score.

	start_time	src	dst	file_size	activity	user	message	topic	scores
0	2019-10-02 23:01:14.884000+00:00	pcncp22.ncp.edu.pk	cmsdcadisk01.fnal.gov	4315562272		cms	OK	0	9.689366
1	2019-10-02 23:20:11.853000+00:00	dcsrm.usatlas.bnl.gov	srm.ndgf.org	2240929216	Production Input	atlas	OK	0	10.007803
2	2019-10-02 23:33:22.834000+00:00	stormfe1.pi.infn.it	eoscmsftp.cern.ch	2298669205		cms	TRANSFER Operation timed out	15	10.094810
3	2019-10-02 23:34:26.342000+00:00	ccsrm.in2p3.fr	cmsdcadisk01.fnal.gov	3841451136		cms	OK	0	10.174850
4	2019-10-02 23:41:51.789000+00:00	srm.triumf.ca	bohr3226.tier2.hep.manchester.ac.uk	5463304877	Production Input	atlas	OK	0	9.912497

Table 1: Anomaly scores of mostly successful transfers (topic = 0)

[5] See https://arxiv.org/pdf/2009.08451.pdf

	start_time	src	dst	file_size	activity	user	message	topic	scores
45484	2019-10-11 01:08:17.213000+00:00	lcg-lrz-http.grid.lrz.de	atlas-fed-fe1.triumf.ca	4162605415	Production Input	atlas	TRANSFER ERROR: Copy failed with mode 3rd pus...	39	18.135868
75377	2019-10-14 15:39:16.499000+00:00	storage01.lcg.cscs.ch	t2cmcondor.mi.infn.it	594865515	Production Input	atlas	DESTINATION OVERWRITE srm-ifce err: Communicat...	-1	18.136068
35431	2019-10-09 21:14:04.095000+00:00	eoscmsftp.cern.ch	ruhex-osgce.rutgers.edu	411948	ASO	cms	TRANSFER globus_ftp_client: the server respon...	34	18.136483
65824	2019-10-13 07:53:30.428000+00:00	eosatlassftp.cern.ch	dcsrm.usatlas.bnl.gov	35145277	Production Output	atlas	TRANSFER globus_ftp_client: the server respon...	48	18.136489
53304	2019-10-11 17:54:33.528000+00:00	cluster142.knu.ac.kr	srm-cms.gridpp.rl.ac.uk	19898252	rucio	cms	DESTINATION SRM_PUTDONE Error on the surl srm:...	56	18.136585

Table 2: Anomaly scores of failed transfers (topic ≠ 0)

As previously observed, the anomaly score for failed transfers tends to be greater than for successful ones, but that is not always the case. Other factors, such as unusually large or infrequent transfers, also contribute to it. In our experiment, out of the 1,000 most anomalous events, about 85% were actual errors, while about 15% were unusual events (so precision at 1000 is 0.85). One can filter the latter out using the topic value.
The score can also be used as a way to help the support teams prioritize events. However, this suggestion warrants further research and validation.

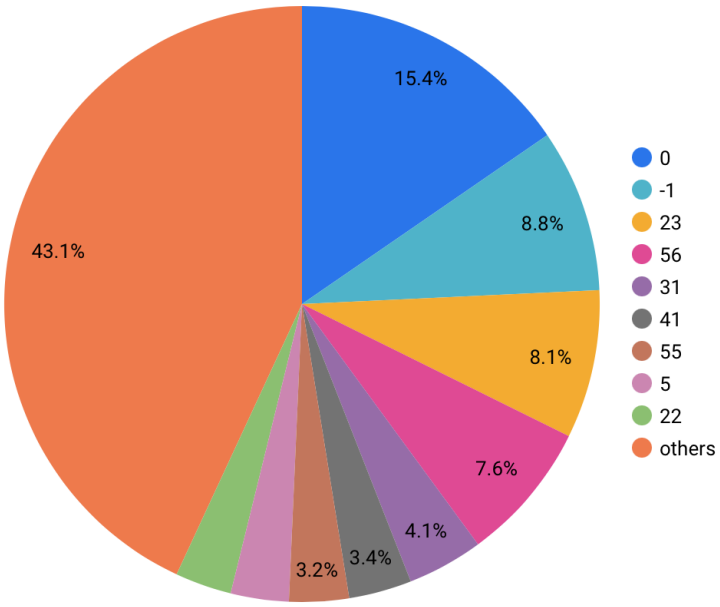


Figure 10: Topic distribution over the 1000 most anomalous entries

Recommendations

The MSTREAM anomaly detection approach in conjunction with topic modelling has shown promising results in identifying and scoring error patterns. We suggest further investigation with:

- Datasets covering a longer period of time, thus capturing more of the “usual” workload. This will allow us to refine the anomaly detector.
- More features out of the existing data, once their relevance is verified.
- A prototype streaming detector, running as a container, that generates online scores. It can be used to validate the approach with the support team.

Finally, if the end goal is to help the support team prioritize their work, one may consider introducing a measure of severity. It can either be explicit (e.g. set by the user or support team) or implicit (derived from other criteria yet to be determined). Given that, one can then explore recommender-based techniques to compute severity, in addition to anomaly.

Implementation

For experimental purposes, we recommend:

- Google BigQuery to store the input data and the output scores, perform data transformation and aggregation.
- AI Platform Notebooks for development and training of the models.
- Datastudio for reporting and visualization, accelerated by BigQuery BI engine.
- Google Kubernetes Engine or AI Platform prediction service to deploy the trained models as containers in production.

References

S. Bhatia et al., *MIDAS: Microcluster-Based Detector of Anomalies in Edge Streams,* AAAI 2020
S. Bhatia et al., *MSTREAM: Fast Anomaly Detection in Multi-Aspect Streams,* WWW 2021
L. McInnes et al., *Umap: Uniform Manifold Approximation and Projection for Dimension Reduction,* arXiv preprint arXiv:1802.03426, 2018
R. Campello et al., *Density-Based Clustering Based on Hierarchical Density Estimates,* PAKDD 2013
D. Cer et al., *Universal Sentence Encoder,* arXiv preprint arXiv:1803.11175, 2018