Automatic Detection of Network Traffic Anomalies and Changes

Astha Syal[1], Alina Lazar[1], Jinoh Kim[2], Alex Sim[3], Kesheng Wu[3]

[1]Department of Computer Science and Information Systems, Youngstown State University

[2]Department of Computer Science and Information Systems, Texas A&M University-Commerce

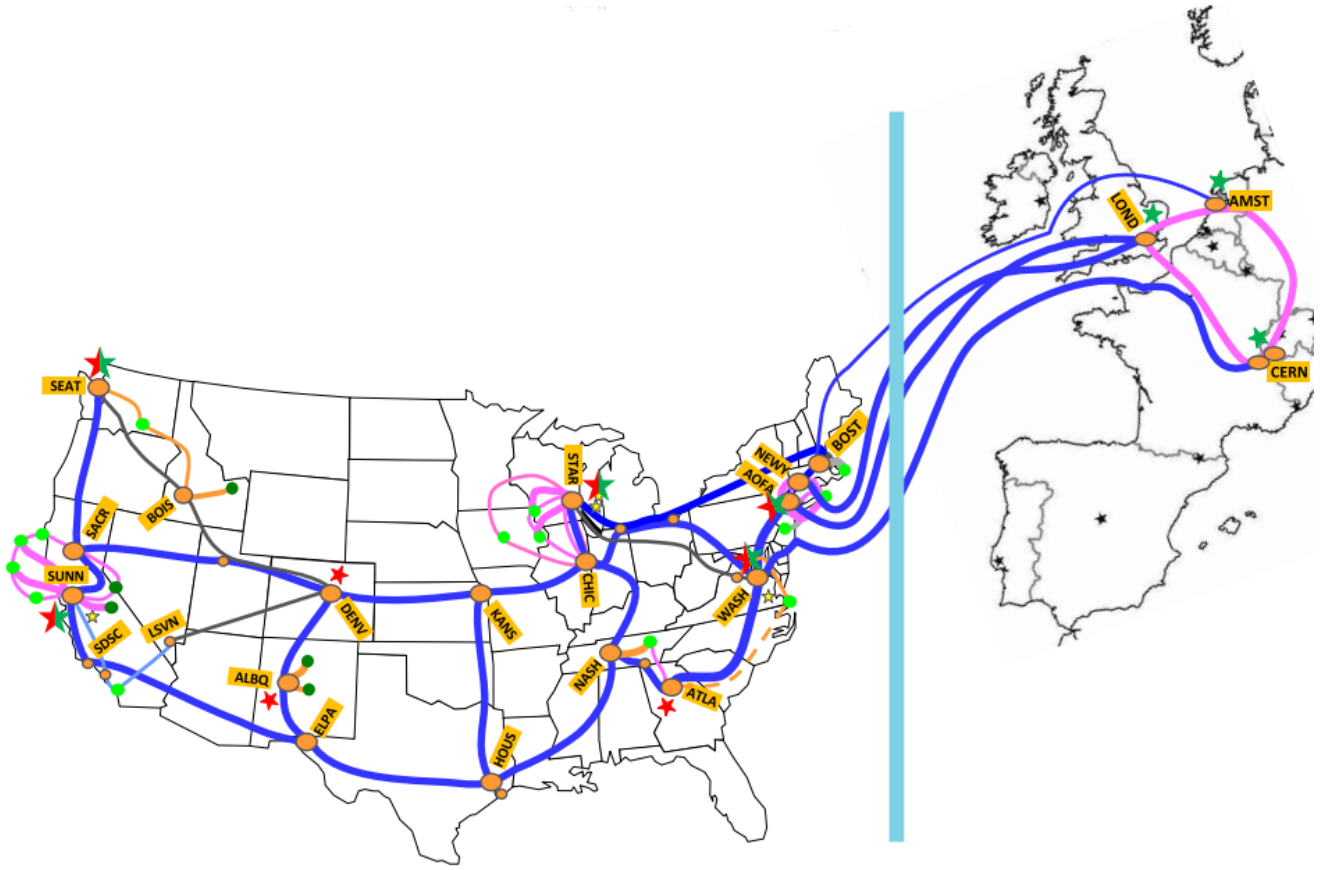
[3]Scientific Data Management Research Group, Lawrence Berkeley National Laboratory

Introduction and Motivation

- Computer networks grow in size and complexity
- Network traffic monitoring and analysis becomes important
- TCP (Transmission Control Protocol)
- UDP(User Datagram Protocol)
- Possible problems: packet losses, duplication and reordering

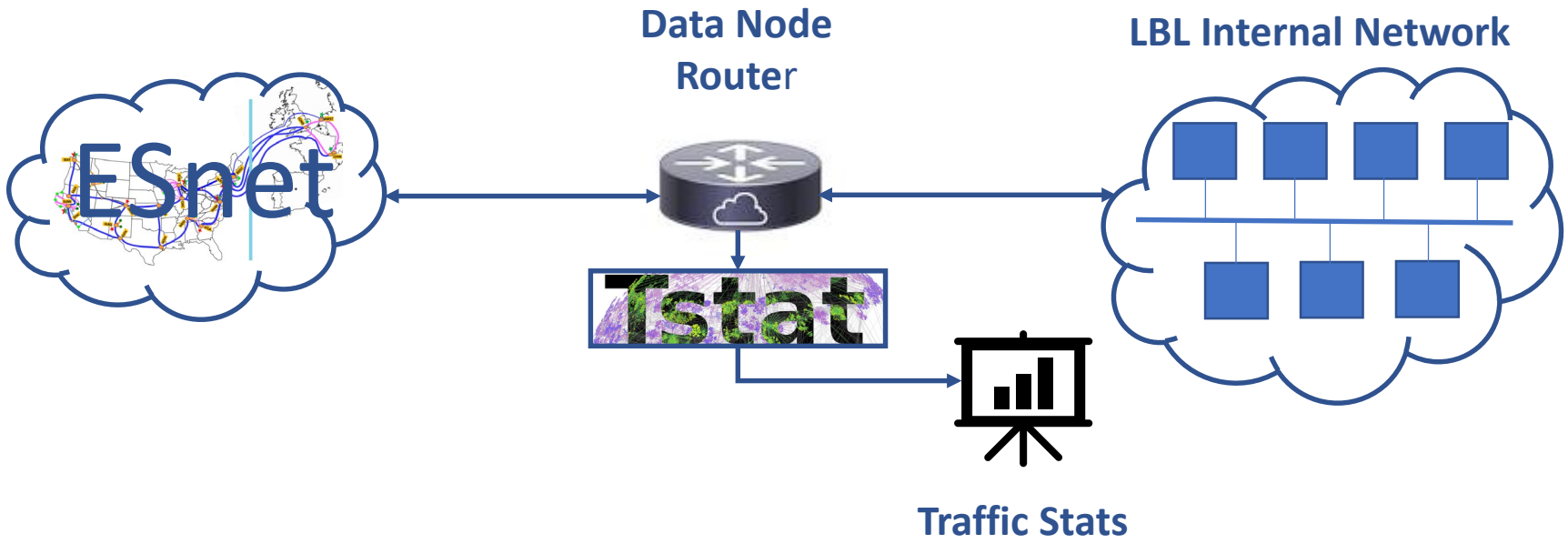
ESnet

- The Energy Sciences Network is a high-performance, unclassified network built to support scientific research
- Funded by the U.S. Department of Energy's Office of Science (SC) and managed by Lawrence Berkeley National Laboratory



Network Traffic Congestion

Tstat Data Collection at LBNL



- Network Data recorded during File Transfers
  - TCP statistics (Tstat files)
- Research Question: **Using large datasets of Tstat data can we identify time windows with consistent low throughput?**

Current Drawbacks

Problems:

- Not real-time, only checked when something is wrong
- Large datasets to analyze

Ideal case: Automate detection of low throughput and raise an alert

- Network traffic data is high volume, heavy stream of high dimensionality data
- No training (labelled) datasets available
- Data does not follow the same distribution
- Machine learning models are black boxes

Main Contributions

Statistical Analysis:

- To extract throughput threshold values to label the time intervals as 'low' versus 'normal'.

UMAP:

- To provide a 2-dimensional representation of the datasets to understand the structure of the data.

Supervised Machine Learning:

- Classification experiments performed on real network data collected from eight DTNs using **Tstat.**

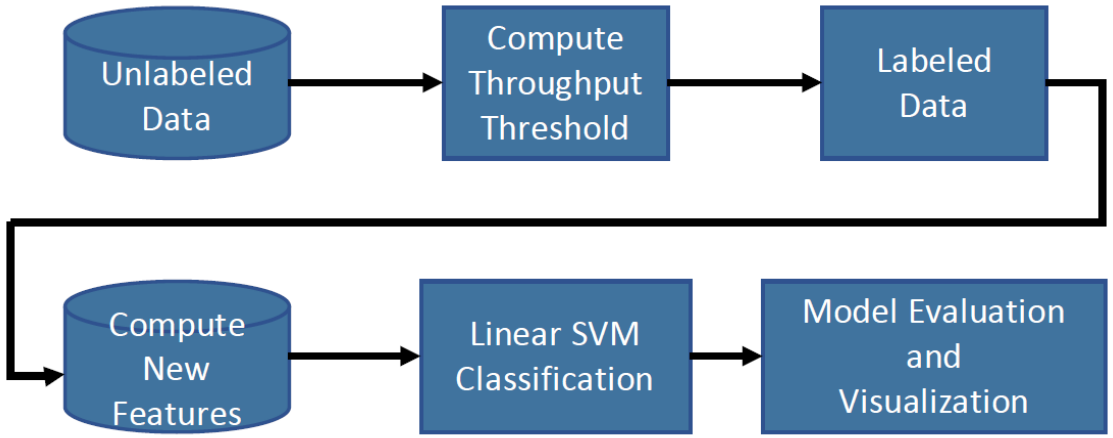
Evaluation:

- Checking the results of this supervised learning approach, especially the false positives and false negatives, using throughput plots.

Machine Learning Approach

Research Question:

Using Tstat data features can we classify upcoming time intervals as low throughput with good accuracy, precision and recall?



UMAP – Dimensionality Reduction

UMAP is a dimension reduction technique that works well for 2D visualizations, and for general non-linear dimension reduction.

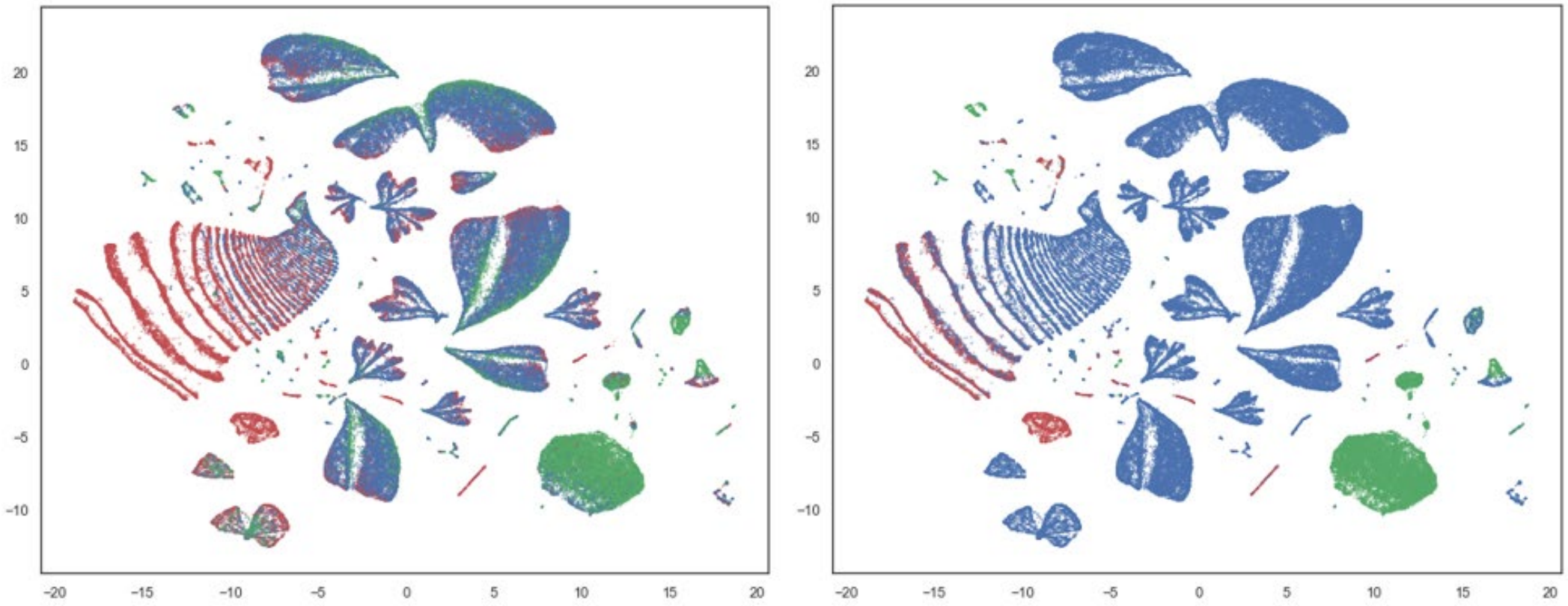
Based on manifold theory and fuzzy topological data analysis.

The algorithm builds a weighted k-neighbor graph to efficiently approximate the k-nearest-neighbor computation and calculates spectral embeddings.

The embeddings are optimized using Stochastic Gradient Descent algorithm.

The algorithm assumes the data is uniformly distributed on a Riemannian manifold, assumption that does not always hold for real data.

UMAP - Representation

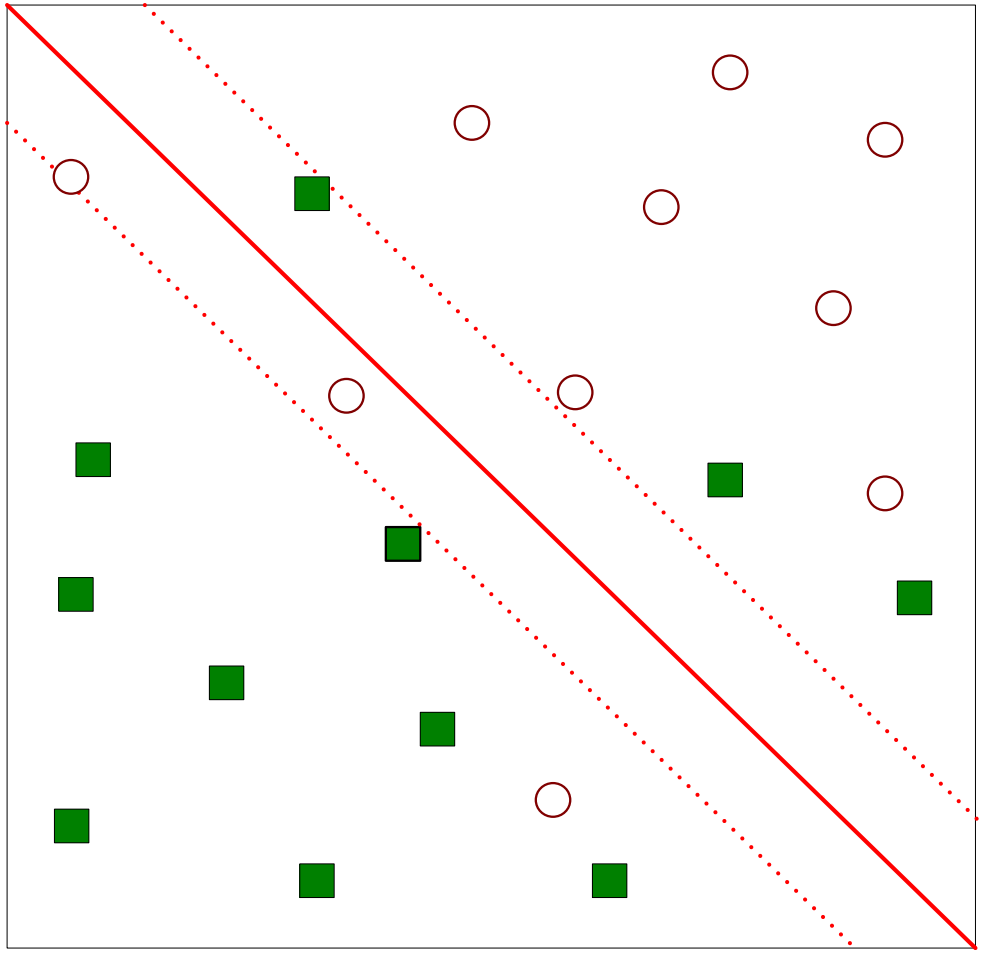


Linear SVM

An SVM training algorithm builds a linear hyperplane with the property of having the largest margin.

Data points, are mapped so the two categories are divided by the optimal hyperplane that has the largest gap.

Linear SVMs can be solved efficiently using the coordinate descent algorithm for the optimization step.



Tstat (TCP Statistic and Analysis Tool)

- Network traffic statistics at the flow level can be collected using passive monitoring tools such as **Tstat**.
- For TCP connections, congestion window size, out-of-sequence segments, duplicated segments, number of bytes and segments retransmitted, and RTT values are just some of the statistics collected.
- **Tstat** only saves completed flows.
- **Tstat** also records UDP messages. UDP communication contributes a very low percent of the total bytes moved from/to the major computer center.

Tstat (TCP STatistic and Analysis Tool) Datasets

Node	Number of Flows	60 minutes	30 minutes	5 minutes
1	2,447,602	4,232	8,450	47,310
2	1,119,470	2,639	4,372	13,618
3	5,131,592	4,029	7,845	36,089
4	455,244	3,286	5,716	21,006
5	135,531	421	659	2,140
6	166,116	598	1,060	4,359
7	157,247	412	659	2,439
8	169,233	626	1,096	4,539

Summary Statistics

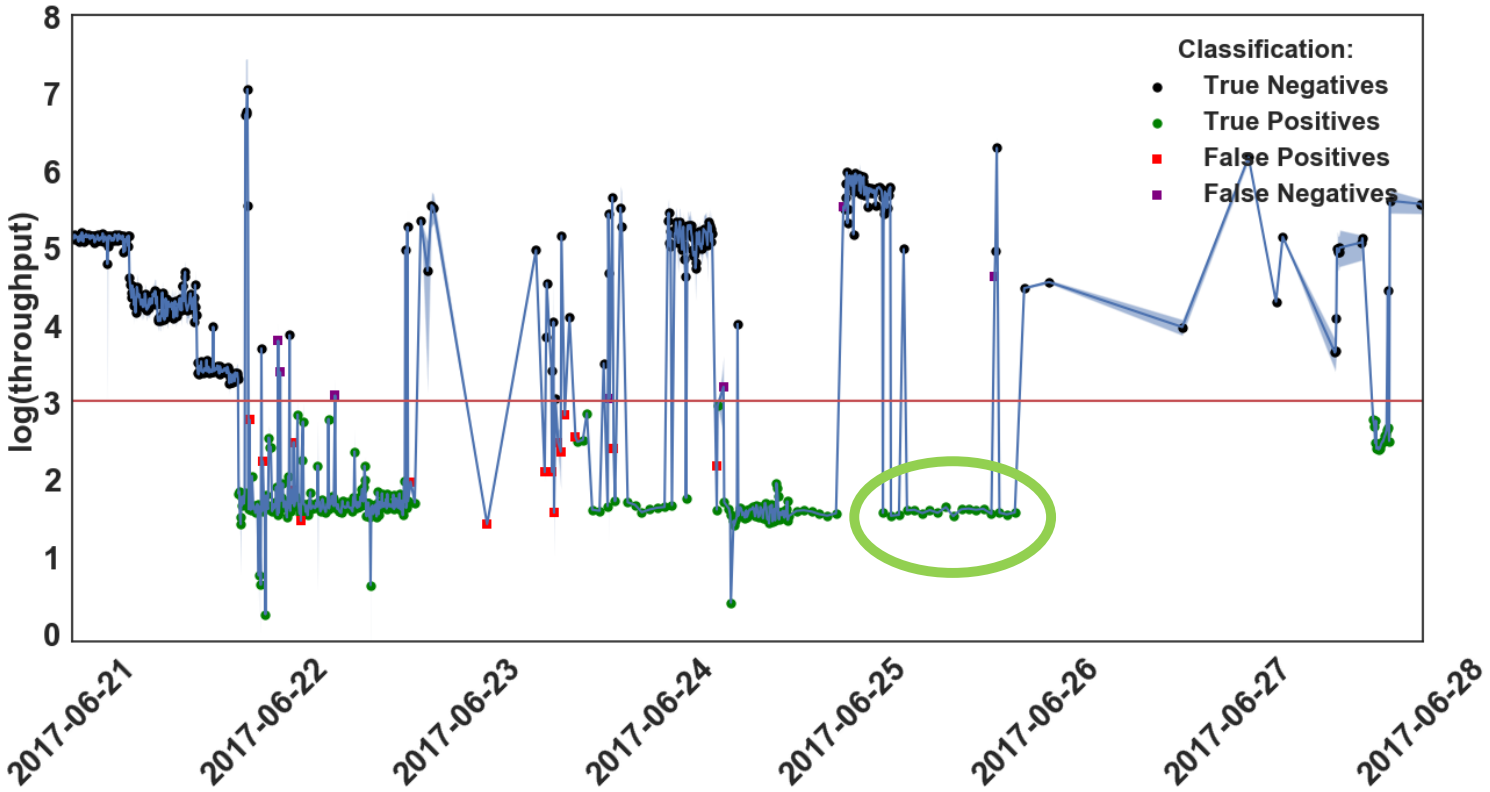
Node	count	mean	std	min	5%	25%	50%	75%	max
1	2,447,602	129.46	317.78	0.00003	13.941	31.636	57.901	111.15	9853.994
2	1,119,470	98.836	115.29	0.00009	20.175	56.643	82.353	103.64	9883.041
3	5,131,592	881.61	978.5	0.00005	43.448	174.85	377.81	1521.1	9889.773
4	455,244	174.46	375.75	0.00007	7.66	42.672	87.981	162.7	9725.279
5	135,531	105.23	69.401	0.0008	22.825	57.376	92.339	135.83	3003.8
6	166,116	262.35	335.53	0.0007	22.329	64.642	107.12	198.24	5920
7	157,247	91.439	81.206	0.005779	21.242	47.877	77.219	117.02	3048
8	169,233	268.05	631.1	0.008	26.714	60.551	93.842	149.5	3271.2

Experiments and Results

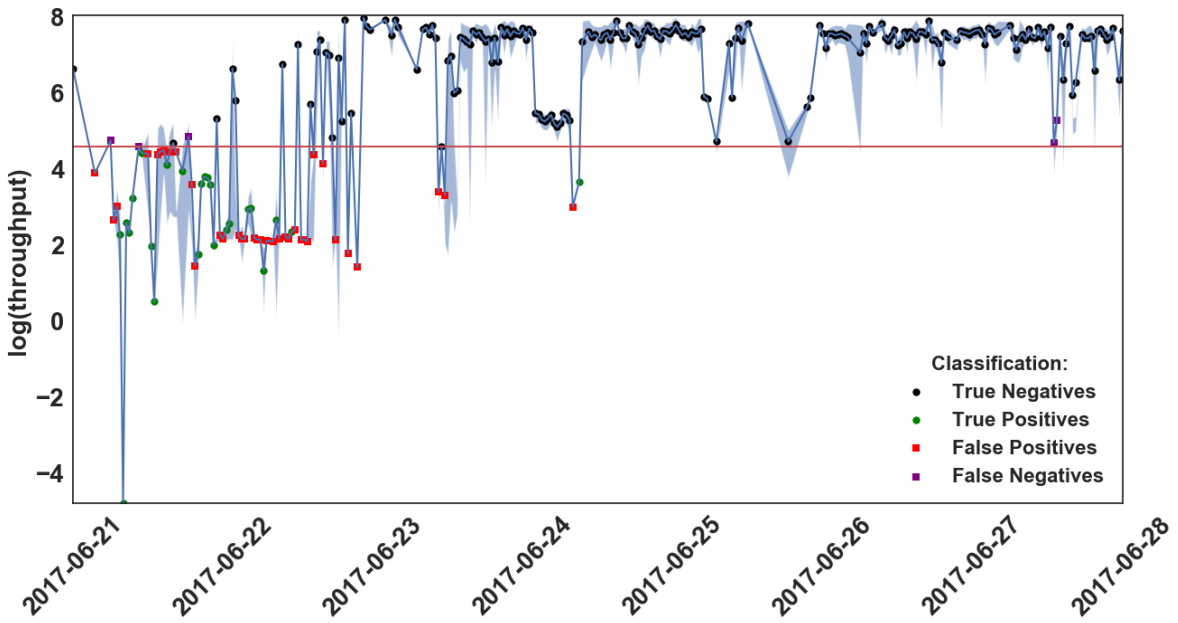
- All features are normalized using the MinMax Scaling procedure.
- The datasets are divided into time intervals based on three time frequencies: 5, 30 and 60 minutes.
- Averages for all the features including throughput are calculated and saved for further input into the classification algorithm.
- First, we used the entire datasets for all the eight nodes and divided each of them based on the time stamp in training and testing sets. The last week or last month of data is used for testing and the rest to build the model.
- Second, the second set of experiments are done for node 1 through 4, where the datasets contain a larger number of network flows collected over 6 months. For these datasets we run cross validation using a Time Series Split with k=6 sets.

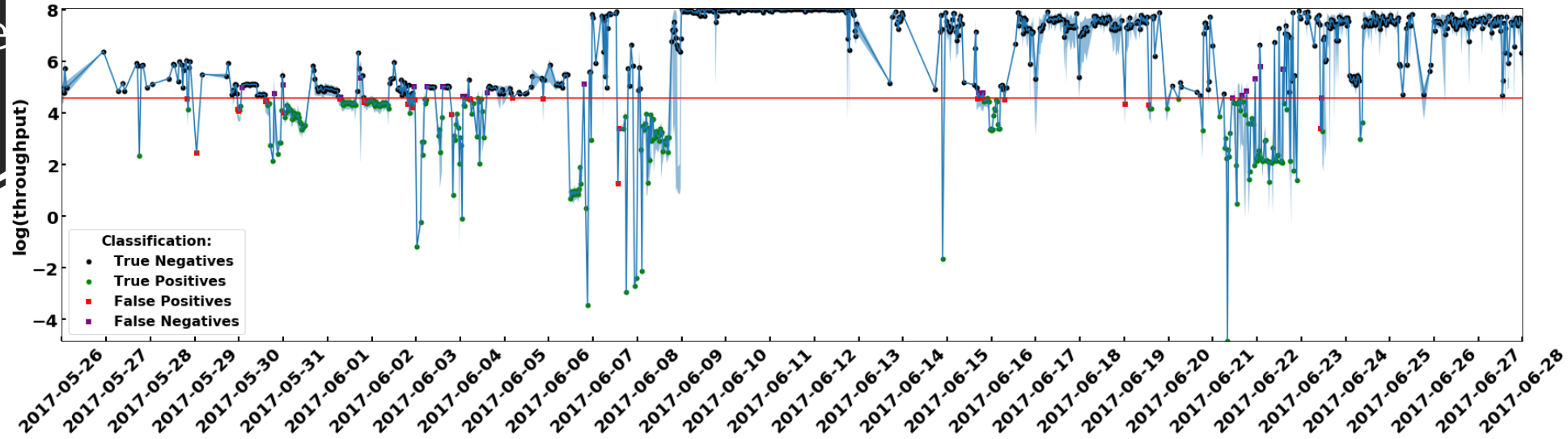
	Minutes	Accuracy	Precision	Recall	F1
node 1	5Min	0.9013	0.8632	0.76	0.808
	30Min	0.8574	0.6878	0.584	0.632
	1H	0.8664	0.6875	0.588	0.634
node 2	5Min	0.9312	0.9401	0.785	0.856
	30Min	0.9055	0.9189	0.713	0.803
	1H	0.8985	0.9008	0.699	0.787
node 3	5Min	0.9181	0.8765	0.809	0.841
	30Min	0.9288	0.8261	0.813	0.82
	1H	0.9024	0.848	0.697	0.765
node 4	5Min	0.9396	0.9058	0.647	0.755
	30Min	0.9121	0.8917	0.535	0.669
	1H	0.9221	0.9107	0.531	0.671
node 5	5Min	0.9413	0.9758	0.921	0.948
	30Min	0.8349	0.9891	0.722	0.835
	1H	0.8862	0.832	0.925	0.899
node 6	5Min	0.9311	0.8274	0.76	0.792
	30Min	0.9279	0.8281	0.828	0.828
	1H	0.9176	0.8621	0.714	0.781
node 7	5Min	0.9703	0.9782	0.952	0.965
	30Min	0.9167	0.9545	0.866	0.908
	1H	0.878	0.9216	0.81	0.862
node 8	5Min	0.8824	0.8878	0.431	0.58
	30Min	0.8562	0.7742	0.4	0.528
	1H	0.8916	0.8	0.533	0.64

Average throughput for node 7, 5min time windows



Average throughput for node 8, 30min time windows

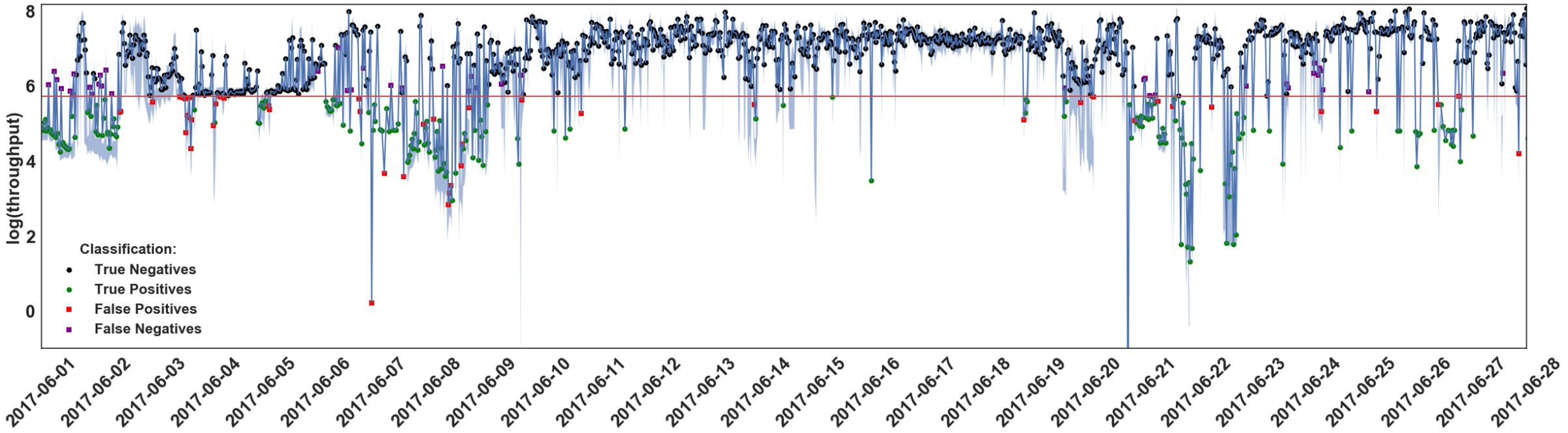




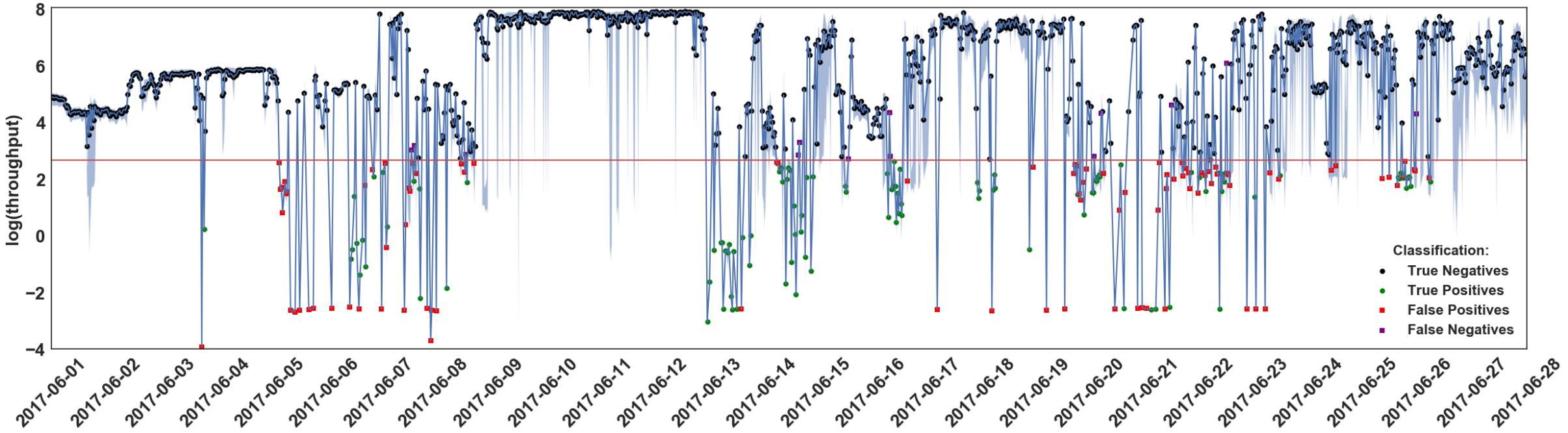
Cross-validation Results

	Window Size	Accuracy	Precision	Recall	F1 Score
node 1	5Min	0.8920 ± 0.0076	0.8587 ± 0.0109	0.8132 ± 0.0373	0.8272 ± 0.0301
	30Min	0.8884 ± 0.0127	0.8381 ± 0.0339	0.7924 ± 0.0519	0.8045 ± 0.0501
	1H	0.8753 ± 0.0118	0.8181 ± 0.0427	0.7638 ± 0.0504	0.7794 ± 0.0502
node 2	5Min	0.9157 ± 0.0105	0.8666 ± 0.0346	0.8237 ± 0.0363	0.8369 ± 0.0332
	30Min	0.8758 ± 0.0195	0.8175 ± 0.0359	0.8099 ± 0.0270	0.8068 ± 0.0327
	1H	0.8643 ± 0.0132	0.7972 ± 0.0246	0.8052 ± 0.0234	0.7987 ± 0.0229
node 3	5Min	0.9128 ± 0.0144	0.8702 ± 0.0260	0.8244 ± 0.0413	0.8375 ± 0.0324
	30Min	0.9176 ± 0.0150	0.8858 ± 0.0146	0.8702 ± 0.0225	0.8727 ± 0.0148
	1H	0.8980 ± 0.0139	0.8638 ± 0.0099	0.8349 ± 0.0444	0.8341 ± 0.0348
node 4	5Min	0.8785 ± 0.0259	0.8474 ± 0.0270	0.7887 ± 0.0244	0.8106 ± 0.0216
	30Min	0.8686 ± 0.0245	0.8354 + 0.0235	0.7716 ± 0.0244	0.7889 ± 0.0231
	1H	0.8614 ± 0.0257	0.8221 ± 0.0282	0.7586 ± 0.0286	0.7719 ± 0.0274

Average throughput node 3, 30min time windows



Average throughput node 4, 30min time windows



Conclusions

Reliable network transfers are essential for successful operations at large scientific facilities where petabytes are transferred daily.

To identify possible problems such as low throughput, we propose to classify the traffic flows captured by **Tstat** with a linear SVM classification algorithm.

Our system splits the **Tstat** log streams into chunks so as to make predictions in near real-time.

Tests show that this new method is able to accurately detect abnormally low throughput time intervals.

Conclusions

This paper presents a supervised data analytics system that effectively mines network traffic data. The proposed methodology is based on a two-phase approach:

- assigns binary classification labels to network transfers using an adaptive threshold based on the throughput mean.
- builds a classification model to predict new data labels in real-time to identify traffic with low throughput.

Conclusions

The tests on the proposed method showed its ability to accurately identify large windows of low throughput, but also showed problems in case of isolated or alternating intervals.

In future, we plan to extend the current system:

- To include of more time related features.
- To use feature selection techniques to eliminate highly correlated features.
- To apply the same approach to other datasets to investigate the generalization capabilities of the presented method.

Acknowledgements

This work was supported by:

- *The Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.*
- *The Office of Workforce Development for Teachers and Scientists (WDTS) under the Visiting Faculty Program (VFP), Office of Science, the U.S. Department of Energy.*
- *The Office of Research at Youngstown State University.*
- *This research used resources of the National Energy Research Scientific Computing Center.*
- *Student travel grant for HPDC'19, sponsored by NSF.*