Training Classifiers to Identify TCP Signatures in Scientific Workflows

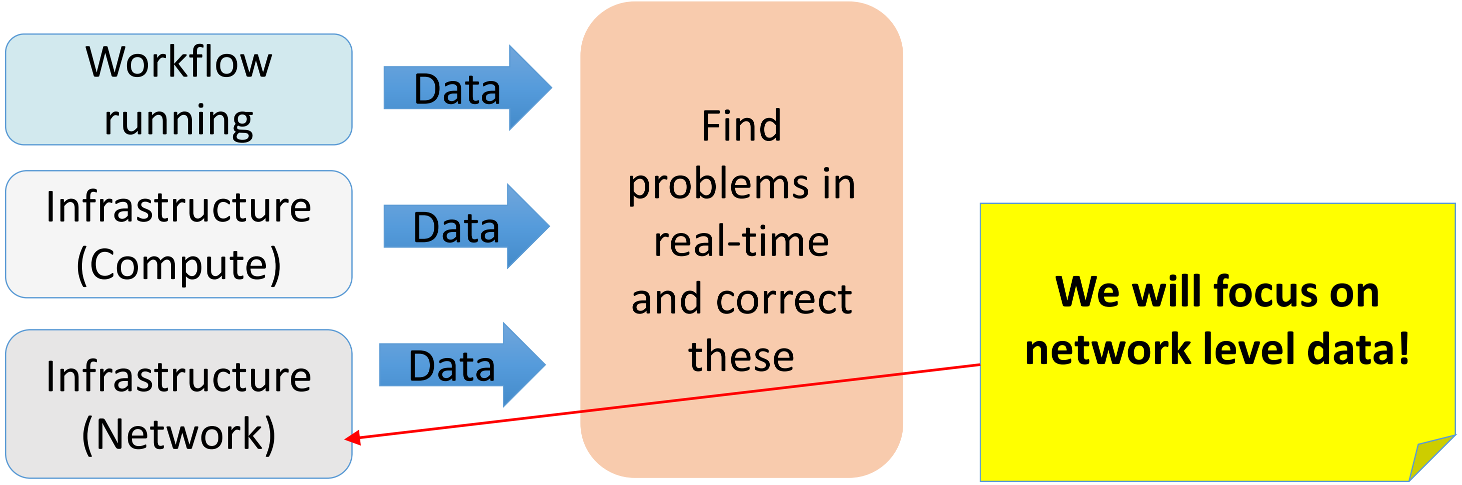
George Papadimitriou, Mariam Kiran, Cong Wang, Anirban Mandal, Ewa Deelman

International Workshop on Innovating the Network for Data Intensive Science – INDIS 2019
November 17, 2019

Funded by the US Department of Energy under Grant #DE-SC0012636M

Machine learning (ML) for Performance Data

Panorama 360 (Performance Data Capture and Analysis for End-to-end Scientific Workflows)



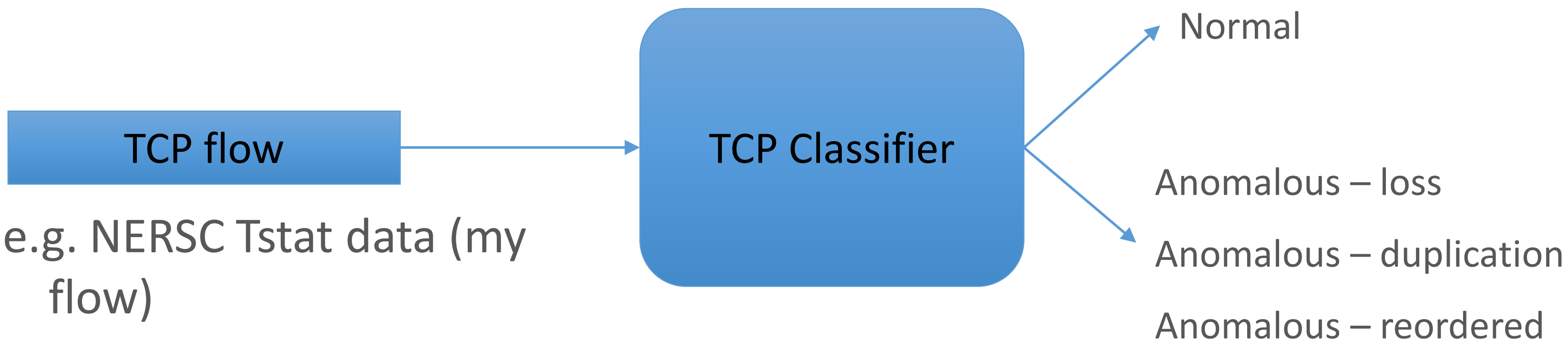
- TCP used in science workflows
- Tstat tool (http://tstat.polito.it/)
  - Approx. 150 variables
    - Ip addresses, port nums, Average RTT, bytes sent, ACK sent/rec, completion time, when first ACK received, etc
    - Throughput = Bytes transmitted/Completion time

Our Objective: Recognize unique TCP behaviors when anomalies exist (loss, duplication and reordering)

- Multiple TCP congestion algorithms are being used
- Approaches explore anomaly detection using rule systems [1], or predicting throughput [2]
- Current approaches do not:
  - Differentiate elephant and mice flow behaviors
  - Focus on simple rule based approaches to classify
- Our approach:
  - Use supervised classification methods to identify behaviors as normal and abnormal across TCP CUBIC, RENO, HAMILTION, BBR
  - Elephant and mice flows
  - Scientific workflows

[1] M. Mellia, M. Meo, L. Muscariello, and D. Rossi, "Passive analysis of tcp anomalies," Comput. Netw. , vol. 52, pp. 2663–2676, Oct. 2008.
[2] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A machine learning approach to tcp throughput prediction," IEEE/ACM Trans. Netw., vol. 18, pp. 1026–1039, Aug. 2010.

The Critiques: Why would we need this?



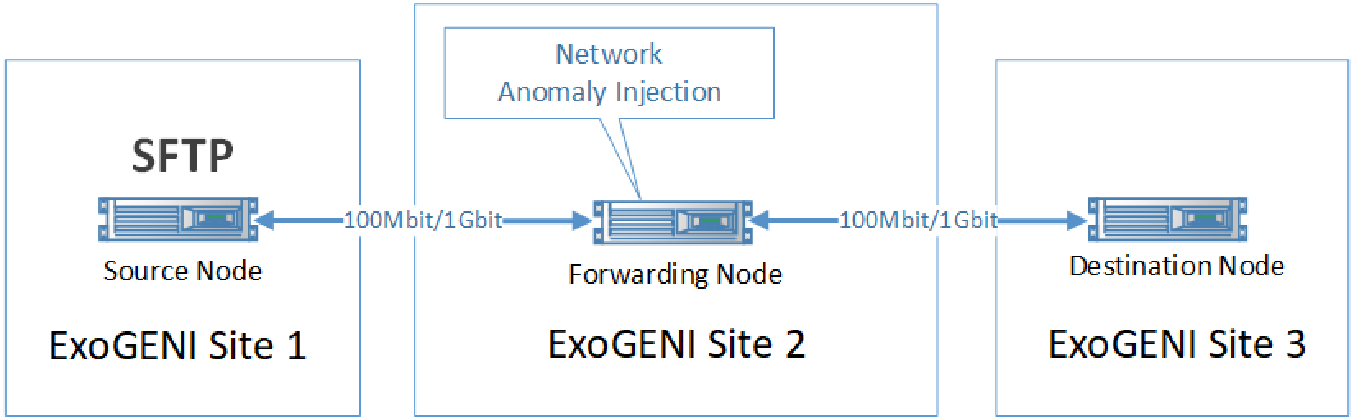
e.g. NERSC Tstat data (my flow)

- Basic Tstat could help see if loss is happening (e.g. retransmits are high)
  - What about the other anomalies?
- Different TCP congestion algorithms behave differently
- Reduce work for us to check which TCP is being configured
- We build extensive “labeled” data sets (next slide)

Labeled data sets and Experiment setup

- Sftp to transfer, Linux traffic Control for adding anomalies, tstat at source
- TCP flows under “normal” conditions (>1000 flows)
- TCP flows when “loss” is added: Synthetic anomalies (>1000 flows)
  - Same for duplication and reordering

- Flow distribution:
  - Elephant: 1-1.2GB link bandwidth 100 Mbps
  - Mice: 80MB and 120MB link bandwidth 1Gbps



Training data collected

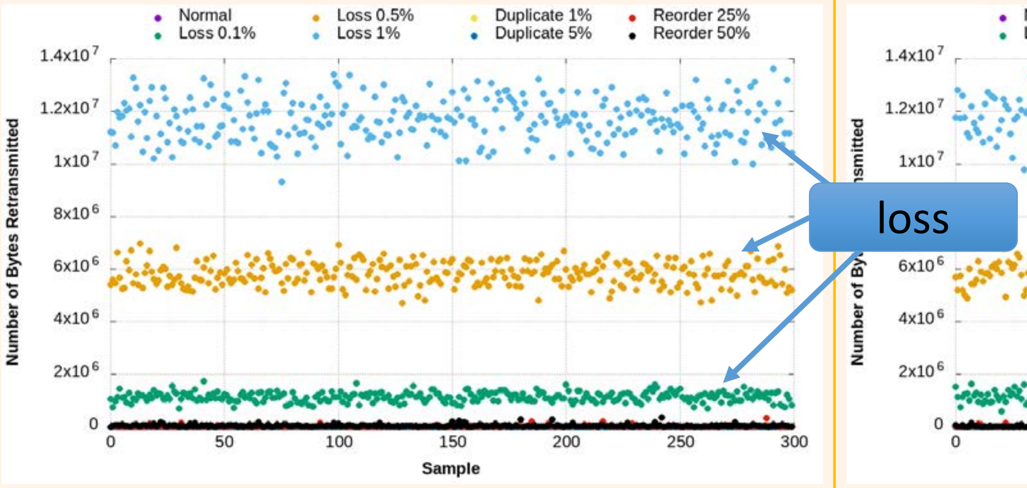
Number of flows (Mice-M, Elephant-E) using CUBIC, RENO, HAMILTON, BBR

Type	TCP Congestion Algorithm							
	Cubic		Reno		Hamilton		BBR	
	M	E	M	E	M	E	M	E
Normal	1000	300	1000	300	1000	300	1000	300
Loss 0.1%	1000	300	1000	300	1000	300	1000	300
Loss 0.5%	1000	300	1000	300	1000	300	1000	300
Loss 1%	1000	300	999	300	998	300	1000	300
Dupl. 1%	1000	300	1000	300	1000	300	1000	300
Dupl. 5%	1000	300	1000	300	1000	300	1000	300
Reord. 25%	1000	300	1000	300	1000	300	1000	300
Reord. 50%	1000	300	1000	298	1000	297	1000	299

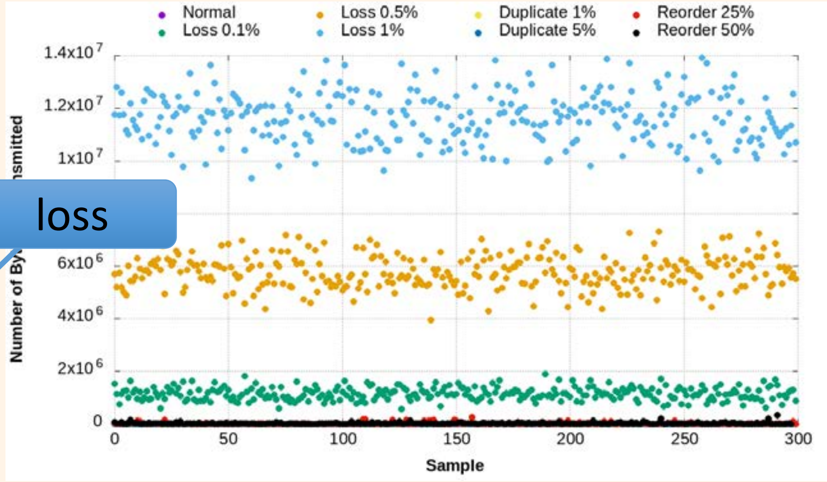
TABLE II: Number of Mice and Elephant flows generated to train the classifiers under normal and anomalous conditions. M: Mice, E: Elephant.

Initial Analysis: Retransmits - Elephant

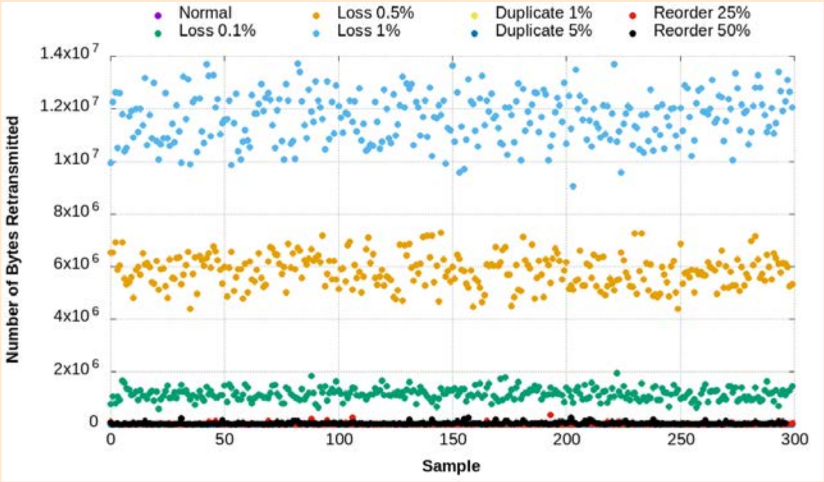
Cubic



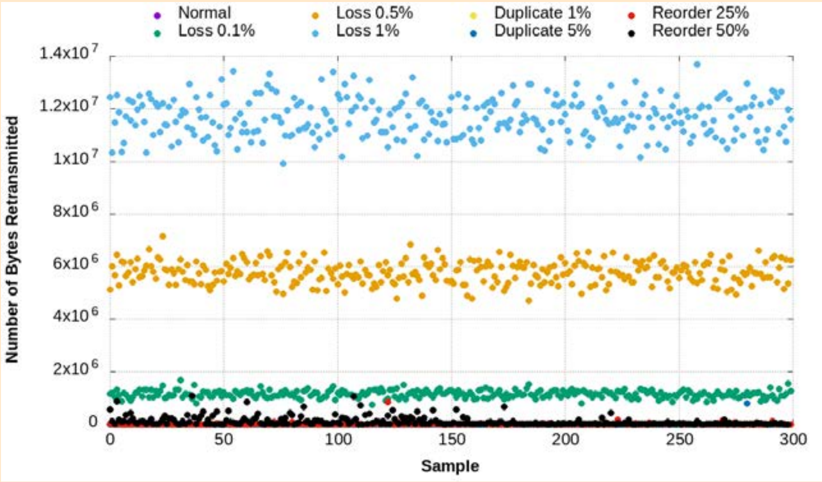
Reno



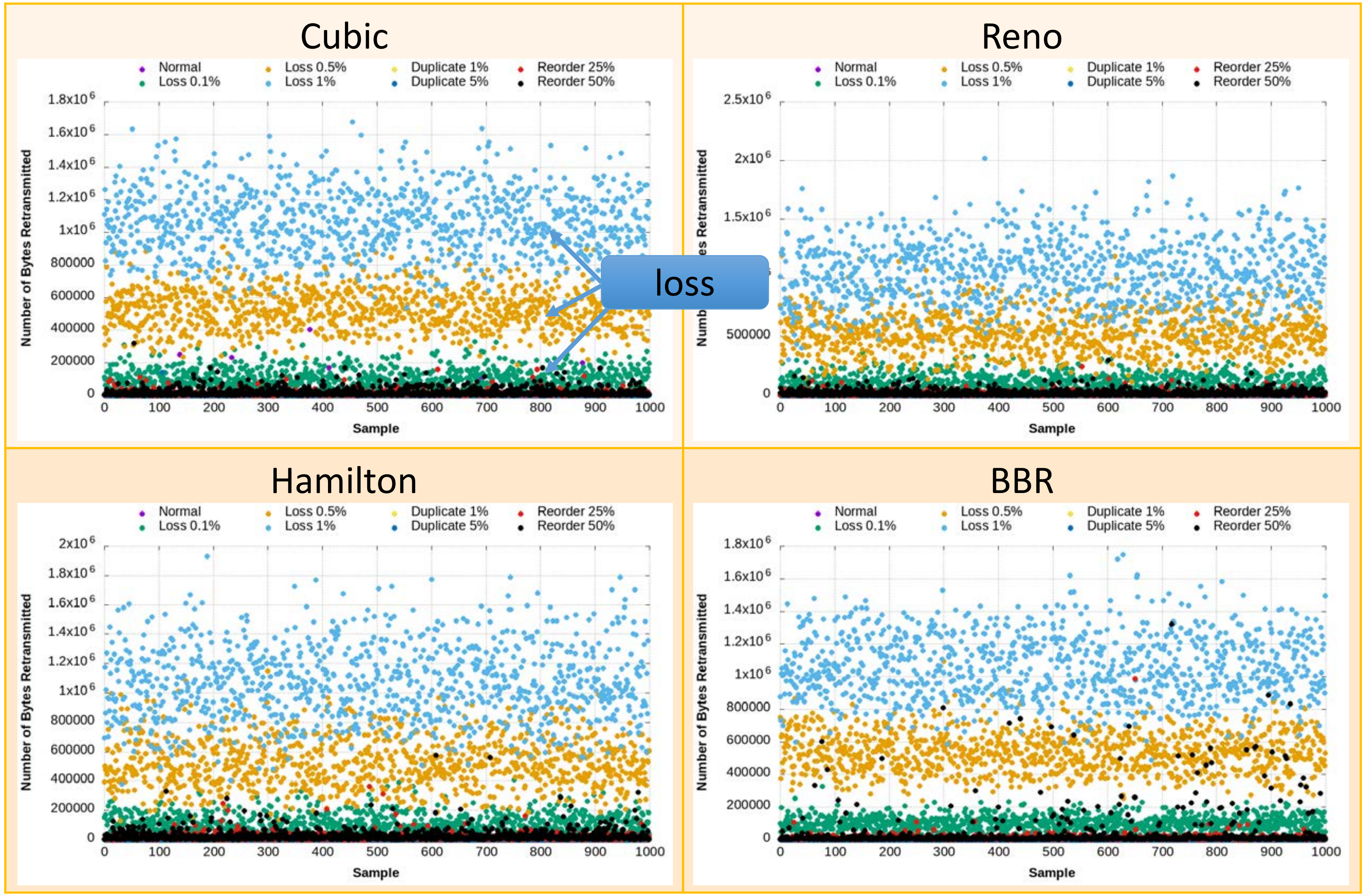
Hamilton



BBR

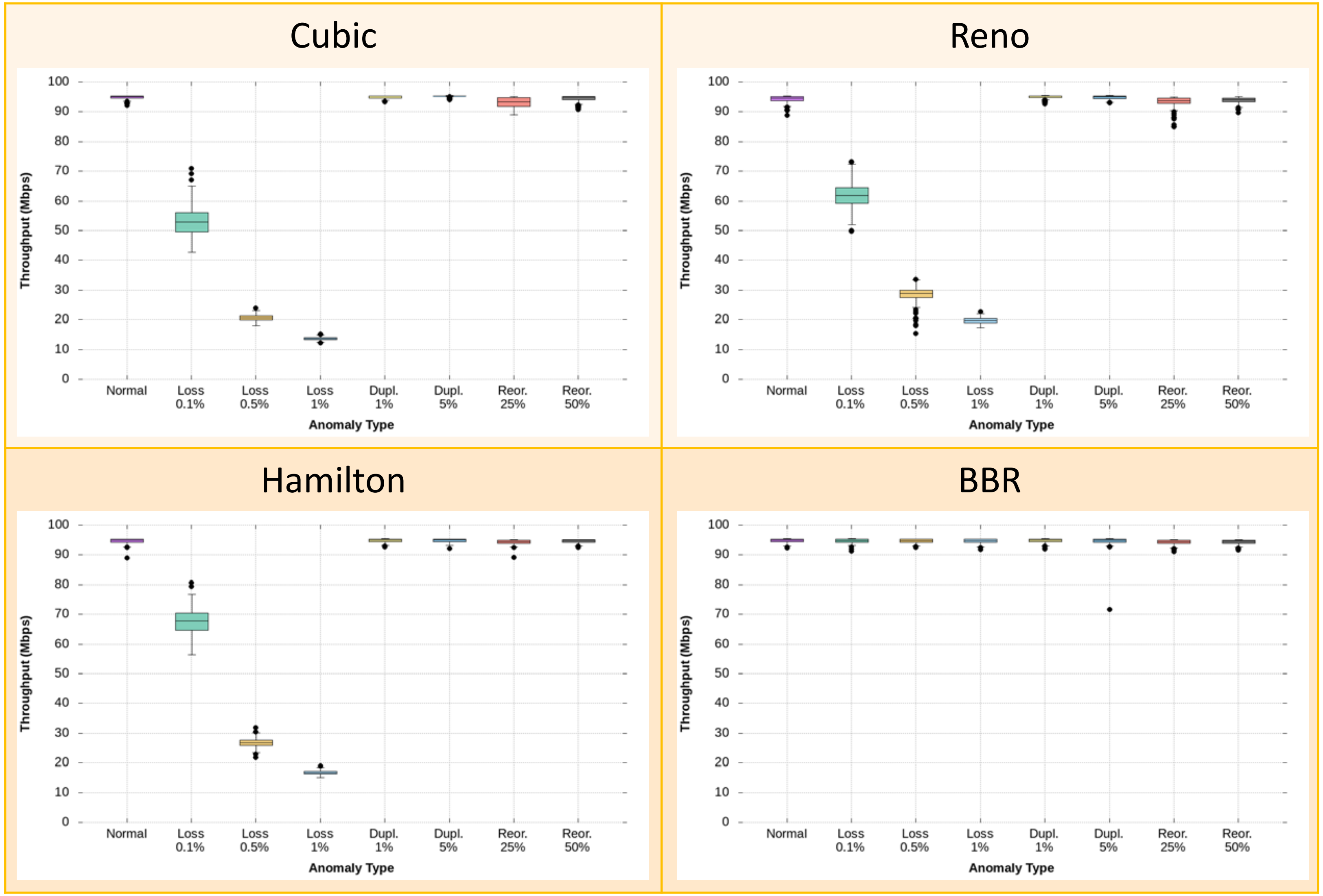


Initial Analysis: Retransmits - Mice

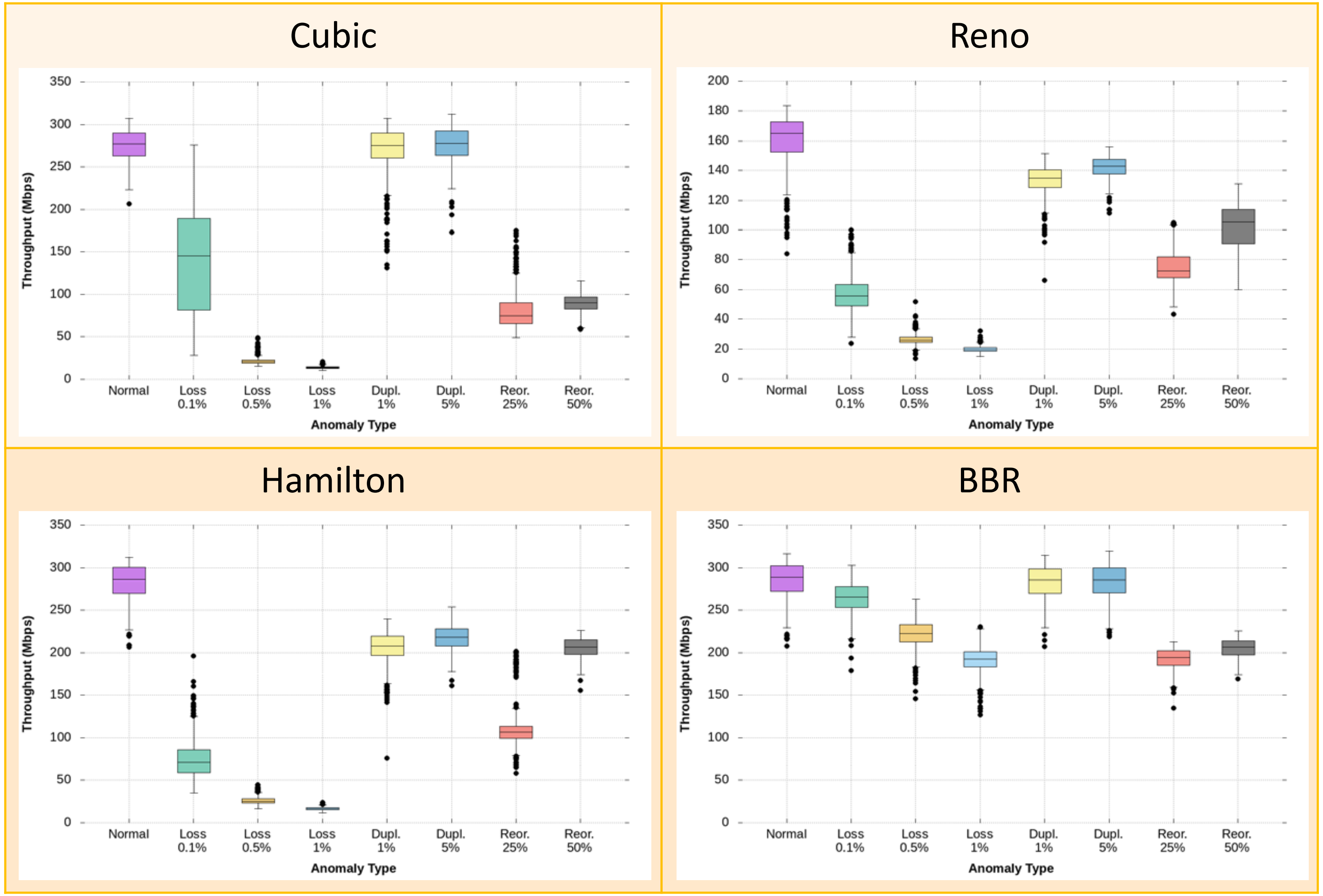


Elephant flows are long enough to reflect the loss, while mice are shorter flows

Initial Analysis: Throughputs – Elephant

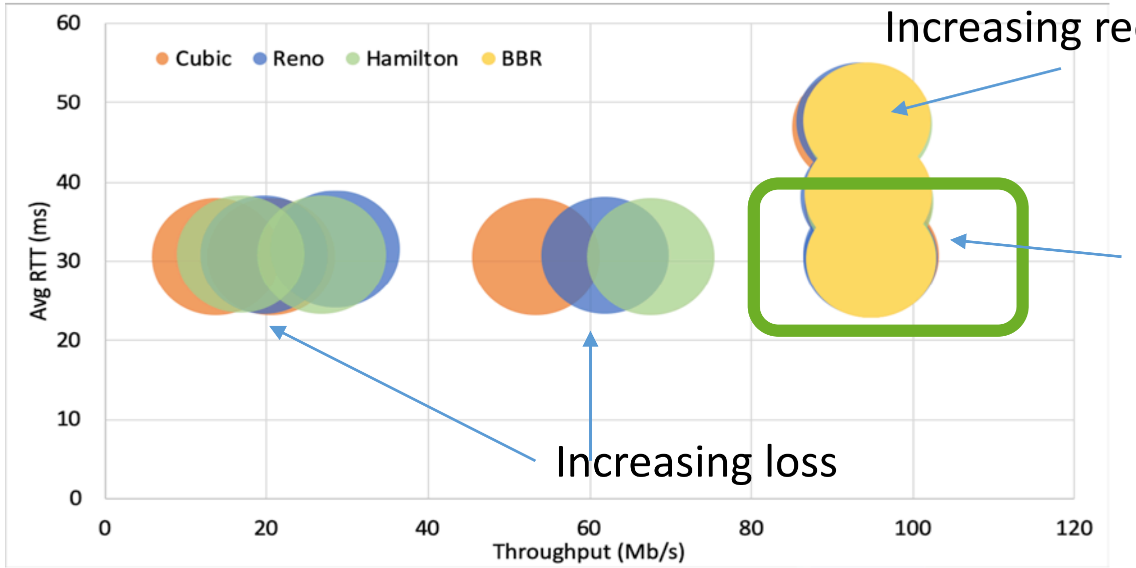


Initial Analysis: Throughputs – Mice



- While elephant flows have clear behavior, TCP slow start causes different behavior in mice flows

Initial Analysis: Relationship with RTT



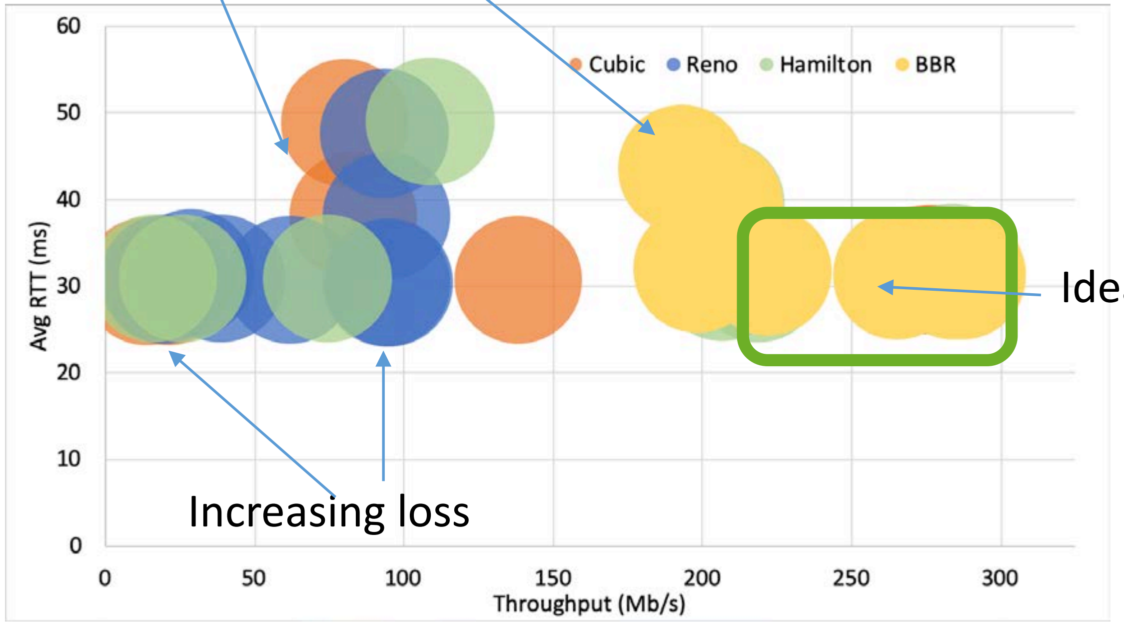
Increasing reorder %

Ideal

Elephant flows:
- High loss decreases Throughput, no affect on RTT
- High reorder increases RTT, no affect on Throughput

Mice flows:
- High loss decreases Throughput, no affect on RTT
- High reorder increases RTT and reduces Throughput

Increasing reorder %



Ideal

What we learn so far

- Elephant and mice flows behave differently - if we rely in just looking at retransmits and throughputs, this will not be enough
- Each TCP congestion algorithm behaves differently
- Critiques: We have a STRONG case for having a Classifier

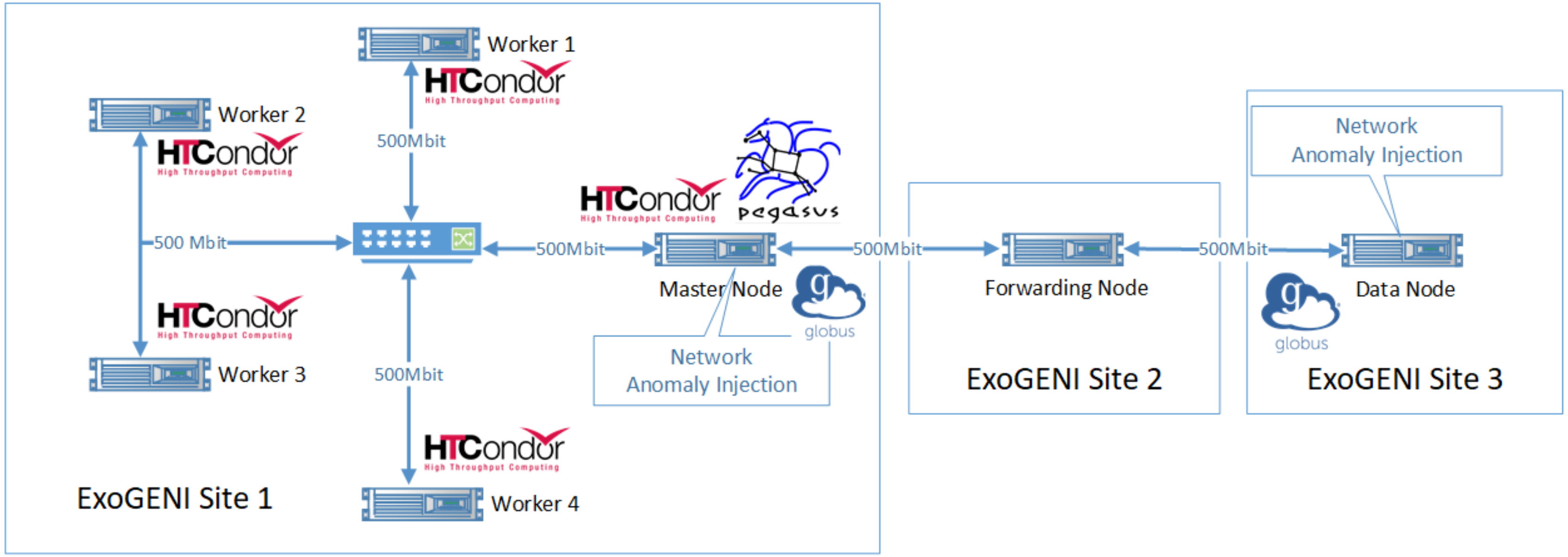
Which ML to use?

- We have labeled data sets
- We experimented with unsupervised classification techniques, but results were not promising:
  - difficult to understand how the classifier was making the decision
- Supervised classification techniques: Decision tree and random forest
  - White box techniques
  - Outputs all the rules the classifier learns from the training data
- Results here are presented using Random forest tree

Building the Test Dataset: Experiment Setup

- We used HTCondor and Pegasus WMS to execute a data intensive workflow that processes data from the 1000 Genome project.
- Transfers were carried out by the Globus transfer service.
- To create our experimental environment we used the ExoGENI Testbed.

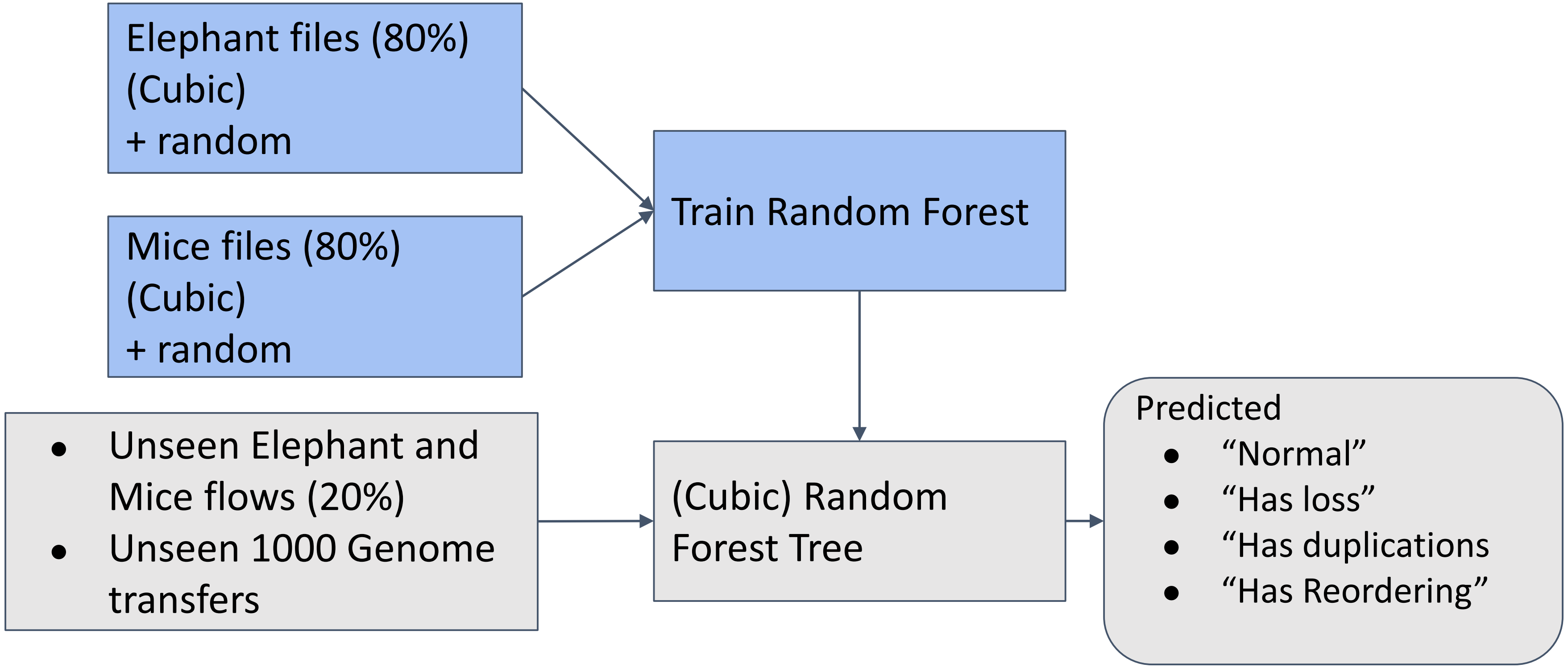
- Capacity on the links was 500Mbit.
- And anomalies were introduced at both Master and Data nodes.



Test data collected: 1000Genome - TCP transfers

Type	TCP Congestion Algorithm			
	Cubic	Reno	Hamilton	BBR
Normal	257	265	258	221
Loss 1%	273	257	277	225
Loss 3%	281	277	289	0
Loss 5%	285	277	273	0
Dupl. 1%	265	265	265	225
Dupl. 5%	265	265	265	217
Reord. 25%	265	269	264	217
Reord. 50%	269	253	302	217

Overall architecture



- We construct 4 different classifiers: Cubic, Reno, Hamilton and BBR
- Hyperparameter tuning: each tree tuned separately for optimal results

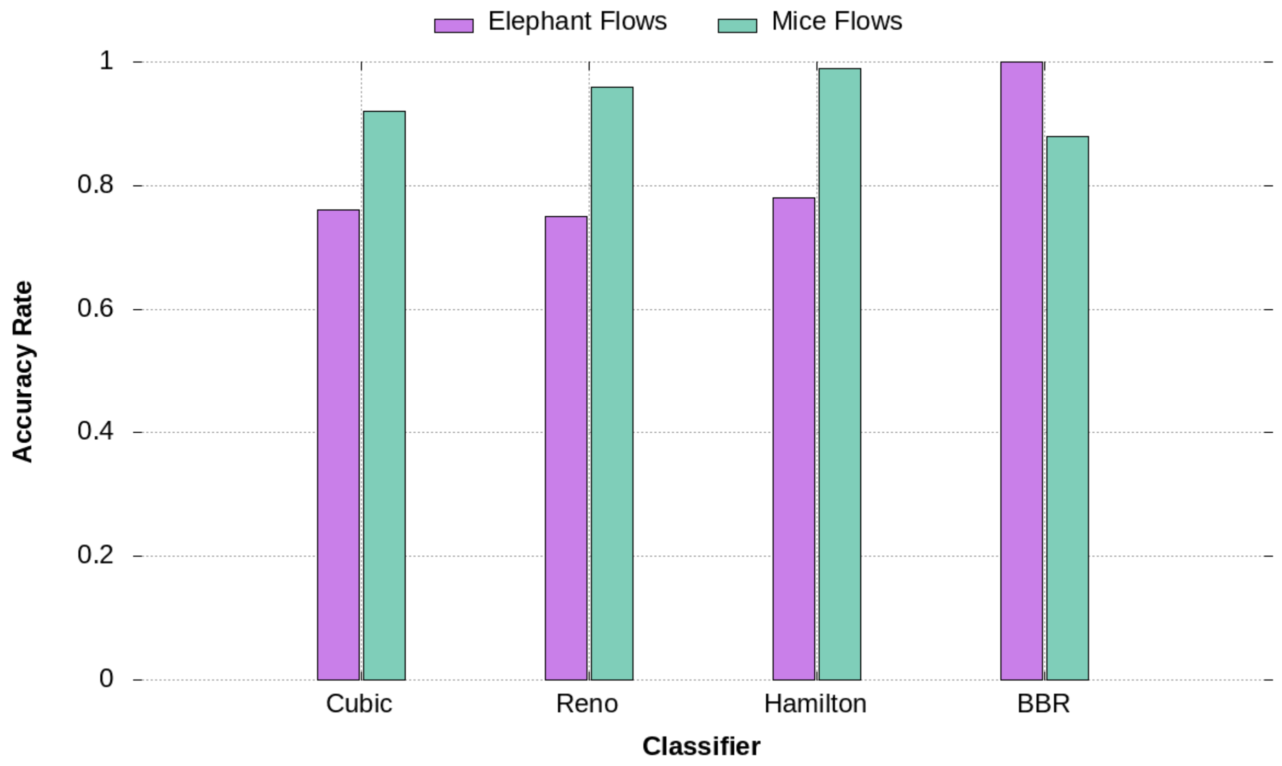
Problem found: Data leakage

- Classifier is generalizable?
  - Get 100% classification results on test elephant and mice flows
  - This is bad: not generalizable to workflow transfers
- Data leakage: One feature using a simple rule in responsible for recognizing a class, e.g.:
  - Retransmits> 10,000 -> loss (might not be true in other workflow transfers)
- To make classifier generalizable:
  - Add randomness to the training data
  - Turn pure to Impure training data: improve accuracy

Predictions
Elephant and Mice flows

Accuracy of Classifier – Mice and Elephant Flows

- Recognize anomalies in mice flows better than elephant
- Rules recognized for Elephant flows:
  - Duplication not recognized in Cubic and Reno
  - Reordering behavior:
    - Retransmits from Server side (s_bytes_retx) are less
    - First ACK received (c_first_ask) high



Predicting Anomalous Workflow Transfers

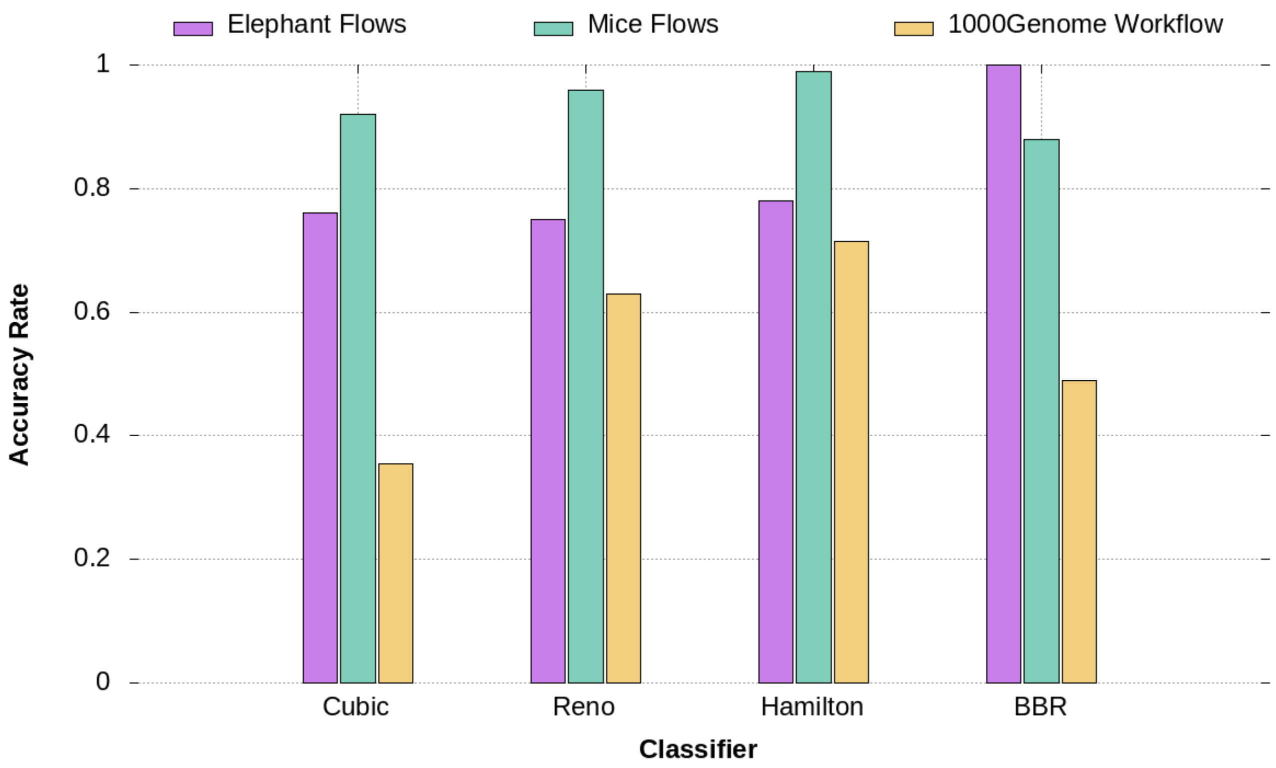
- Testing on unseen 1000 Genome flows
- Using Globus to transfer (4 parallel streams):
  - Classifiers recognize them as mice flows

Recognized?	Cubic	Reno	Hamilton	BBR
Normal	50% recognized	Y	Y	Y
Loss	60% recognized	50% recognized	80% recognized	20% recognized
Duplication	30% recognized	X	60% recognized	X
Reordering	Y	Y	40% recognized	80% recognized

Predictions
Scientific Workflow Data

Accuracy for Workflow Test Data

- Hamilton Classifier performs better
- Issue:
  - Need more flows with mixed characteristics to improve classifier



Future Work Extensions

- Classifier is learning unique behaviors of TCP congestion algorithm
- Improve this by improving the training data used (more diverse flow distributions)
- ML approach: Random forest is a rule-based approach on features
  - Not learning feature relationships
- Solution: Exploring Deep Neural Network may improve results
  - Learn weights among the features rather than values (prevent data leakage)