SAND2019-7399PE

# Taxonomist: Application Detection through Rich Monitoring Data

Emre Ates[†], Ozan Tuncer[†], Ata Turk[†], **Vitus J. Leung**[‡], Jim Brandt[‡], Manuel Egele[†], and Ayse K. Coskun[†]

[†] Boston University, Boston, MA

[‡] Sandia National Laboratories, Albuquerque, NM

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

# Why Application Detection

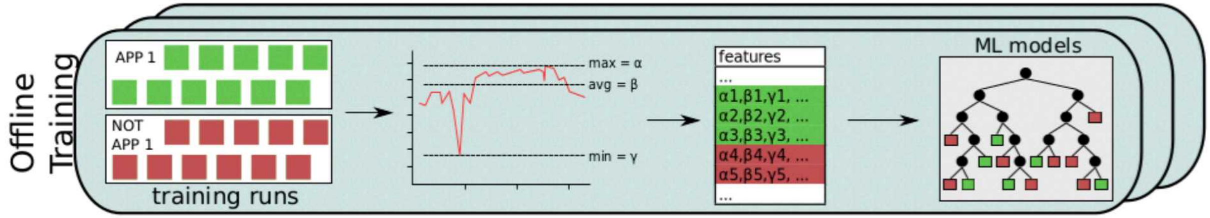
- Supercomputers
  - 100s of users, 1000s of projects
  - Submitted using hard-to-parse scripts
  - Binaries compiled elsewhere
- Current operators don't know which applications are running

# Benefits of Application Detection

- Significant body of work for application specific system management
  - Lower network contention *[Bhatele et al., SC'13]*
  - Lower power consumption *[Auweter et al., ISC'14]*
- Better management of resources
  - Assign developers to most used application
  - Decisions about the next generation system

# Contributions

- Taxonomist: A technique to identify running applications using numeric monitoring data.



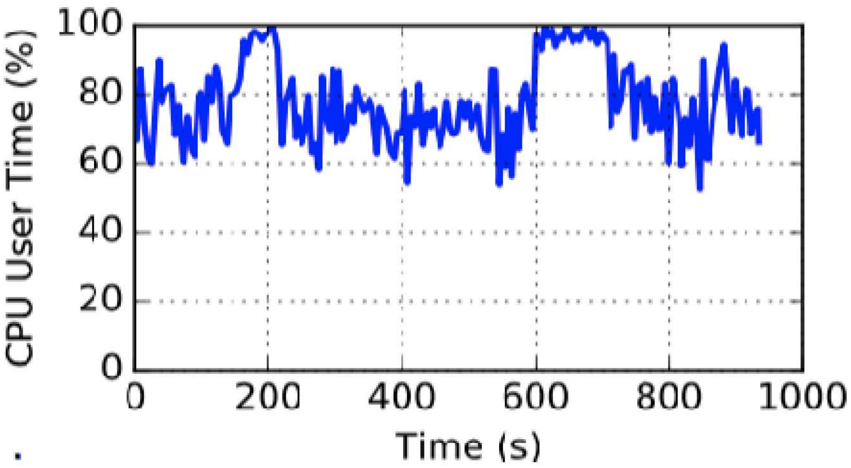

- Evaluation on Volta with benchmarks, cryptocurrency miners, normal system usage. Over 95% F-score

# Other Approaches

- Comparing application binaries (BinDiff) *[Flake et al., DIMVA'04]*
  - Shown to be not accurate enough *[Egele et al., USENIX Security'14]*
- MPI calls, communication patterns

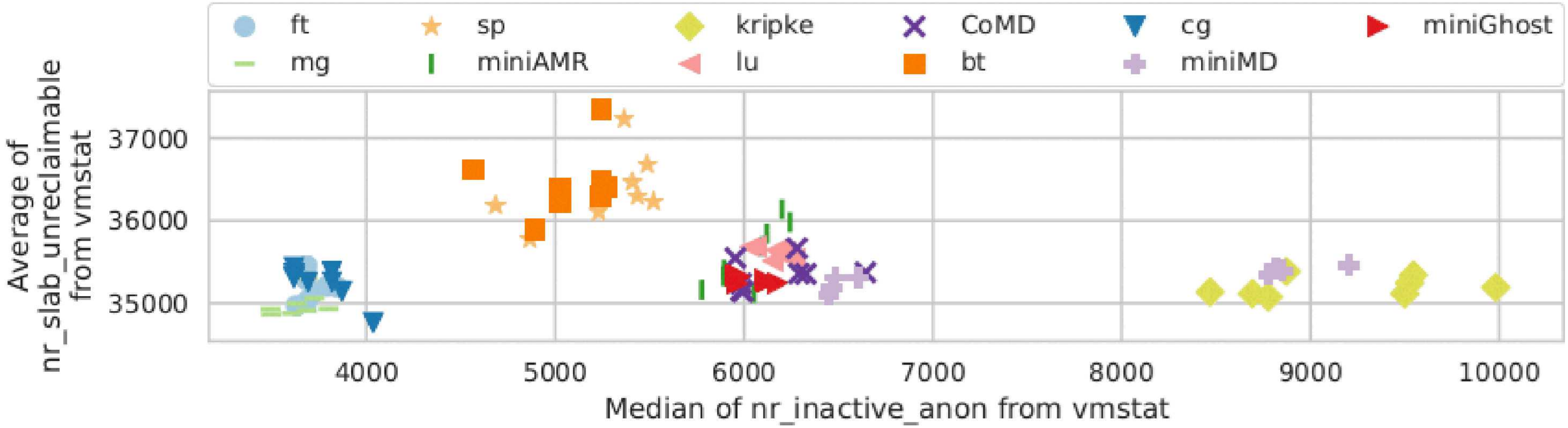
  *[DeMasi et al., CLHS'13, Whalen et al., Pattern. Recognit. Lett.'13]*
  - 5% overhead to intercept – Too high
- Power signatures *[Combs et al., E2SC'14]*

# Numeric Monitoring Data



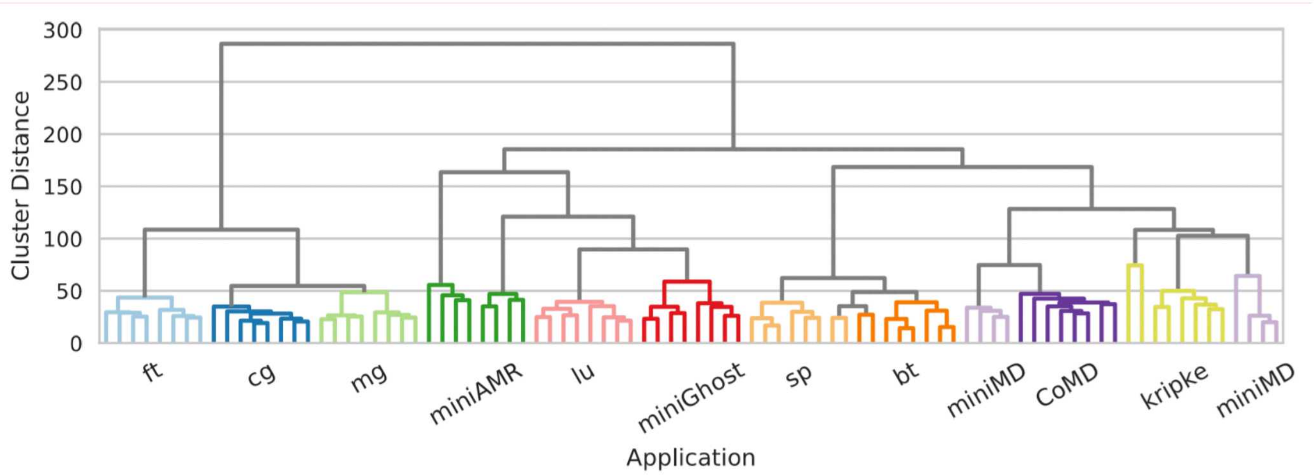

- Contents
  - OS Counters
  - System Utilization
  - Network/Hardware Counters
- Challenge:
  - Vast data volume: TBs per day, 100s of metrics
- Advantage:
  - Covers many subsystems, separates nodes
  - Readily available

# Numeric Monitoring Data



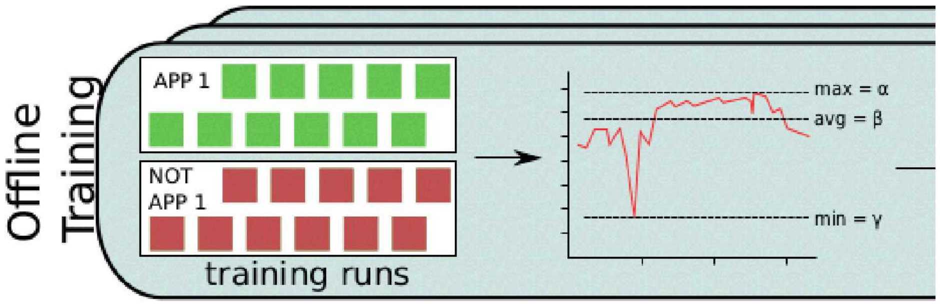

- Applications are naturally split into groups by resource allocation
- Two metrics are not adequate to split many applications
  - Thankfully we have more

# Resource Usage Fingerprints



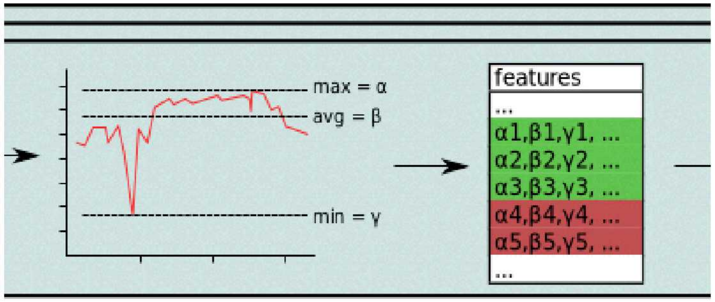

- Clustering of 11 applications – unsupervised
- Each application has its own resource usage fingerprint
- Not perfect, but promising

# Taxonomist: Data Collection




- Collect data from applications of interest
- 100s of time-series per node

# Taxonomist: Feature Generation




- Use statistical features to summarize time-series
- Keep the trend, remove noise

# Feature Extraction

**Min, max, average** Basic features

**Percentiles** 5th, 25th, 50th, 75th, 95th

**Standard Deviation** Amount of dispersion

$$\sigma = \frac{1}{N}\sqrt{\sum_{t=1}^{N}(x_t - \mu)^2}$$

**Skewness** Lack of symmetry

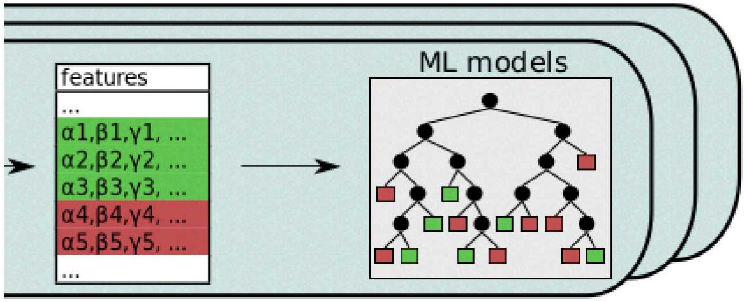
$$S = \frac{1}{N\sigma^3}\sum_{t=1}^{N}(x_t - \mu)^3$$

**Kurtosis** Heaviness of the tails

$$K = \frac{1}{N\sigma^4}\sum_{t=1}^{N}(x_t - \mu)^4$$

$N$: length of time-series
$\mu$: mean
$\sigma$: standard deviation
$x_t$: value at time $t$

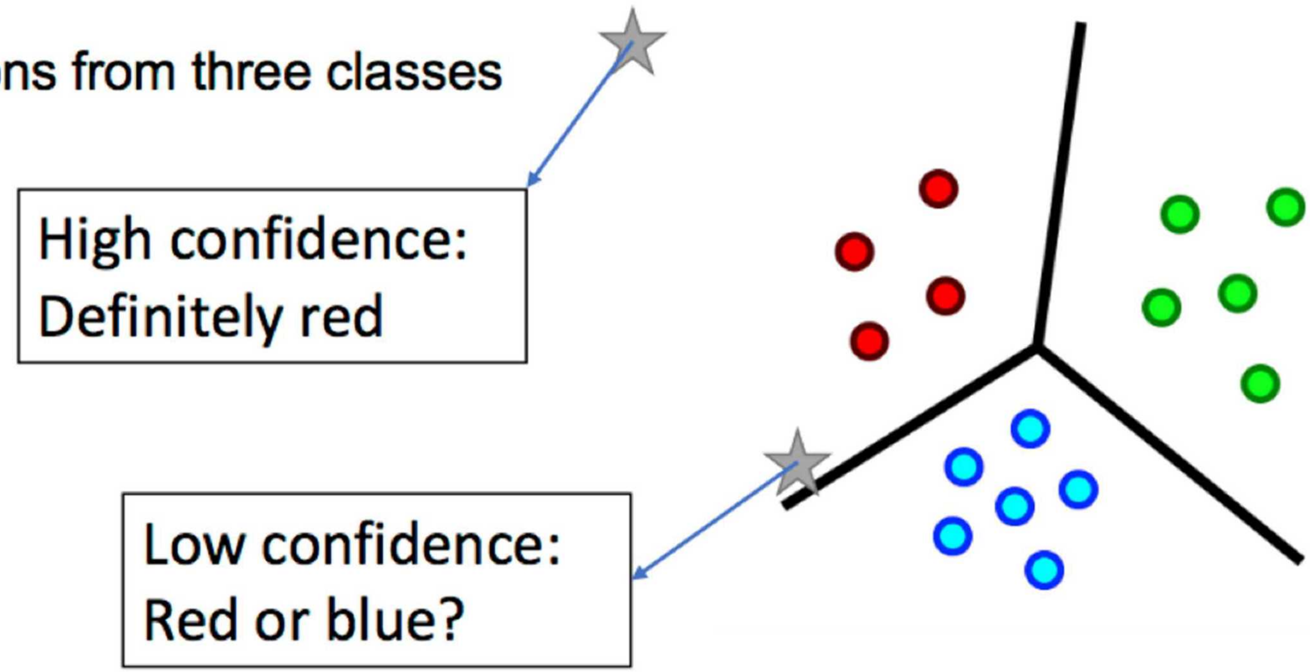
# Taxonomist: Training



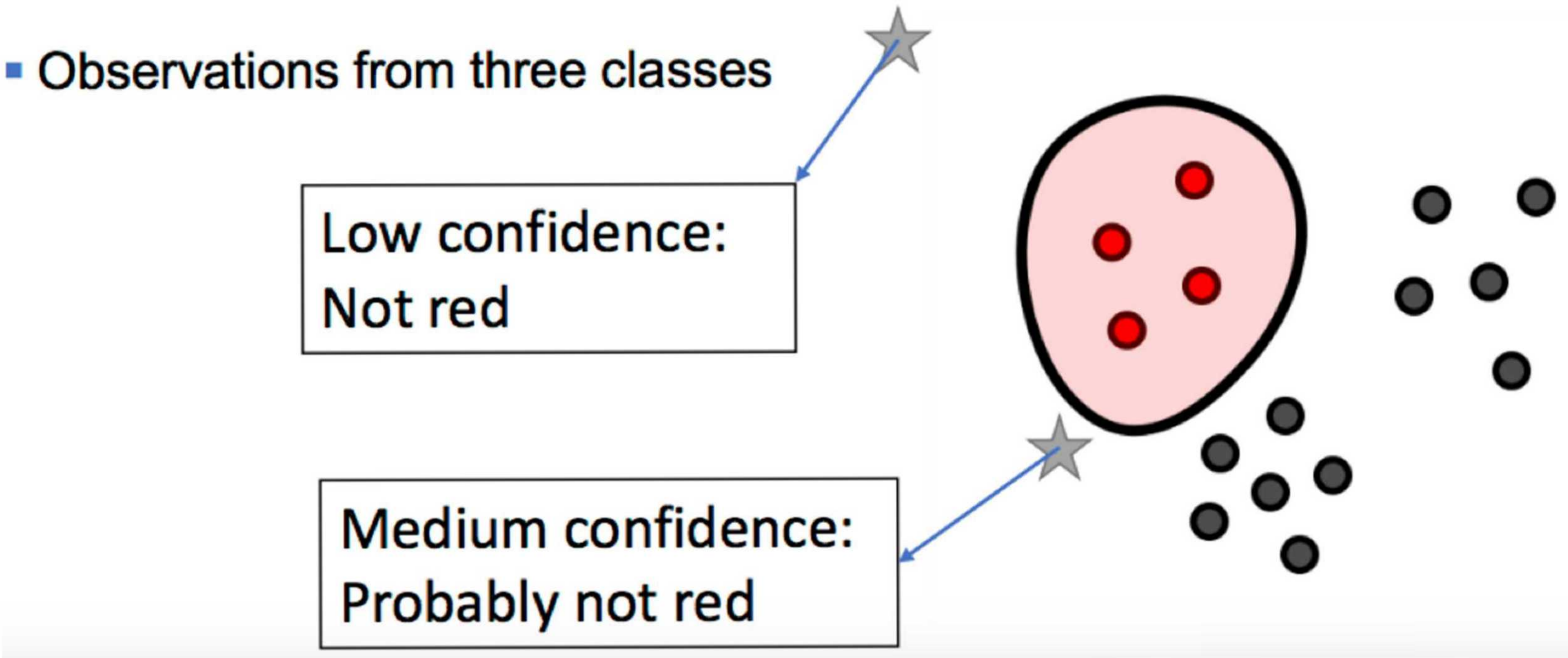

- Machine learning models are trained per application
  - One versus rest
- Tested with Random Forest, Decision Tree, SVM, Extra Trees
- Parameter tuning: perform 5-fold cross validation within training set
  - Pick model parameters with best f-scores

# Multiclass Classifier

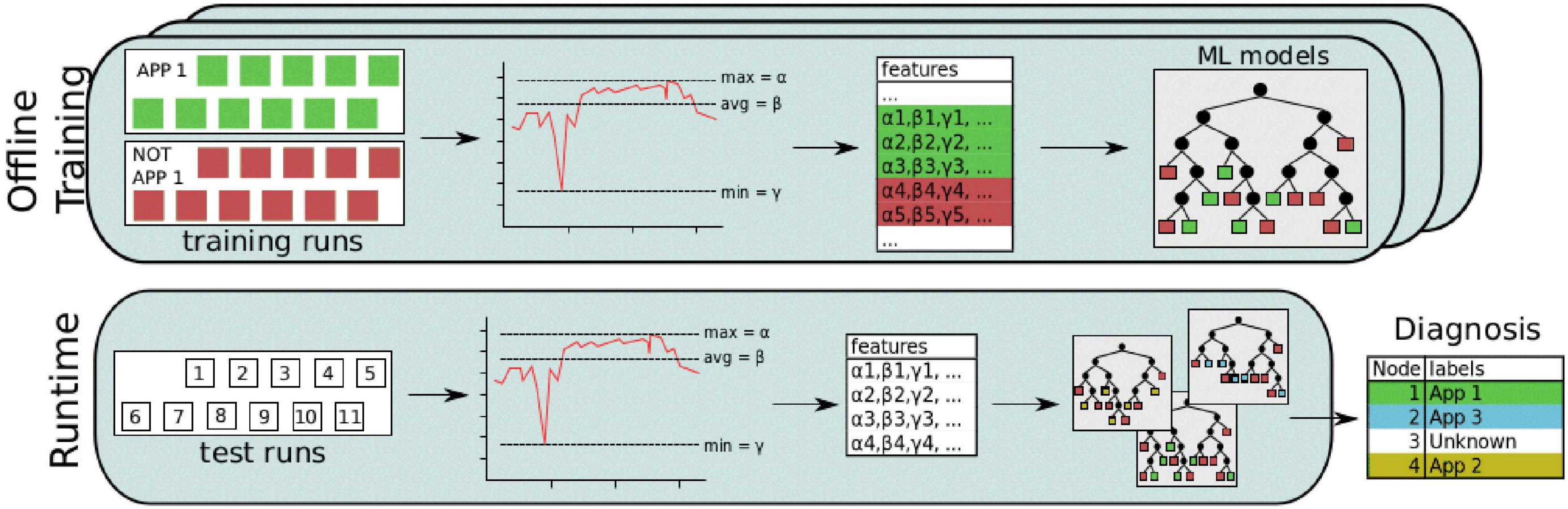
- Observations from three classes



# One vs Rest Classifier



# Taxonomist: Runtime




- At runtime, take prediction confidence from every classifier
  - If confidence is under a threshold, mark as *unknown*

# Methodology

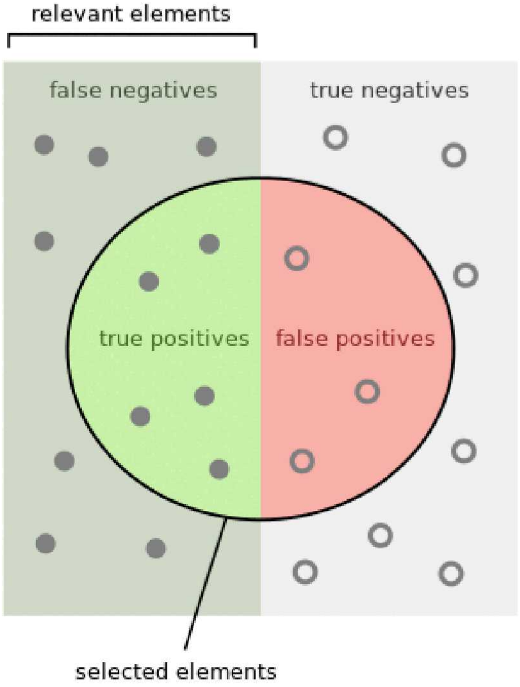
- System:  Volta – Cray XC30m supercomputer at SNL
  - 52 nodes
- LDMS (Lightweight Distributed Metric Service *[Agelastos et al., SC’14]*
  - 721 metrics per node every second
  - <2 MB RAM usage per node, CPU overhead ~0.01%
- Baseline Method – Combs:  *[Combs et al., E2SC’14]*
  - Only input is power consumption – already collected
  - More features – serial correlation, nonlinearity, trend
  - Random forest classifier, no support for unknown applications

# Applications

- 3-4 input configurations per application, running on 4-32 nodes
  - BT, CG, FT, LU, MG, SP from NAS Parallel Benchmarks *[Bailey et al., j-IJSA'91*
  - miniAMR, miniMD, CoMD, miniGhost from Mantevo *[Heroux et al., SNL'09]*
  - Kripke from LLNL Proxy Applications *[Kunen et al., LLNL'15]*
- 6 unwanted applications
  - 5 cryptocurrency miners
  - 1 password cracker

# Evaluation

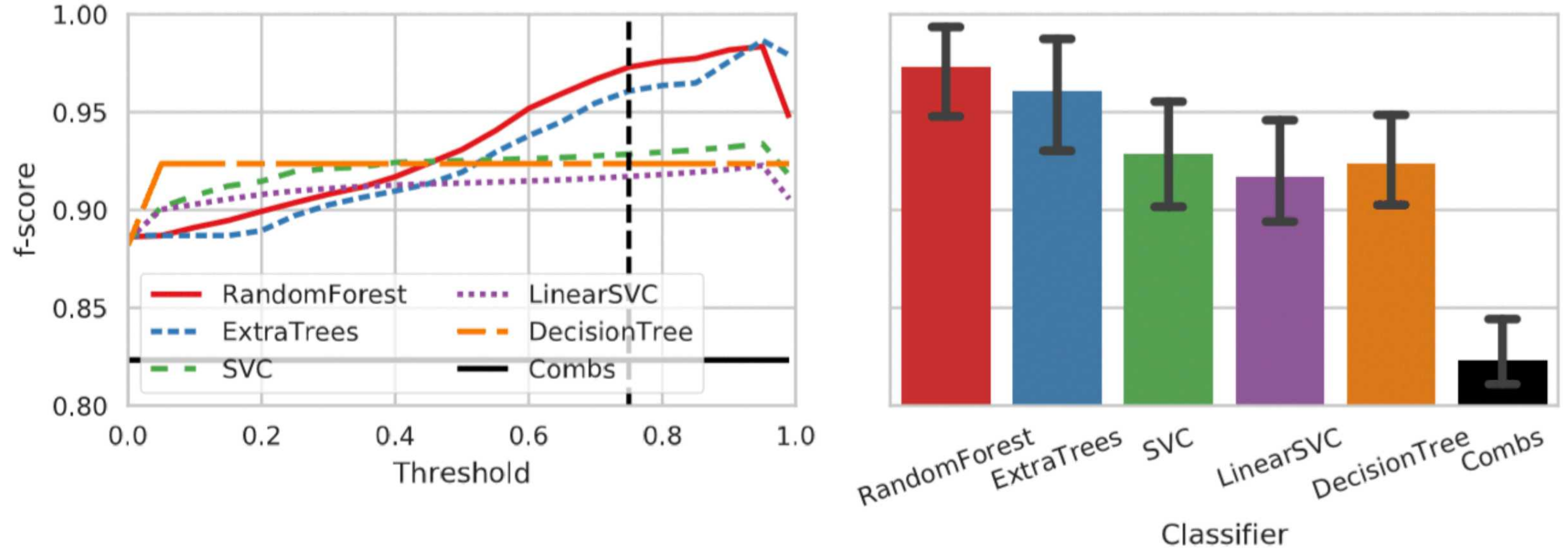
- F-Score
  - Harmonic mean of precision and recall


*[Walber, Wikimedia, 2014]*

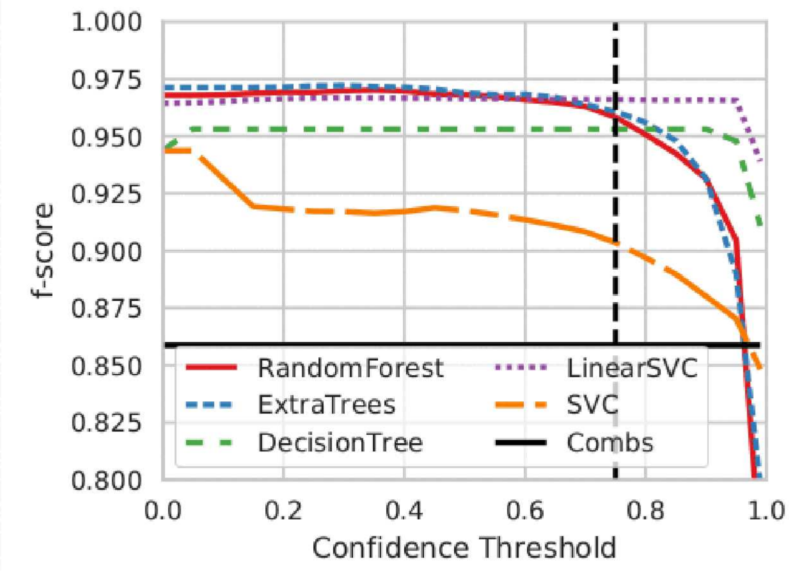
# Evaluation

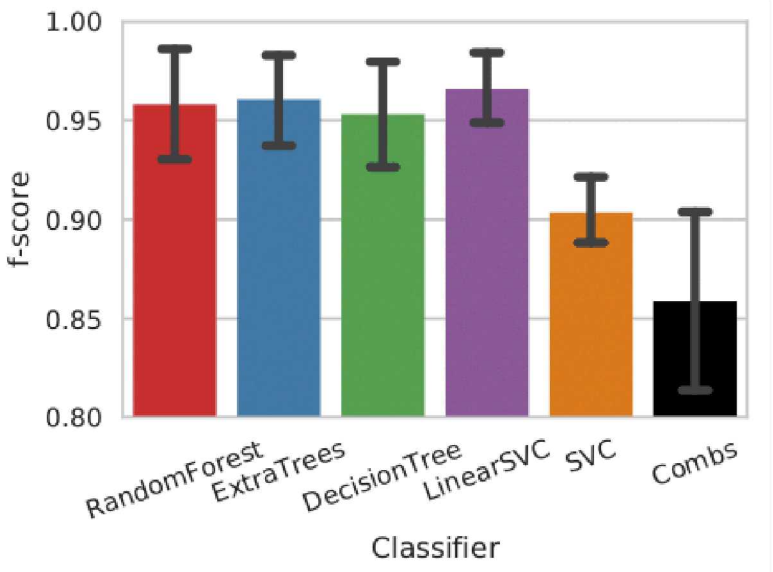
- Train with 10 applications, test with 11
  - Unknown application correctly identified



# Evaluation

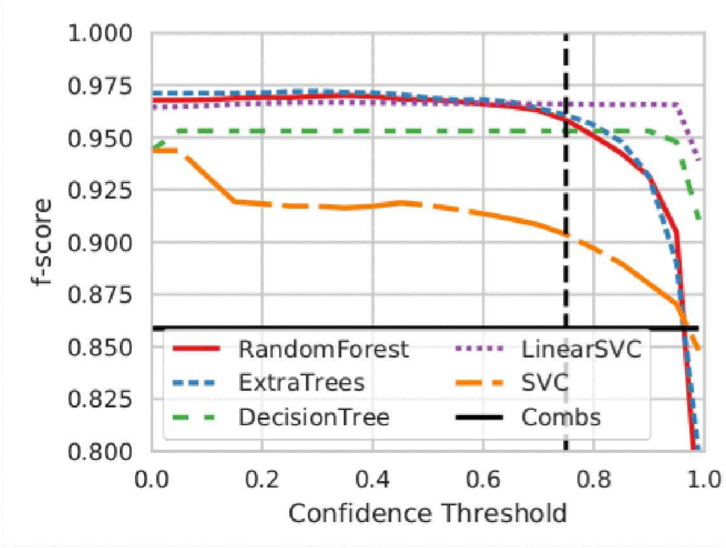
- Train with one input configuration missing
  - Unknown input correctly identified

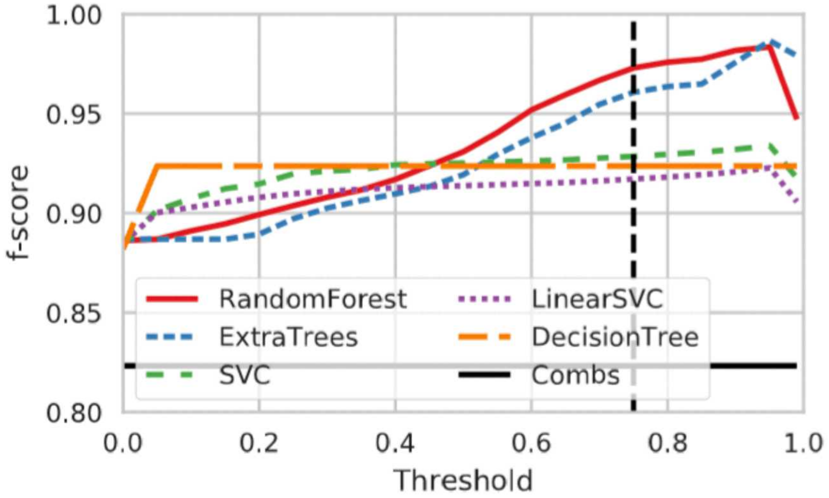


# Confidence Threshold Selection

- For training set, perform two tests
  - Remove one application from training, sweep over thresholds
  - Remove one input from training, sweep over thresholds

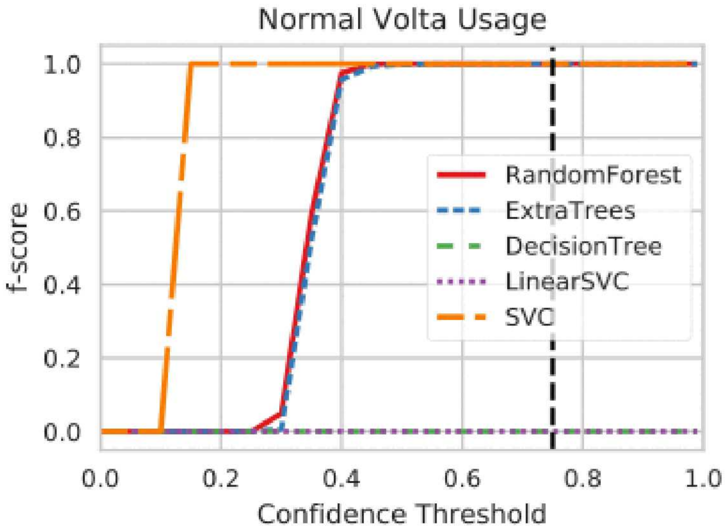


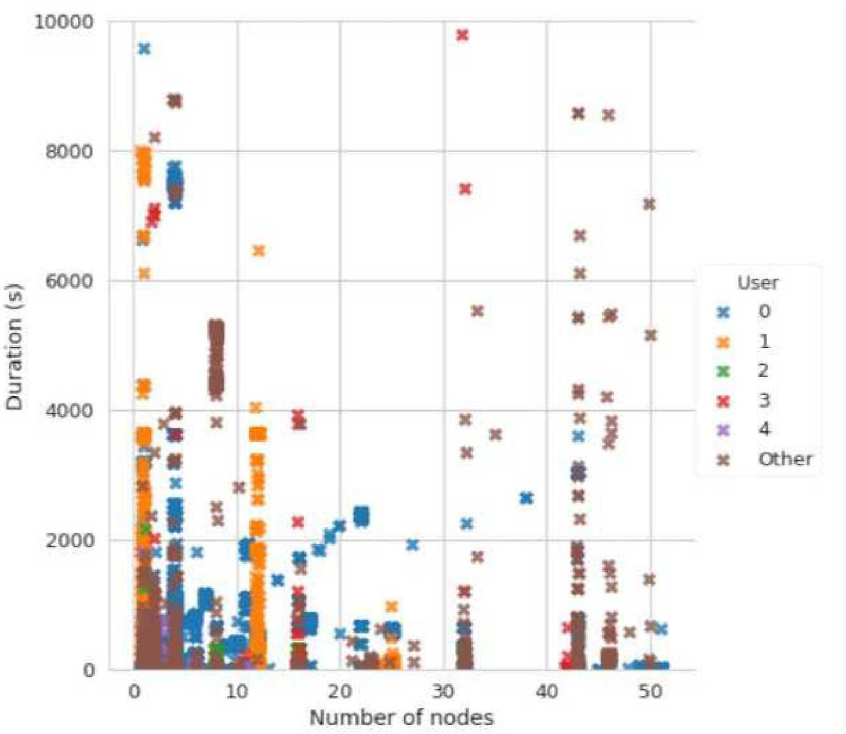



# Evaluation

Train with 11 applications

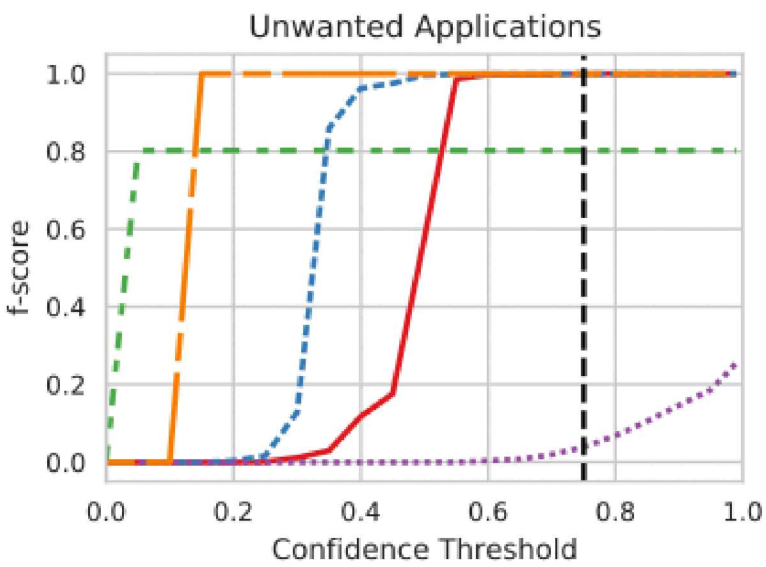
Train with Volta usage over 6 months



# Evaluation

- Train with 11 applications
- Test with 5 cryptocurrency miners and password crackers
- Perfect classification



# Summary

- A technique to identify applications
  - Based on resource usage data
  - Using concise features, low-overhead
- Our technique outperforms existing methods
  - Over 95% F-score
- Artifact with data, code, more plots: https://doi.org/10.6084/m9.figshare.6384248