

# Comparative Analysis of Classical & Deep Learning Anomaly Detection Methods

Benchmarking Server Monitoring (SMD) & Cybersecurity (BETH) Datasets under Oracle and Deployable Thresholds

Authors: Joshua Streets & Husnu S. Narman | Dept. of Computer Science, Marshall University



SPONSORED BY

**Institute of Cyber Security**

Marshall University

# Motivation: Do Deep Models Beat Classical Methods in Realistic Deployment?

---

## MONITORING & CYBERSECURITY

### Why Unsupervised Detection

- Anomalous events are rare and hard to model explicitly.
- Unsupervised methods learn normal behavior and flag deviations without labeled anomalies.
- Excessive false positives reduce the practical use of detectors.

## THE RESEARCH QUESTION

### Are Deep Models Always Superior?

- LSTM-based deep learning has gained popularity for modeling temporal dependencies.
- Do deep models outperform classical methods once thresholds must be estimated without test labels?
- Advantages may vary by dataset, evaluation strategy, and domain.

## THE EVALUATION GAP

### The Evaluation Gap: Oracle vs. Deployable

- Oracle thresholds use test labels: a useful upper bound, but not deployable.
- Few studies compare classical and deep learning methods while also measuring this gap.
- Threshold estimation is underexplored relative to model architecture.

# Key Contributions: Classical vs. Deep Learning, Oracle vs. Deployable

---

01

## Classical vs. Deep Learning Comparison

Three classical methods (PCA, IForest, KNN) vs. two LSTM models on two benchmarks: SMD (server monitoring) and BETH (cybersecurity).

02

## Oracle vs. Deployable Assessment

Comparing oracle thresholds (using test labels) with validation derived thresholds ( $\mu + 3\sigma$ ,  $\mu + 4\sigma$ , POT) to measure the gap between theoretical and practical performance.

03

## Threshold Estimation Under Distribution Shift

Investigating how differences between validation and test score distributions affect deployable performance, evaluated with F1 and F0.1.

# Evaluated Methods: 3 Classical Baselines vs. 2 Deep Learning Architectures

---

## CLASSICAL BASELINES

- **Principal Component Analysis (PCA)**

Subspace reconstruction error, keeping components that preserve 95% of the training variance.

- **Isolation Forest (IForest)**

300 isolation trees. Anomalies are isolated in fewer splits (shorter average path length).

- **K-Nearest Neighbors (KNN)**

Euclidean distance to the 20th nearest neighbor ( $k = 20$ ); larger distance means more anomalous.

## Deep Learning Architectures

- **LSTM Encoder-Decoder**

Reconstruction-based sequence model. Three-layer encoder ( $128 \rightarrow 64 \rightarrow 32$  units) with a mirrored decoder. Score is Euclidean reconstruction error over the window.

- **LSTM Forecaster**

Forecasting model (single 64-unit LSTM layer). Predicts the next observation from the previous 100. Score is Euclidean prediction error.

# Benchmark Datasets: Multi-Machine Telemetry & Cybersecurity Attacks

---

## SERVER MACHINE DATASET (SMD)

**Domain:** Server monitoring telemetry (large internet company)

**Scale:** 28 server machines, each evaluated independently

**Features:** 38 monitored system metrics per observation

**Sampling:** 1 minute intervals; train/test split provided

**Labels:** Only test sets contain anomalies, labeled by domain experts from incident reports.

## BETH CYBERSECURITY DATASET

**Domain:** Cybersecurity host telemetry

**Scale:** 23 cloud-hosted honeypots, >8 million events

**Features:** eBPF-based host monitoring

**Content:** Benign activity and real attack behavior

**Labels:** Predefined train/validation/test partitions; only the test partition contains malicious activity.

# Evaluation Framework: Oracle Upper Bound vs. Deployable Thresholding

---

## STEP 1: PIPELINE SETUP

### Preprocessing & Windowing

- Min-Max scaling fit on training data and applied to training and testing data.
- Windows of 100; a window is anomalous if any observation in it is.
- SMD: 70/30 train/validation split of training data; provided test set.
- BETH: predefined partitions; stride of 5.

## STEP 2: ORACLE STRATEGY

### Theoretical Upper Bound

- Evaluates every unique test score.
- Selects the threshold maximizing F1 using test labels.
- Not deployable: used only as an upper-bound reference.
- Metrics: F1 balances precision and recall equally; F0.1 ( $\beta = 0.1$ ) weights precision far more, penalizing false alarms.

## STEP 3: DEPLOYABLE STRATEGY

### Validation-Based Thresholds

- Thresholds from validation scores only; test labels never used.
- Statistical:  $\mu + 3\sigma$  and  $\mu + 4\sigma$ .
- Peak Over Threshold (POT) via Extreme Value Theory.
- Reported result = best of the three per model and dataset.

## Finding 1: Oracle Thresholds Overestimate Deployable Performance on SMD

| Model           | Oracle F1 | Deployable F1 | Deployable F0.1 |
|-----------------|-----------|---------------|-----------------|
| PCA             | 0.604     | 0.296 (Top)   | 0.378           |
| IForest         | 0.376     | 0.222         | 0.373           |
| KNN             | 0.513     | 0.232         | 0.312           |
| LSTM Enc/Dec    | 0.496     | 0.226         | 0.295           |
| LSTM Forecaster | 0.409     | 0.227         | 0.421 (Top)     |

### CORE INSIGHTS

- Every Method Drops: F1 falls by roughly 41% (IForest) to 55% (KNN) from oracle to deployable thresholds.
- PCA Leads Deployable F1: PCA reaches 0.296, ahead of both LSTMs, and methods converge (0.222–0.296).
- LSTM Forecaster Leads F0.1: the top precision-weighted score (0.421) comes from a deep model.

Averages across SMD machines. Deployable = best of  $\mu + 3\sigma$ ,  $\mu + 4\sigma$ , POT per model.

# Finding 2: Workload Sensitivity & Machine-Specific Behavior (SMD Case Study)

## SMD MACHINE 2-5 (TABLE II)

- **PCA:** Oracle F1 of 0.757 falls to 0.249 deployable; possibly due to validation vs. test score distribution differences.
- **LSTM Forecaster:** Best deployable F1 (0.412), far above the best classical method (PCA, 0.249); its oracle F1 was only 0.425.
- **F0.1 differs:** PCA has the top deployable F0.1 (0.919 vs. 0.705 for the Forecaster).

| Model           | Oracle F1 | Deployable F1 | Deployable F0.1 |
|-----------------|-----------|---------------|-----------------|
| PCA             | 0.757     | 0.249         | 0.919 (Top)     |
| IForest         | 0.327     | 0.169         | 0.698           |
| KNN             | 0.407     | 0.204         | 0.769           |
| LSTM Enc/Dec    | 0.356     | 0.202         | 0.767           |
| LSTM Forecaster | 0.425     | 0.412 (Top)   | 0.705           |

SMD machine 2-5 (Table II). Deployable = best of  $\mu + 3\sigma$ ,  $\mu + 4\sigma$ , POT per model.

# Implications: No Single Method Wins Everywhere

---

## **NO SINGLE WINNER**

Average scores hide machine level differences; no approach was best in every scenario.

## **THRESHOLD SENSITIVITY**

Thresholds rely on the upper tail of normal validation scores, so small shifts can change deployable results a lot.

## **APPLICATION DRIVEN SELECTION**

Choose methods by application needs (e.g., cost of false alarms), not average benchmark scores alone.

## Finding 3: Deployable Thresholds Generalize on BETH; Classical Methods Stay Competitive

| Model           | Oracle F1 | Deployable F1  | Deployable F0.1 |
|-----------------|-----------|----------------|-----------------|
| PCA             | 0.936     | 0.023 (Failed) | 0.367           |
| IForest         | 0.978     | 0.978          | 0.976           |
| KNN             | 0.980     | 0.980          | 0.992 (Top)     |
| LSTM Enc/Dec    | 0.973     | 0.972          | 0.963           |
| LSTM Forecaster | 0.982     | 0.981 (Top)    | 0.985           |

### BETH TAKEAWAYS

- Contrast with SMD: On BETH, deployable F1 is close to oracle for four of five methods (~0.97–0.98).
- Classical Matches Deep: KNN deployable F1 is 0.980 vs. 0.981 for the LSTM Forecaster, at lower computational cost. IForest (0.978) is close behind.
- PCA Exception: Oracle F1 0.936 but deployable 0.023; it separates the classes, but validation-based thresholding failed.

*BETH test partition. Deployable = best of  $\mu + 3\sigma$ ,  $\mu + 4\sigma$ , POT per model.*

# Study Limitations: Scope & Interpretation of Results

---

## **LIMITATION 01**

### **Threshold Estimation**

- Large oracle to deployable gaps remained on SMD, so results may be partly limited by thresholding, not scoring alone.
- Deployable results are the best of  $\mu + 3\sigma$ ,  $\mu + 4\sigma$  and POT per model and dataset, not one predetermined strategy.

## **LIMITATION 02**

### **Datasets & Tuning**

- Two datasets only (server monitoring, cybersecurity); other domains may differ.
- Baseline implementations without extensive dataset specific tuning.

## **LIMITATION 03**

### **Architecture Scope**

- Only two LSTM-based deep learning models were evaluated.
- Transformer and foundation model approaches may yield different results.

# Conclusions: Model Complexity Alone Does Not Guarantee Better Performance

## KEY TAKEAWAYS

- **Classical Competitiveness:** Even as simple baselines, classical methods stayed competitive. PCA had the best average deployable F1 on SMD; LSTMs gained on some machines.
- **Threshold Estimation Matters:** On SMD, oracle thresholds overestimated deployable performance for every method (e.g., PCA 0.604  $\rightarrow$  0.296). Thresholding may matter as much as model choice.
- **Dataset & Metric Dependence:** On BETH, deployable was close to oracle (PCA excepted). F0.1 rankings differ from F1, so selection should suit the application.

## FUTURE RESEARCH DIRECTIONS

- **Robust Threshold Estimation:** Techniques that hold up when validation and test distributions differ.
- **Additional Datasets:** Other application domains beyond SMD and BETH.
- **Modern Architectures:** Transformer-based and foundation model approaches.

**THANK YOU & QUESTIONS**

