

Predicting Enterprise Application Performance Measures through Time Series Forecasting

Daniel Elsner, 22nd March 2018, Scientific advisor: Pouya Aleatrati Khosroshahi

Chair of Software Engineering for Business Information Systems (sebis)
Faculty of Informatics
Technische Universität München
www.matthes.in.tum.de



Motivation



Research Questions



Research Contributions



Predicting Application Performance



Anomaly Detection

Problem Domains in Application Performance Monitoring (APM)



Performance



Availability



Maintainability

“Companies are sitting on a **treasure trove**
– if only they knew how to use it.”

A. Samuel, Wall Street Journal, 2015

Research Goals in APM

1

Forecast IT system performance for **pro-active** resource planning

2

Reduce IT system outages through **predictive maintenance**

3

Facilitate **root cause analysis** to decrease down-time



1

What are **research gaps** in existing scientific work regarding **application performance forecasting** and **detection of abnormal system behavior** in enterprise applications?

2

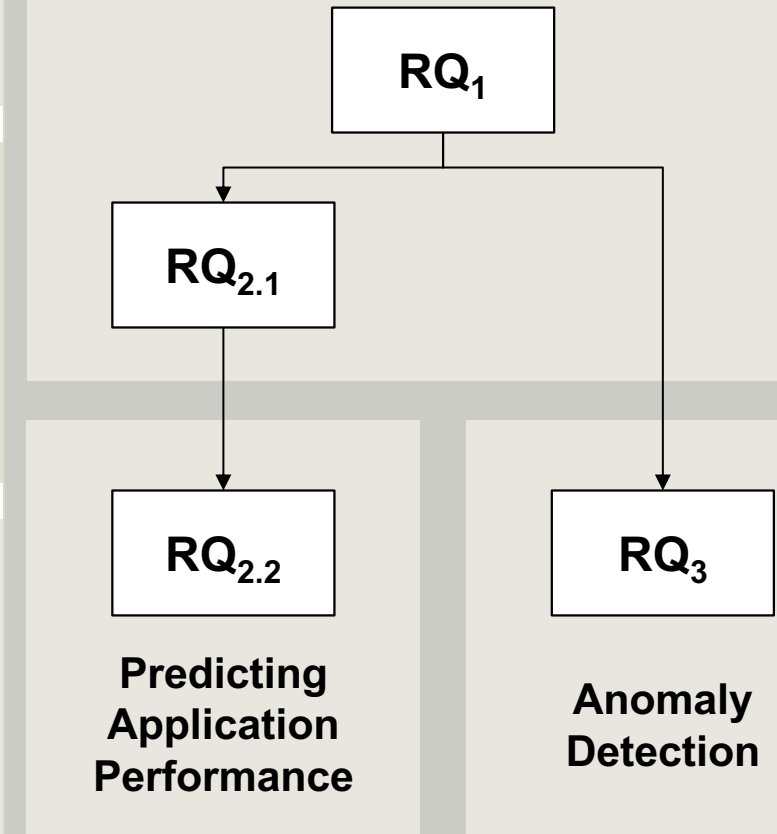
Which are **relevant APM measures to assess application performance** (RQ2.1) and how well can the identified be **predicted based on historical APM measures** (RQ2.2)?

3

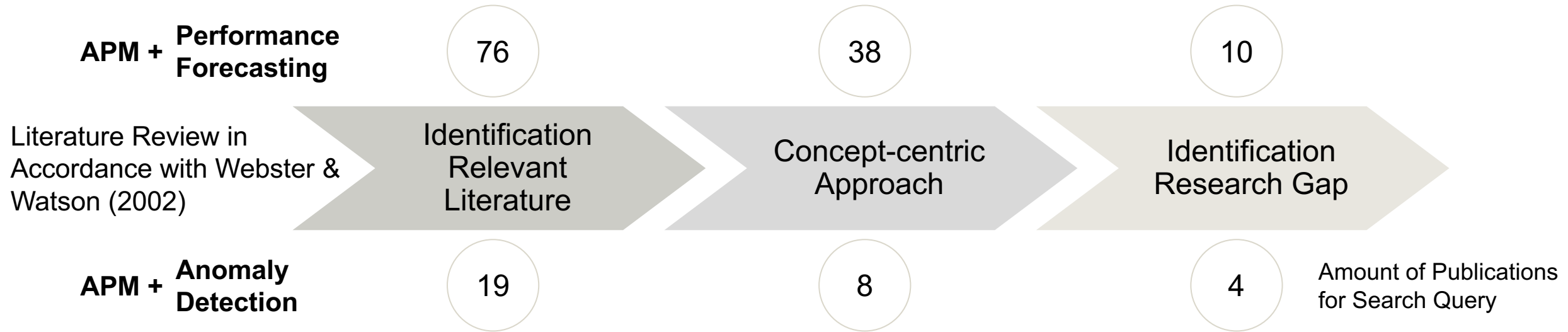
How well can **abnormal system behavior** of applications be **detected based on APM measures**?

Interconnectivity of RQs

Literature Review



Research Contributions (RCs)



1

Implementation and comparison of the **linear and nonlinear** ML techniques linear regression, tree-based regressors, and neural networks for **multivariate multi-step** time series **application performance forecasting**.

2

Implementation of a **multivariate density-based anomaly detection model** providing an indication for the **root cause of abnormal system behavior**.



Predicting Application Performance – Data Architecture

Layers

Sources

Client



Application

▲ ATERNITY®

Middleware



Web server

Web server

Nagios

Application server



Application server

Application server

Application server

Java

dynatrace

Database



Database

Database

InfluxDB
 Grafana



Identified Research Scope

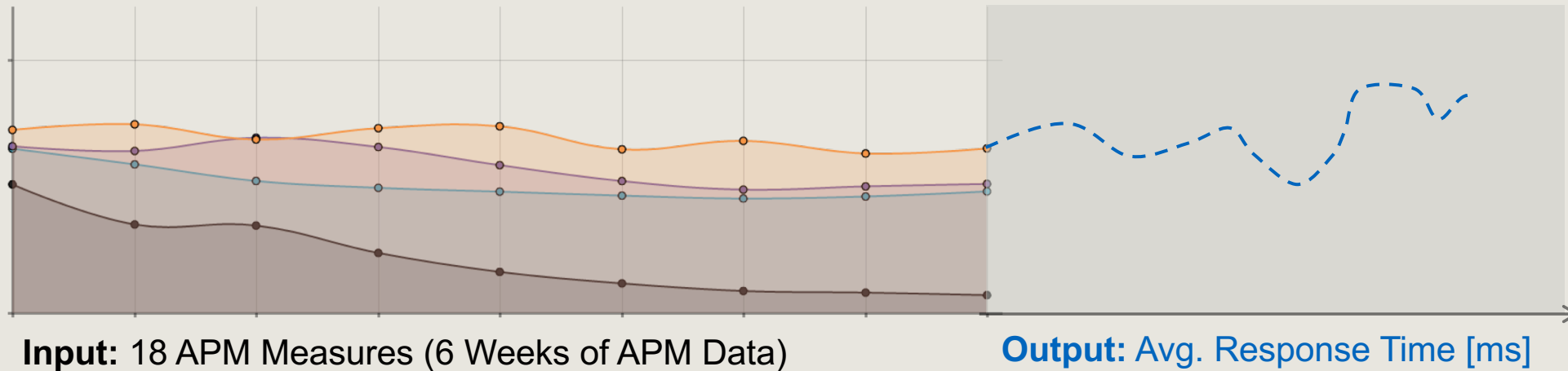
2 Applications



18 APM Measures as
1/min-1/60min Time Series

Multi-step Forecast

30 Minutes
Avg. Response Time



1

Linear vs. Non-Linear
Regression Models

2

Uni- vs. Multivariate
Modeling

3

Multi-step Forecasting
Strategy



Predicting Application Performance – Evaluation

APP1

| | Baseline |
|-----------------|----------|
| MAE [ms] | 266,29 |
| MAPE [%] | 0,30 |
| RMSE [ms] | 559,67 |
| RMSLE [log(ms)] | 0,42 |
| RMSPE [%] | 0,85 |

| | LinearRegression | LSTM | MLP | RandomForestRegressor |
|-------|------------------|-------------|--------|-----------------------|
| MAE | 212,43 | 213,81 | 220,60 | 205,64 |
| MAPE | 0,25 | 0,26 | 0,27 | 0,25 |
| RMSE | 424,29 | 418,54 | 431,55 | 415,88 |
| RMSLE | 0,33 | 0,33 | 0,33 | 0,32 |
| RMSPE | 0,76 | 0,74 | 0,80 | 0,76 |

| | LinearRegression | LSTM | MLP | RandomForestRegressor |
|-------|------------------|--------|--------|-----------------------|
| MAE | 223,81 | 233,64 | 220,28 | 196,97 |
| MAPE | 0,27 | 0,29 | 0,27 | 0,23 |
| RMSE | 431,05 | 439,65 | 422,86 | 398,02 |
| RMSLE | 0,34 | 0,35 | 0,33 | 0,30 |
| RMSPE | 0,76 | 0,77 | 0,78 | 0,70 |

APP2

| | Baseline |
|-----------------|----------|
| MAE [ms] | 195,67 |
| MAPE [%] | 0,36 |
| RMSE [ms] | 366,95 |
| RMSLE [log(ms)] | 0,48 |
| RMSPE [%] | 0,69 |

| | LinearRegression | LSTM | MLP | RandomForestRegressor |
|-------|------------------|---------------|-------------|-----------------------|
| MAE | 173,45 | 156,22 | 160,24 | 164,49 |
| MAPE | 0,33 | 0,26 | 0,25 | 0,31 |
| RMSE | 277,87 | 292,15 | 300,33 | 269,07 |
| RMSLE | 0,39 | 0,39 | 0,42 | 0,37 |
| RMSPE | 0,40 | 0,32 | 0,34 | 0,40 |

| | LinearRegression | LSTM | MLP | RandomForestRegressor |
|-------|------------------|---------------|--------|-----------------------|
| MAE | 267,20 | 148,42 | 177,89 | 163,17 |
| MAPE | 0,56 | 0,22 | 0,34 | 0,31 |
| RMSE | 420,72 | 301,33 | 282,98 | 269,86 |
| RMSLE | 1,27 | 0,41 | 0,47 | 0,37 |
| RMSPE | 0,89 | 0,29 | 0,44 | 0,40 |

Univariate to Multivariate

➔ Best results with **multivariate** forecasting with **RandomForestRegressor** and **LSTMs**



Best Results from Analyses (APP1, RF, multivariate + multi-step)

Results

Long-term prediction (multi-step forecast) increasingly inaccurate

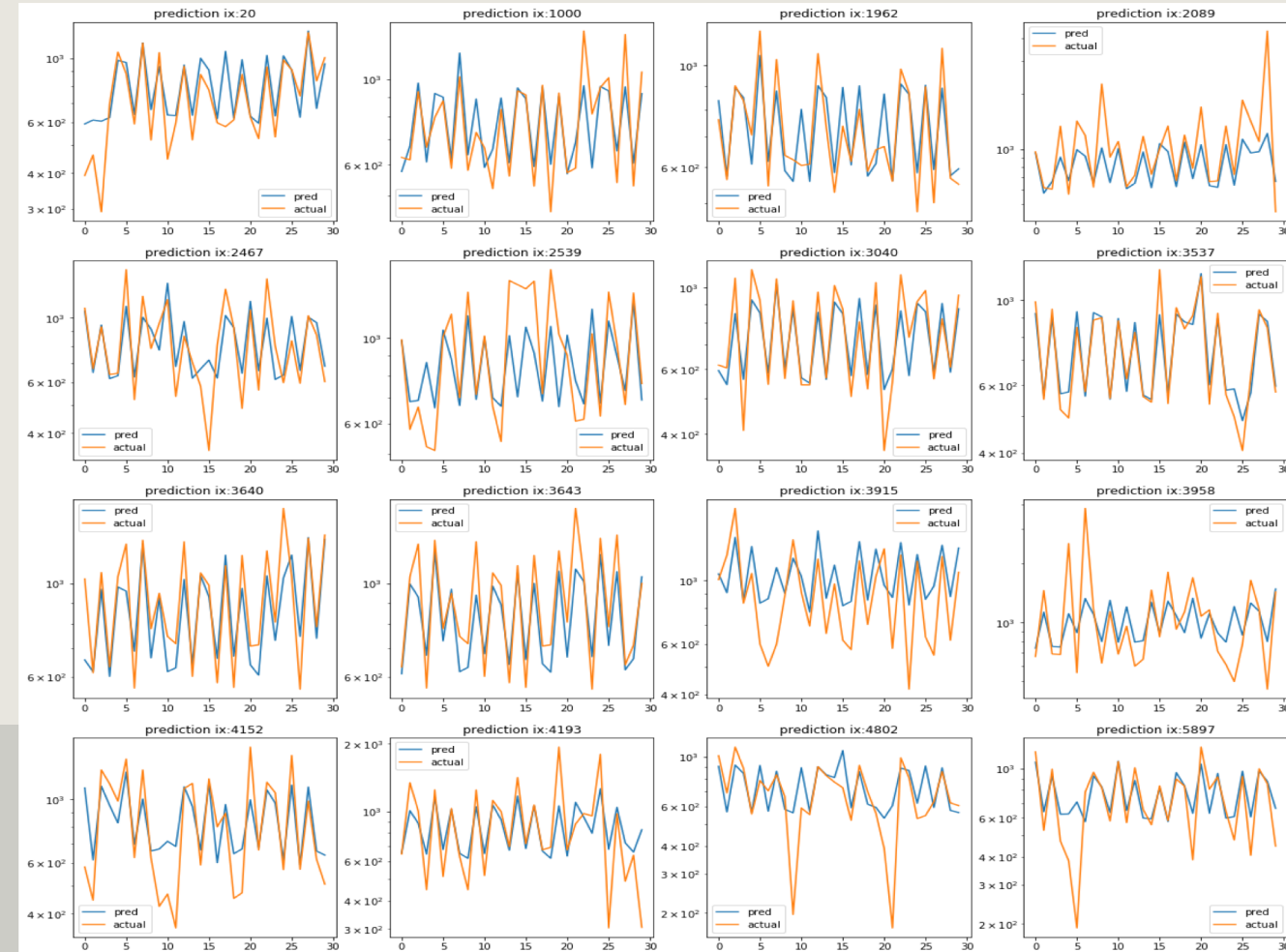
Difficult to model randomness and noise

Extreme values are hard to predict, but most important

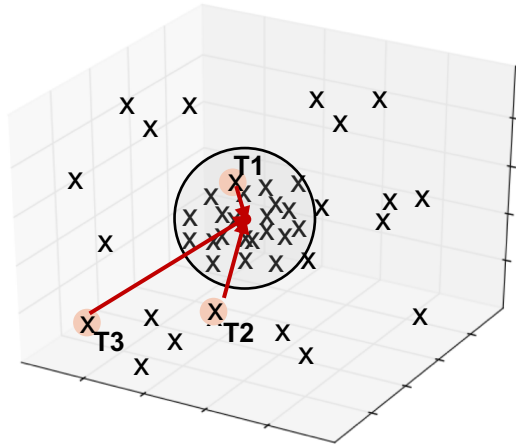
Peaks of response time do not necessarily reflect bad system state/system outage



Move towards anomaly detection for abnormal system behavior



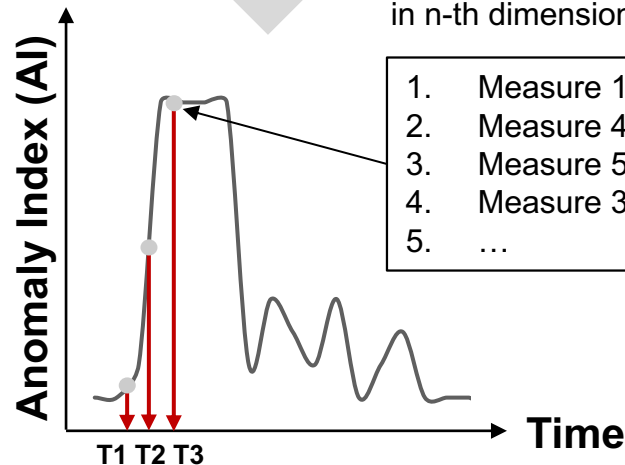
Density-based Clustering (DBSCAN)



Euclidean Distance

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Indicator for root cause by distance in n-th dimension



Input

Standardized Multivariate Feature Vector X

$$X = \begin{pmatrix} \text{Avg. Response Time} \\ \text{CPU util. (\%)} \\ \text{Amount Requests} \end{pmatrix}$$

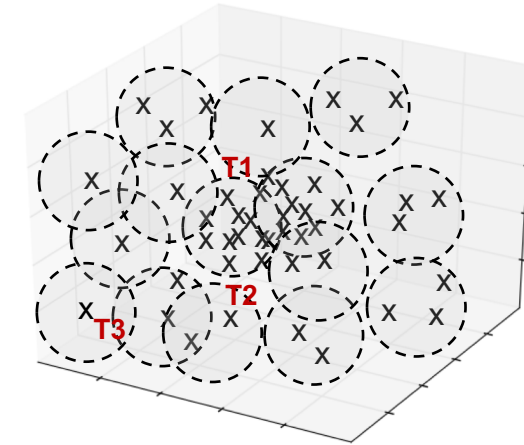
Output

Continuous measure reflecting the degree of abnormality

DBSCAN
 $y = \text{Anomaly Index (AI)}$
 + Indicator for root cause

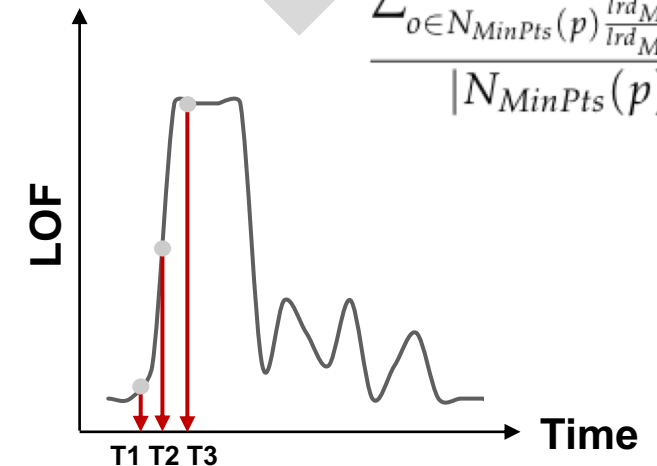
LOF
 $y = \text{Local Outlier Factor (LOF)}$

Local Outlier Factor (LOF)



(Local) Outlier Factor

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$



Anomaly Detection – Software Labelling Solution

Display **real-time** historical and predicted **system health metric (i.e. anomaly index)**

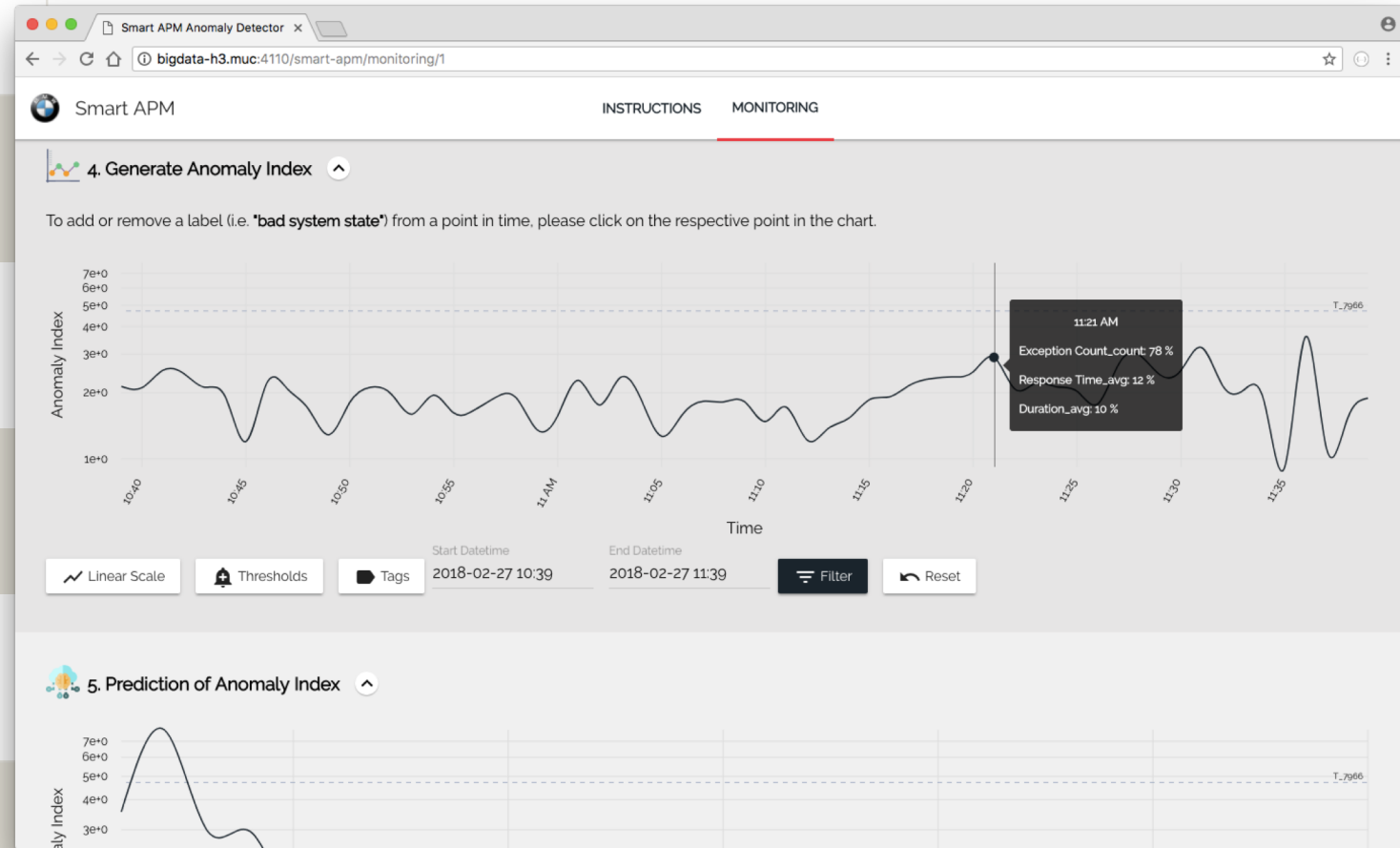
Scalability for fast model training on large datasets

Manual and automatic **threshold management**

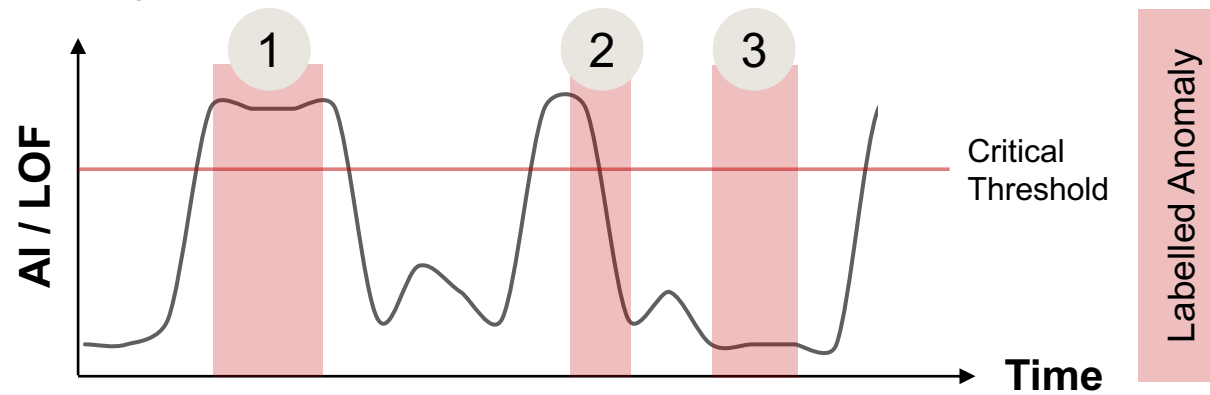
Automatic triggering of **notifications** and **alerts**

Root cause analysis through **drilling down by monitor** (i.e. APM measure)

User input for **ground truth labelling** of system status



Anomaly Detection – Evaluation



Types of Detection

- 1 Completely detected
- 2 Partly detected
- 3 Not detected

8 labelled time periods of abnormal system behavior in 2 months for 1 enterprise application

Simple Recall

True Positives (TP) = 1 + 2

False Negatives (FN) = 3

$$\text{Recall} = \frac{TP}{TP+FN} = \frac{6}{8} = 0.75$$

Problem: Imbalanced durations of anomalies and False Positives (FP)

Point Anomalies

DBSCAN

Precision = 0.11

Recall = 0.04

AUC = 0.73

LOF

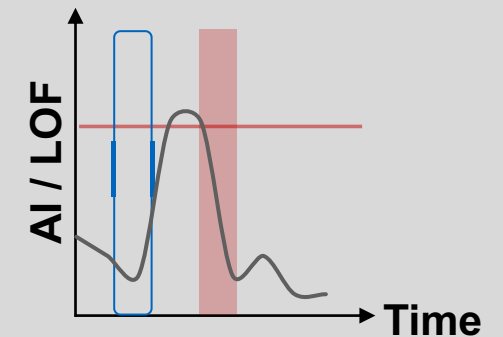
Precision = 0.06

Recall = 0.02

AUC = 0.70

Problem: Volatile APM measures demand collective anomaly detection

Sliding Window



Problem: Bias induced through interpretation of partly detected anomalies




Results

DBSCAN has better **AUC**, which is de facto standard for evaluation of anomaly detection*

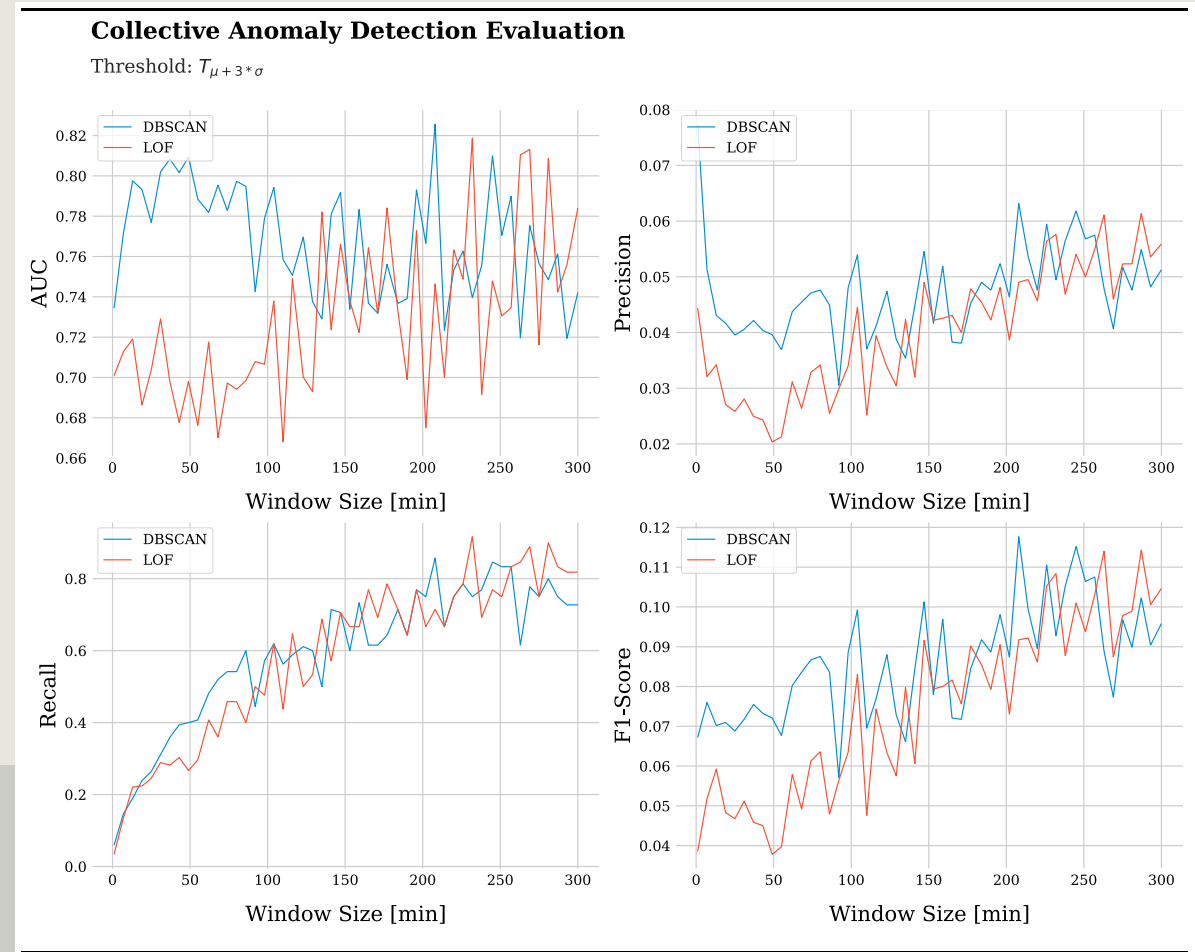
LOF yields better **Precision/Recall**

Careful evaluation necessary due to induced **biases** and **shortcomings** of different **evaluation techniques**

Software solution received very **good feedback**; **transferability** to other time series problems

 **DBSCAN** at least **equivalently useful** as **LOF**, but additionally provides **indication for root cause**

Results Sliding Window Evaluation (Window-Sizes in [1,300])



* According to Goldstein & Uchida (2016)





Cand. M. Sc.

Daniel Elsner

Technische Universität München
Faculty of Informatics
Chair of Software Engineering for Business
Information Systems

Boltzmannstraße 3
85748 Garching bei München

Tel +49.89.289.17135

Daniel.elsner@tum.de
www.matthes.in.tum.de



Bibliography

M. Goldstein and S. Uchida. “A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data.” In: PLoS ONE 11.4 (2016)

J. Webster and R. T. Watson. “Analyzing the Past to Prepare for the Future: Writing a Literature Review.” In: MIS Quarterly 26.2 (2002), pp. xiii –xx

Identified Measures

- Path-based information
 - Transaction count → Count of monitored transactions/minute
 - Path response time (max) → Response time of paths/minute (avg, sum, min, max)
 - Path duration → Path duration of paths/minute (avg, sum, min, max)
 - Path node count → Nodes visited in paths/minute (avg, sum, min, max)
- Middleware (web server)
 - Request/minute → Amount of requests/minute
 - Traffic/minute → Amount of network traffic in KBs/minute
- Application server
 - JVM heap → JMX measure
 - JVM open files → JMX measure
 - JVM threads → JMX measure
- Database
 - Reads/writes → I/O on DB
 - Sessions → Amount of DB sessions
- Server/Infrastructure
 - CPU percentage/load → CPU utilization of linux systems

Forecast

Response time (e.g. 30 min)

Approach

A collection of handwritten notes on sticky papers, organized into several conceptual frameworks. Each note typically includes a 'Problem' section, an 'Idea/Solution' section, and a 'RQ' (Research Question) section. The notes are scattered across a wooden surface and include various annotations and markers.

Concept: Service Clustering

Problem: Polluted architecture, redundant services, complexity of CA/application landscape

Idea/Solution: Service Clusters, "Micro-ops" = "micro-ops" = "micro-ops", Detect architectural flaws, Clear architectural plans, Similarity matrix

RQ: "Micro-ops" = "micro-ops" = "micro-ops"

Concept: Automated CI-Data

Problem: Ticketing overhead

Idea/Solution: Create tickets, identify patterns, Ticket similarity, New ticketing UI

RQ: Ticket similarity, New ticketing UI

Concept: Client Application Manager

Problem: User Experience Index (as ClientCall), Performance (...)

Idea/Solution: Small Tail/App Manager, Drive conditions from app activity data, Learnings about usage patterns (Histograms) by Network I/O

RQ: Drive conditions from app activity data

Concept: Performance pattern detection and root cause analysis model

Problem: ML model with only X predictors, Performance Degradation (response, throughput, throughput)

Idea/Solution: Root Cause Analysis (RCA) model with only X predictors, Predictors: CPU, I/O, Virtualization rate

RQ: Root Cause Analysis (RCA) model with only X predictors

Concept: Last scenario regression

Problem: Server throughput, Agility, network

Idea/Solution: Server load prediction, Regression model to predict load, Product econ. implications

RQ: Regression model to predict load

Concept: Path-relevant performance issues detection

Problem: Root cause analysis

Idea/Solution: Link env changes, Detect user behavior changes, Application/Service versioning

RQ: Link env changes, Detect user behavior changes

Concept: Timeseries-based Response Time Forecast

Problem: High response times, Application/Service crash

Idea/Solution: Timeseries data to predict response time/availability, Abstracts from tech. or econ. interrelations

RQ: Timeseries data to predict response time/availability

Concept: Performance learning on ticketing events

Problem: Ticketing overhead, Complexity in Configuration

Idea/Solution: From ticketing data and ticket transaction reevaluation, Red-e automatically created tickets

RQ: Red-e automatically created tickets

Concept: Resource Scale Recommendations

Problem: Inefficient resource allocation

Idea/Solution: Recommend vertical/horizontal scaling, Compare resource utilization scenarios, Reinforcement Learning by affinity configs

RQ: Compare resource utilization scenarios

Concept: Method significance model for performance pattern detection analysis

Problem: Root Cause analysis, Test scenario repeat manual efforts by test engineers

Idea/Solution: Classify methods which contribute to bottlenecks, Cluster input data by "good" and "bad" test cases, Significance of (resource consumption)

RQ: Classify methods which contribute to bottlenecks

Other Notes:

- Availability Client Monitoring:** Response time, Activity, Timeout, Device
- Performance:** High resource usage, High response times, User experience index
- Maintenance/Management:** Root cause analysis, Method architecture, Complexity in Configuration
- Request Overload:** Mac (cache, RT, disk), Availability, Application service crash

Client

Frontend



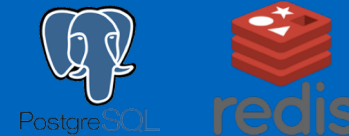
Backend

REST API



Persistence

Database



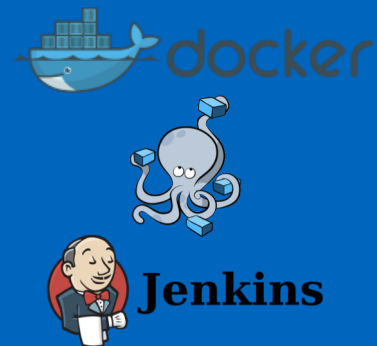
Data Sources

App Monitoring



Platform

Deployment



Worker Nodes



Time Series



Linux Monitoring



Task Scheduler



Model Storage



JVM Monitoring

