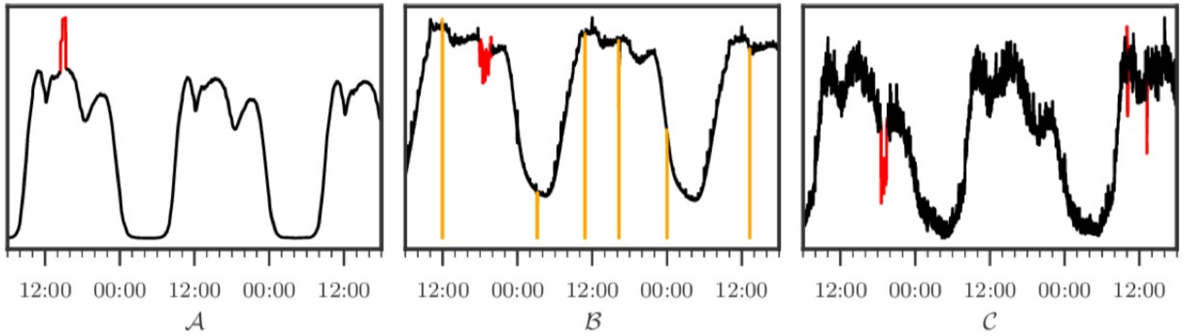
Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications

Haowen Xu[1] Wenxiao Chen[1] Nengwen Zhao[1] Zeyan Li[1] Jiahao Bu[1] Zhihan Li[1] Ying Liu[1]
Youjian Zhao[1] Dan Pei[1] Yang Feng[2] Jie Chen[2] Zhaogang Wang[2] Honglin Qiao[2]
[1]Tsinghua University
[2]Alibaba Group

WWW 2018

Problem Statement

- To detect anomalies on seasonal KPI time series.
  - *KPIs,* the *key performance indicators*, are real-valued system monitoring metrics.
  - $\mathbf{X} = (x_1, x_2, \ldots, x_T)$.
  - KPIs for web applications should reflect user activities, thus are seasonal.

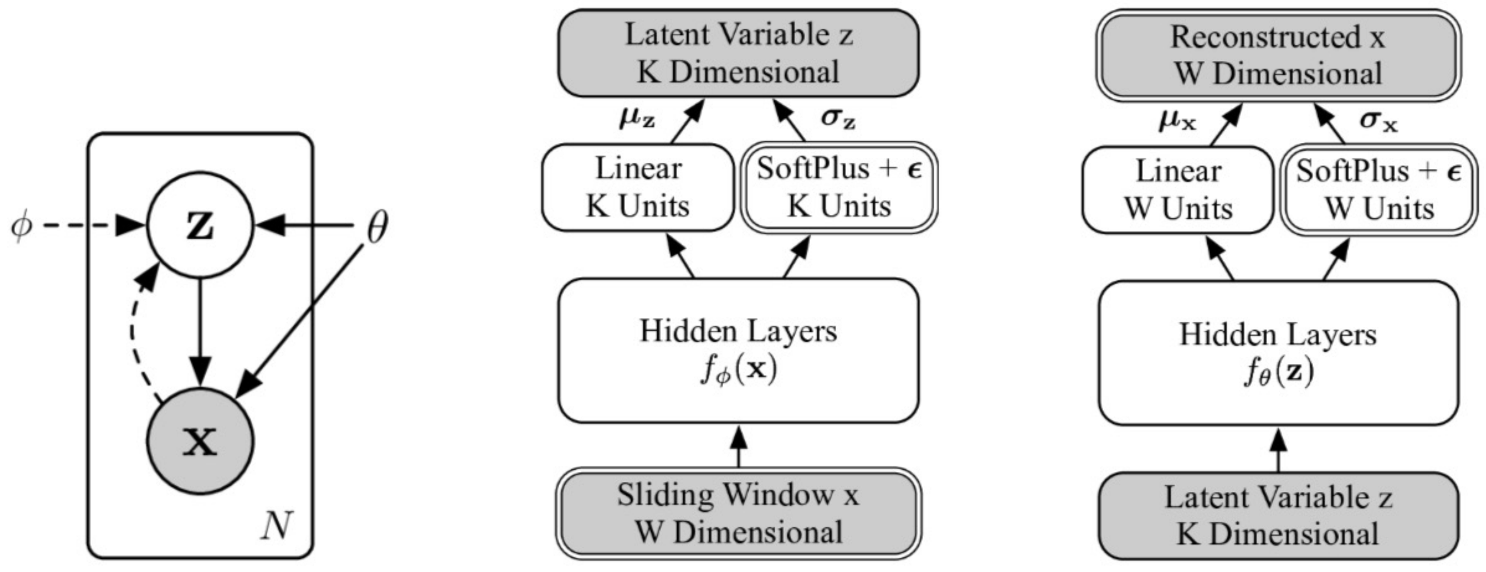


- For each time t, given a window of observations $\mathbf{x} = (x_{t-W+1}, \ldots, x_t)$, consisting of the on-time KPI observation x_t and historical observations of length $W-1$, compute an anomaly score s_t.
- The operators then decide whether to trigger an alert, based on this score.

Existing Methods

- Statistical
  - Anomaly detectors based on traditional statistical models [INFOCOM2012]
- Supervised
  - Supervised ensemble learning with above detectors – Opprentice[IMC2015], EGADS [KDD2015]

Network Structure

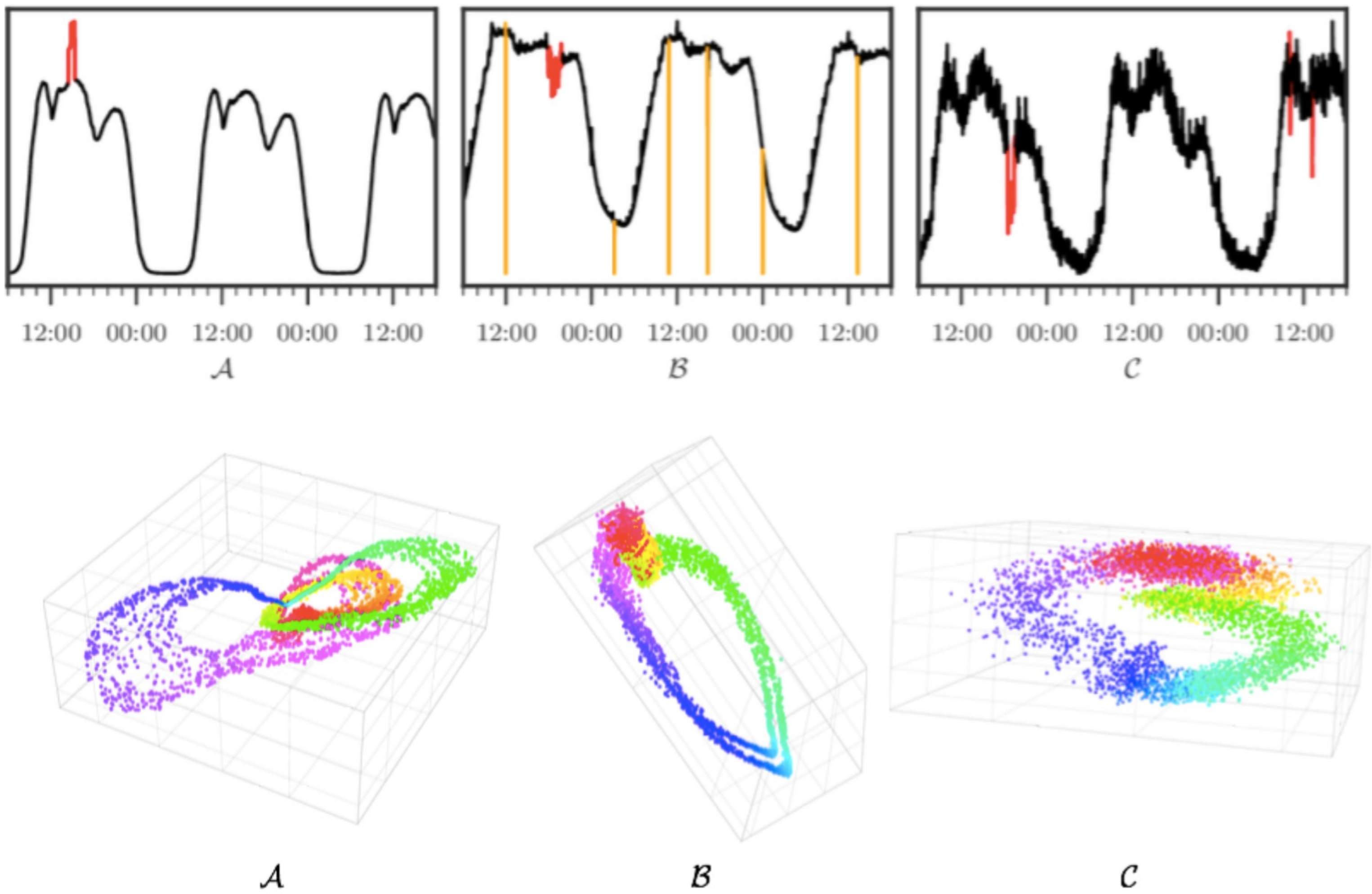


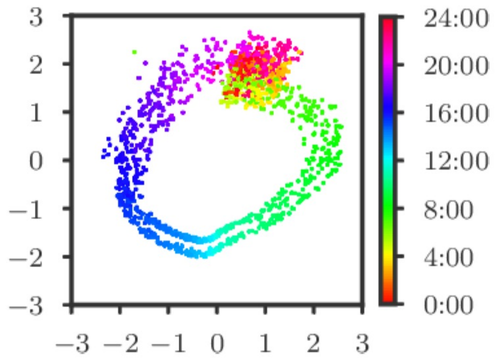
(a) VAE General Structure (b) $q_\phi(\mathbf{z}|\mathbf{x})$ of *Donut* (c) $p_\theta(\mathbf{x}|\mathbf{z})$ of *Donut*

- Variational net: $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{z}}, {\boldsymbol{\sigma}_{\mathbf{z}}}^2\mathbf{I})$.
- Generative net: $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$, $p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}, {\boldsymbol{\sigma}_{\mathbf{x}}}^2\mathbf{I})$.
- SoftPlus Trick: $\boldsymbol{\sigma}_{\mathbf{z}} = \mathrm{SoftPlus}[\mathbf{W}_{\boldsymbol{\sigma}_{\mathbf{z}}}^\top f_\phi(\mathbf{x}) + \mathbf{b}_{\boldsymbol{\sigma}_{\mathbf{z}}}] + \boldsymbol{\epsilon}$, $\mathrm{SoftPlus}[a] = \log[\exp(a) + 1]$. Similar for $\boldsymbol{\sigma}_{\mathbf{x}}$. (otherwise. unbounded)

$$\mathcal{L}_{vae} = \mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}\left[q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z})\right]\right]$$

3D Visualization of the Latent Space



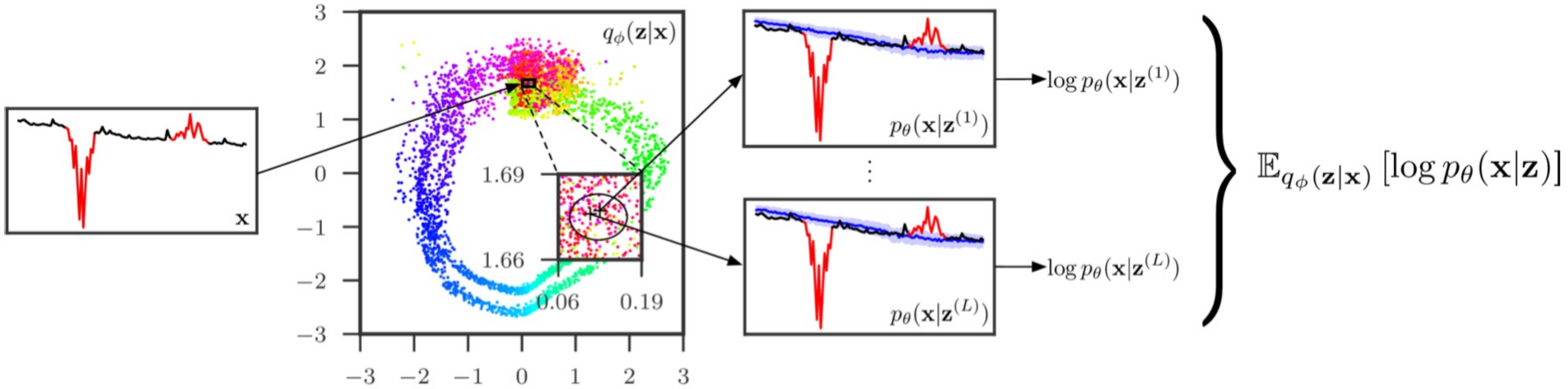


The Time Gradient

Figure 12: 3-d latent space of all three datasets.

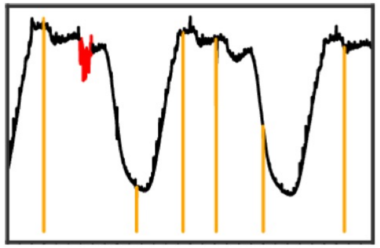
Detection via Donut: Overview

- The observation window $\mathbf{x}$ of time t is fed into VAE.
- L samples of $\mathbf{z}$ is taken from $q_\emptyset(\mathbf{z}|\mathbf{x})$, namely, $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(L)}$.
- For each $\mathbf{z}^{(l)}$, calculate $\log p_\theta\left(x_t|\mathbf{z}^{(l)}\right)$, the element-wise log-likelihood at time t.
  Note $\log p_\theta\left(\mathbf{x}|\mathbf{z}^{(l)}\right) = \sum_{i=t-W+1}^{t} \log p_\theta\left(x_i|\mathbf{z}^{(l)}\right)$.
- The anomaly score $s_t = \frac{1}{L}\sum_{l=1}^{L} \log p_\theta\left(x_t|\mathbf{z}^{(l)}\right)$.
- s_t is the Monte Carlo approximation of the element-wise *reconstruction loss* $\mathbb{E}_{q_\emptyset(\mathbf{z}|\mathbf{x})}[\log p_\theta(x_t|\mathbf{z})]$.



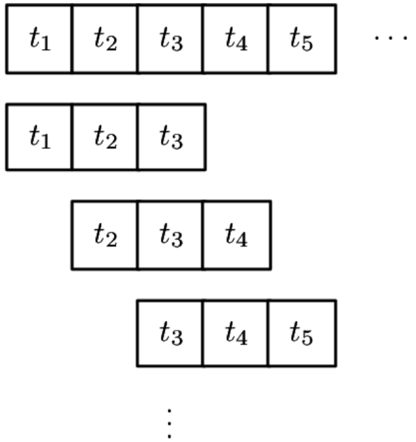
Data Preprocessing

- *Fill missing points with zeros*.



  - Missing points may exist on a KPI series.
  - We fill missing points with zeros, and record the positions of these missing points by an auxiliary binary series $\mathbf{Y} = (y_t)$, where $y_t = 1$ indicates time t is a missing point.

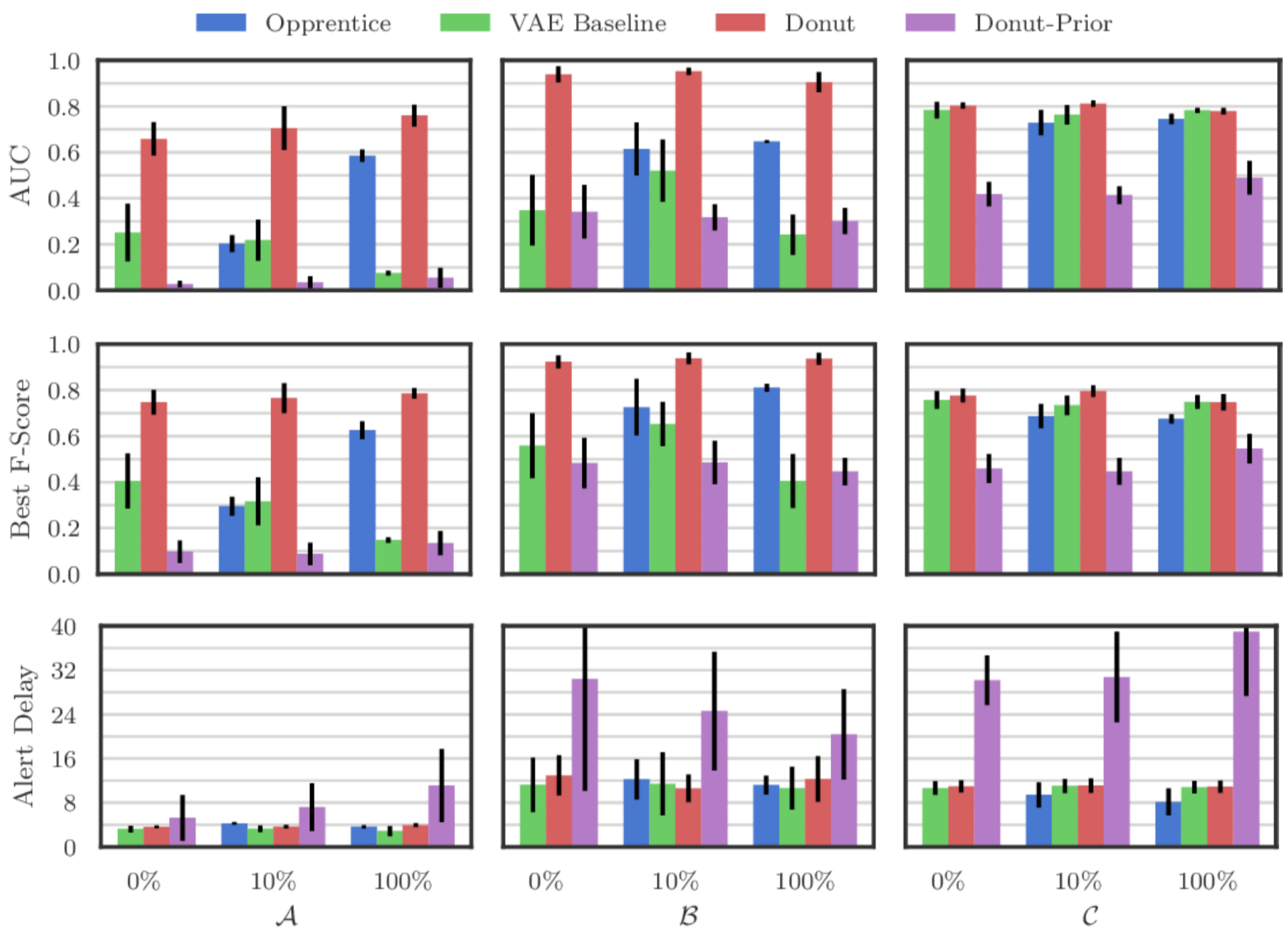
- *Standardization*: $\widehat{x_t} = (x_t - \mu)/\sigma$.
- *Sliding window*: each window $\mathbf{x} = (x_{t-W+1}, \ldots, x_t)$ has length W.



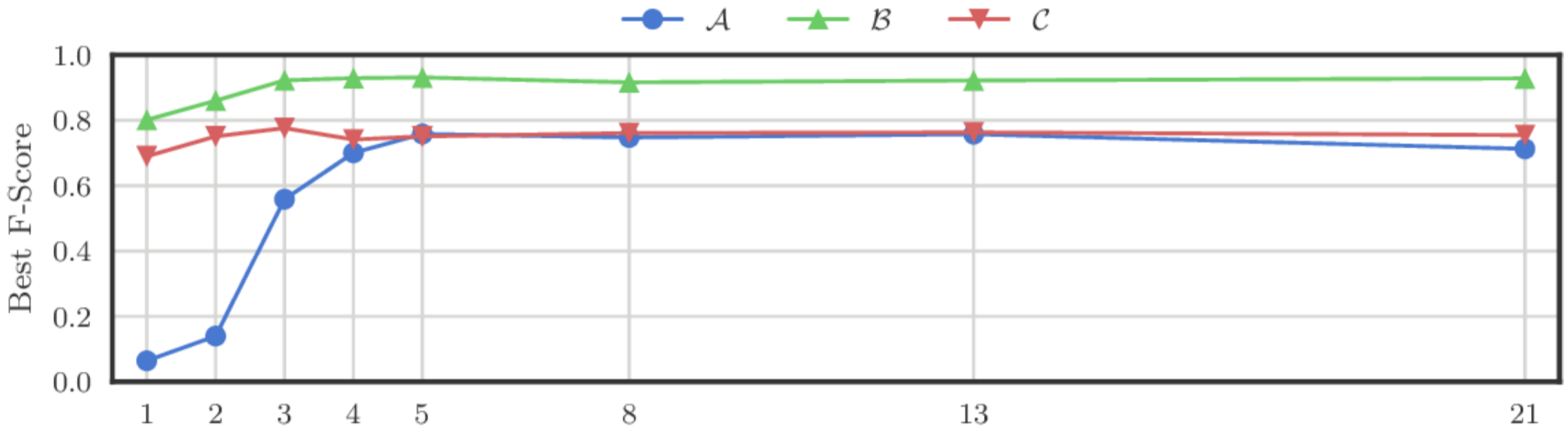
Training

- M-ELBO: modify the original VAE objective, to remove the wrong reconstruction loss caused by the missing points in training objective. For each window $\mathbf{x} = (x_{t-W+1}, \dots, x_t)$,
$$\tilde{\mathcal{L}}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{i=t-W+1}^{t} (1-y_i)\log p_\theta(x_i|\mathbf{z}) + (1-\gamma)\log p_\lambda(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})\right]$$
where $\gamma = \frac{\sum_{i=t-W+1}^{t} y_i}{W}$, indicating the ratio of abnormal points in the window $\mathbf{x}$.
- Missing Data Injection: randomly set 1% points to be missing at every epoch, such that the model should be more robust to true missing points in test data.

Overall Performance



Dimension Reduction is required



Window size is 120, while the best $\mathbf{z}$ dimensionality is no larger than 10.

Conclusion

- Key points of this paper:
  - Use *reconstruction probability* to detect anomalies.
  - Use *dimension reduction*, *M-ELBO*, *missing data injection* and *MCMC imputation*.
  - The *KDE interpretation* and related analysis.
- Donut source code: https://github.com/haowen-xu/donut

Robust and Rapid Clustering of KPIs for Large-Scale Anomaly Detection

Zhihan Li[1], Youjian Zhao[1], Rong Liu[2], Dan Pei[1]

清華大學 [1]
Tsinghua University

STEVENS INSTITUTE of TECHNOLOGY [2]
THE INNOVATION UNIVERSITY®

Outline

- Background
- Algorithm
- Evaluation
- Clustering for KPI Anomaly Detection
- Conclusion

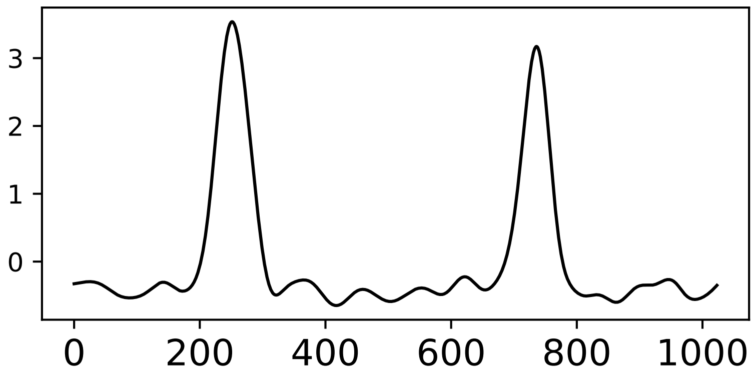
Outline

- Background
- Algorithm
- Evaluation
- Clustering for KPI Anomaly Detection
- Conclusion

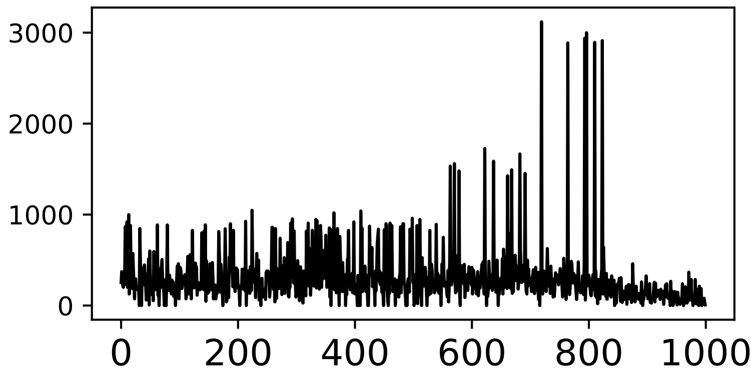
Problem Scenario: KPIs in Internet companies

- Large Internet companies monitor a large number of **KPIs** (**Key Performance Indicators**, *e.g.,* CPU utilization, # of queries per second) to ensure the service quality and reliability.

- KPIs are **time series data**. **Anomalies** on KPIs (*e.g.*, a spike or dip) often indicate potential failures on relevant applications, such as server failures, network overload, *etc*.



Idealized time series data



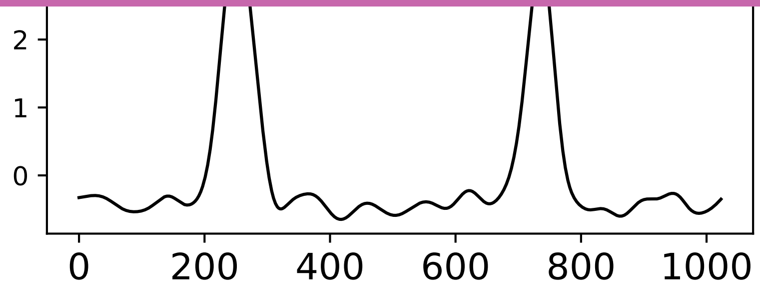
Real-world KPI data (raw)

Problem Scenario: KPIs in Internet companies

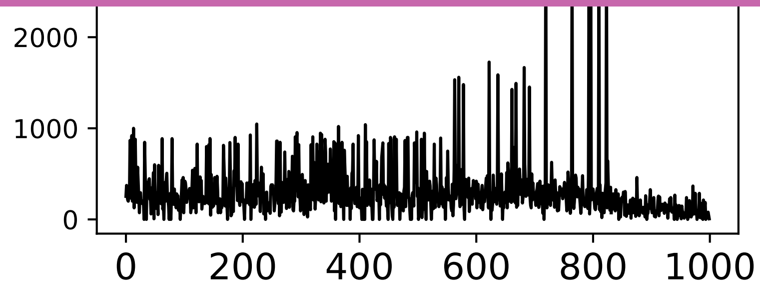
- Large Internet companies monitor a large number of **KPIs** (**Key Performance Indicators**, *e.g.,* CPU utilization, # of queries per second) to ensure the service quality and reliability.

- KPIs are **time series data**. **Anomalies** on KPIs (*e.g.*, a spike or dip) often indicate potential failures on relevant applications, such as server failures,

Use **Anomaly Detection** techniques to detect anomalous events timely!



Idealized time series data



Real-world KPI data (raw)

Problem Scenario: Large-Scale KPI Anomaly Detection

- Most anomaly detection algorithms (*e.g.*, Opprentice[1], DONUT[2]) assume that an **individual model** is needed for **each KPI**.

- **Large-scale anomaly detection** is very challenging due to the large overhead of model selection, parameter tuning, model training or anomaly labeling.

- Many KPIs are **similar in underlying shape** due to their implicit associations and similarities.

- Identify **homogeneous KPIs** and apply **one** anomaly detection model per cluster.

Problem Scenario: Large-Scale KPI Anomaly Detection

- Most anomaly detection algorithms (*e.g.*, Opprentice[1], DONUT[2]) assume that an **individual model** is needed for **each KPI**.

- **Large-scale anomaly detection** is very challenging due to the large overhead of model selection, parameter tuning, model training or anomaly

KPI Clustering can help!

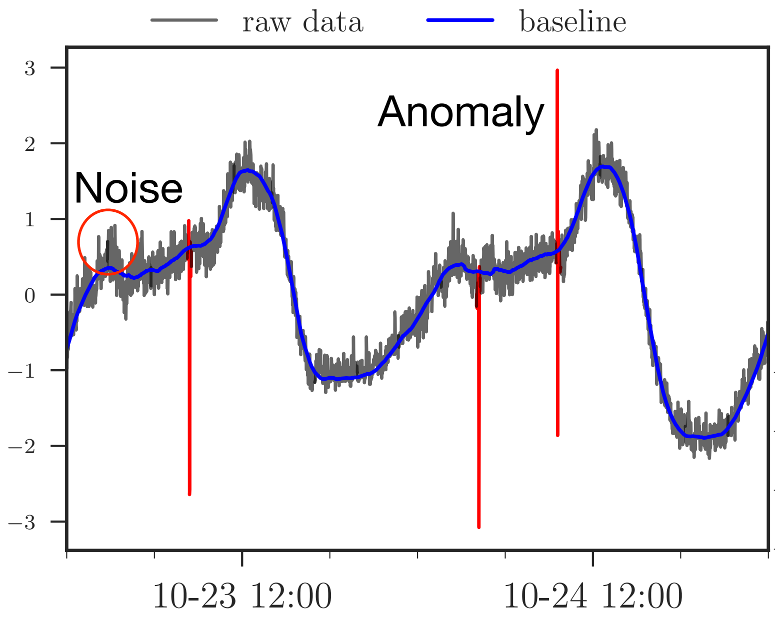
and similarities.

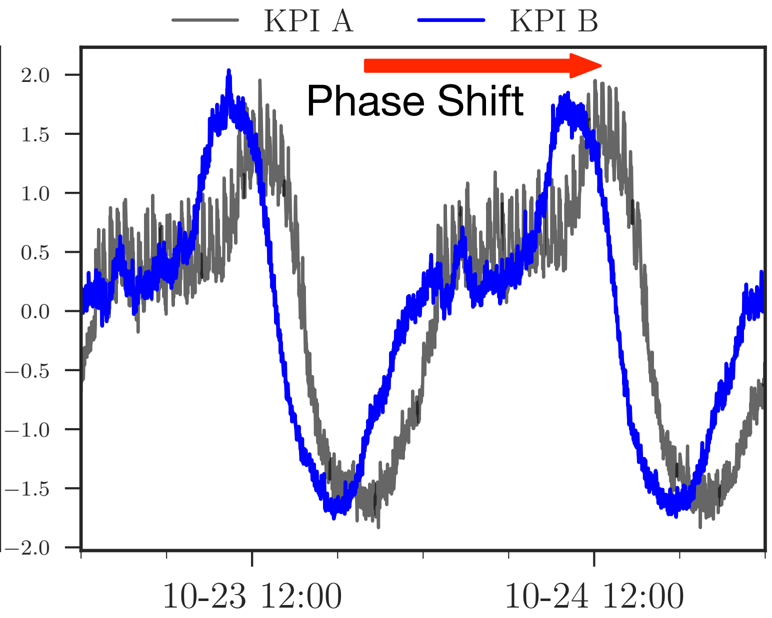
- Identify **homogeneous KPIs** and apply **one** anomaly detection model per cluster.

Major Challenges

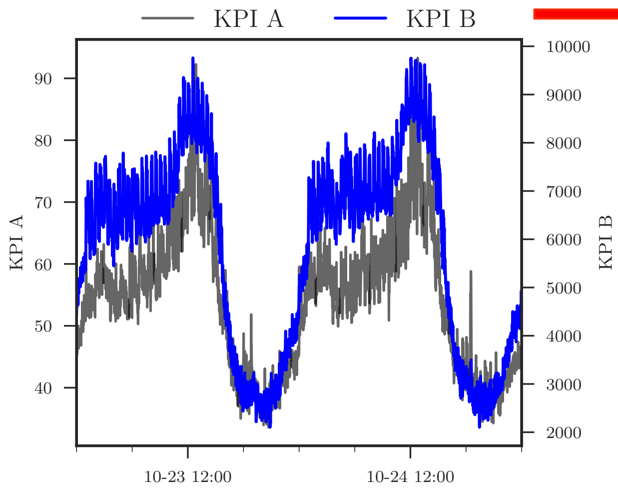
- **Shape Variations**
  - ➢ Anomalies
  - ➢ Noises
  - ➢ Phase Shifts
  - ➢ Amplitude Differences

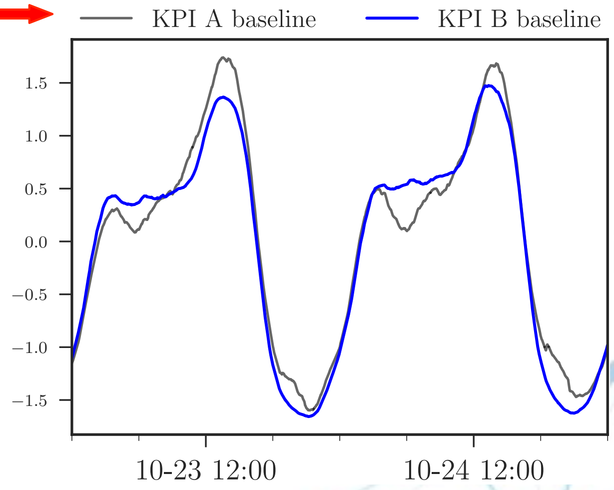
- **High Dimensional**





Standardization &
Baseline Extraction



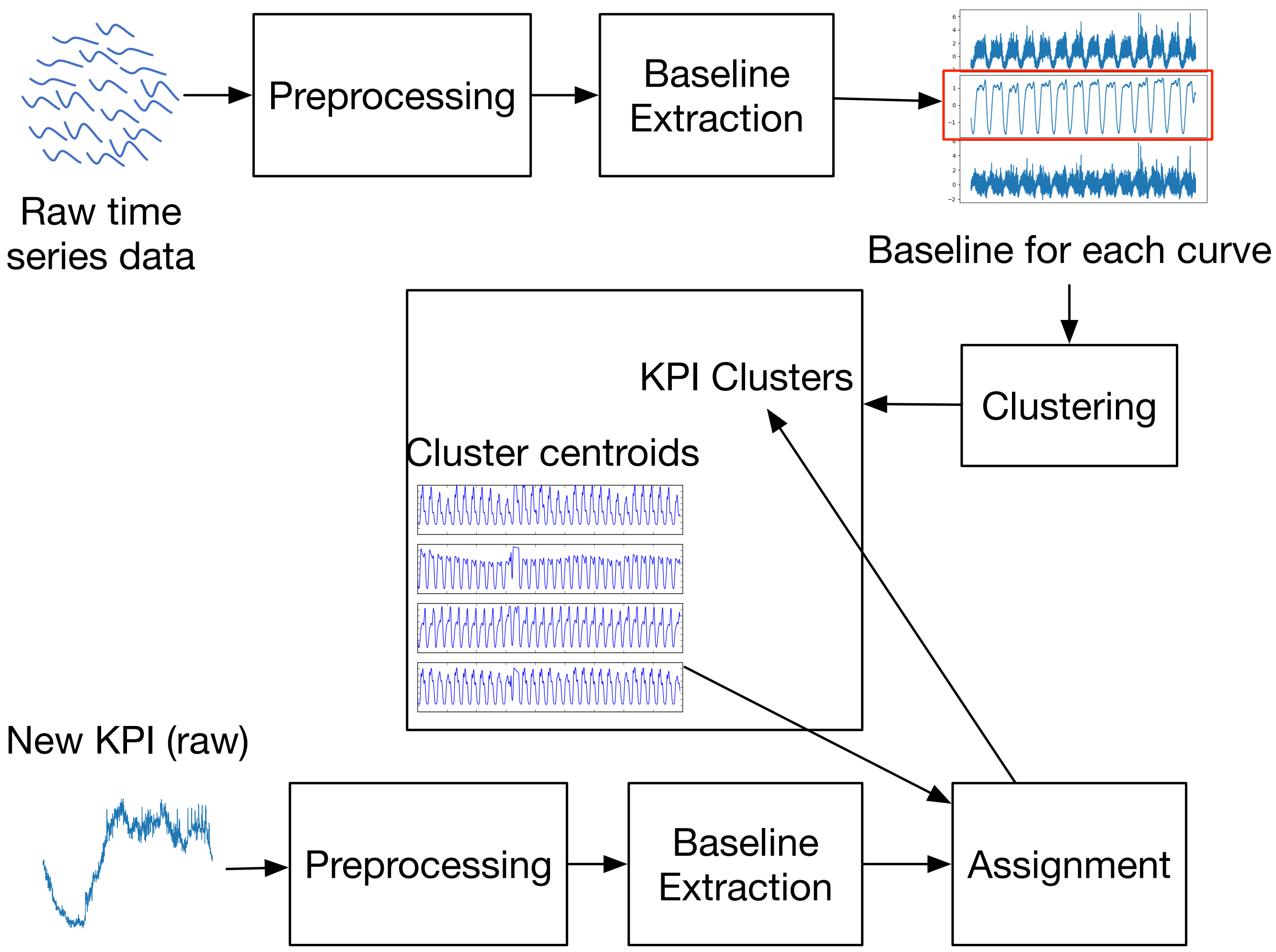


Amplitude
Difference

Outline

- Background
- Algorithm
- Evaluation
- Clustering for KPI Anomaly Detection
- Conclusion

Overall Framework of ROCKA



Preprocessing

- Fill missing values with linear interpolation

- **Standardization** (remove **amplitude differences**)

$$\widehat{x_t} = (x_t - \mu_x)/\sigma_x$$

x_t are the original KPI values, μ_x and σ_x are the mean and standard deviation of x_t.

Linear interpolation

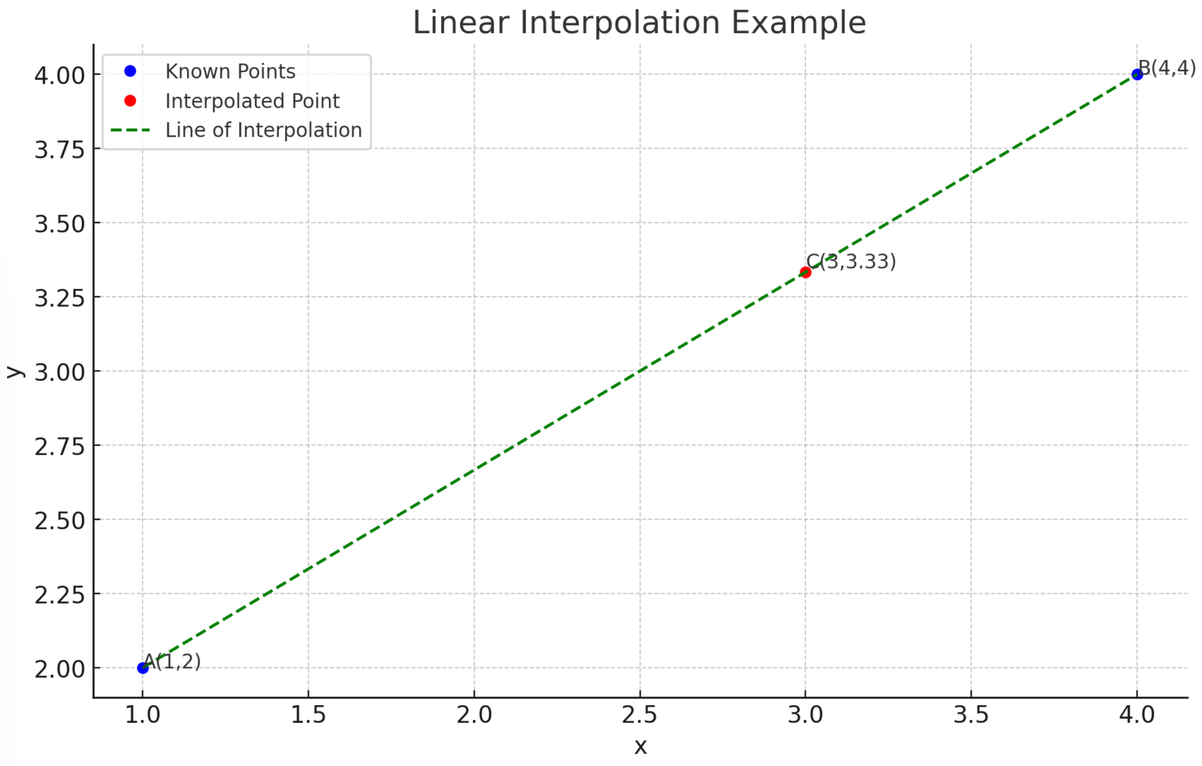
Linear interpolation is a method used to estimate or find a point within a known set of points on a graph or two known points on a line. It assumes that the rate of change between the points is linear, meaning the line connecting these points is straight.

Let's consider a simple example to illustrate linear interpolation. Suppose you have two points on a graph, $A(x_1, y_1)$ and $B(x_2, y_2)$, and you want to estimate the value of y at some point x that lies between x_1 and x_2.

The formula for linear interpolation between two points is given by:

$$y = y_1 + \frac{(x - x_1) \cdot (y_2 - y_1)}{x_2 - x_1}$$

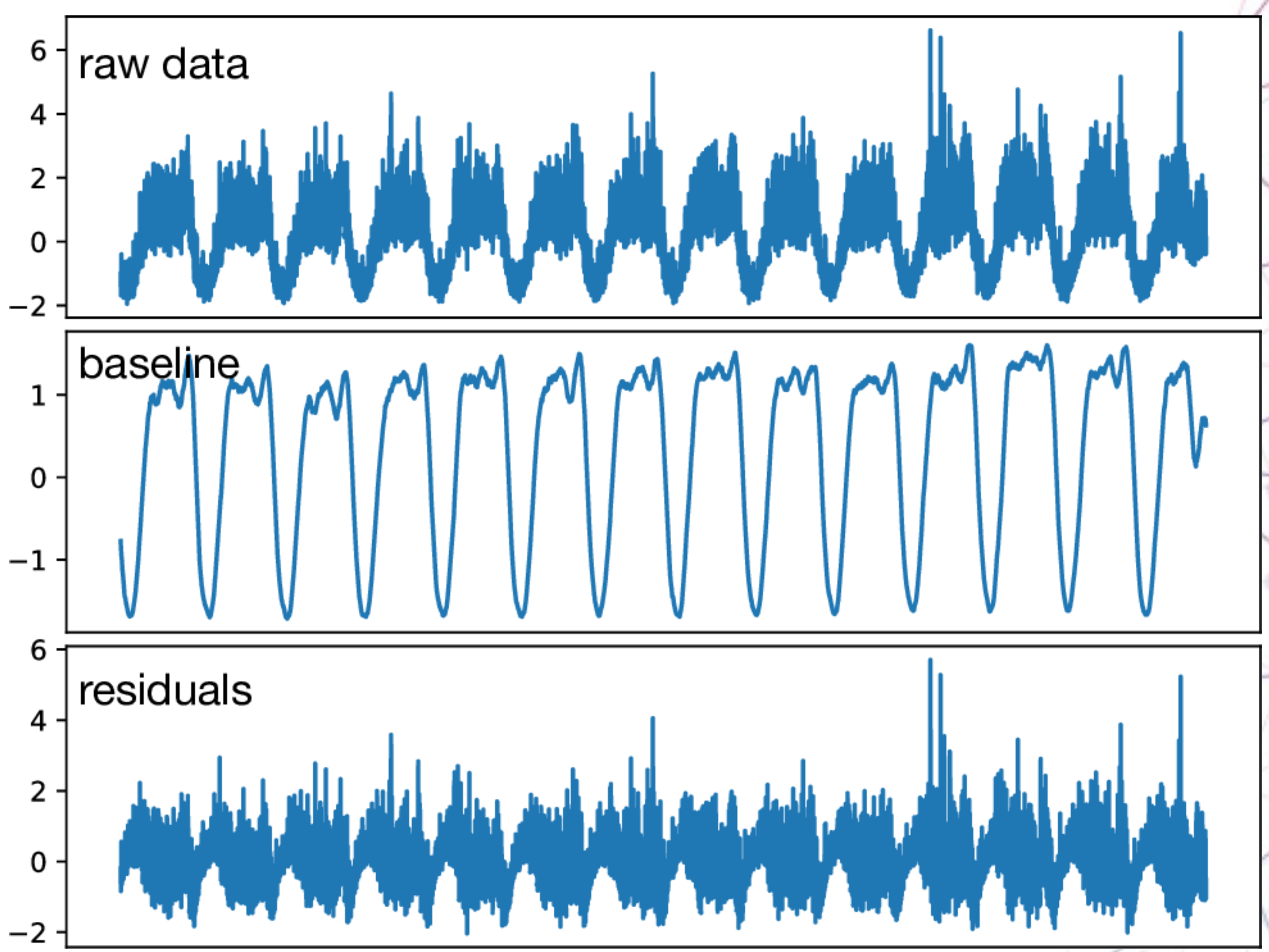
This formula essentially finds the slope between the two points (x_1, y_1) and (x_2, y_2) and then uses it to estimate the value of y at the point x.



Baseline Extraction

- Smoothing extreme value
  - Remove the top 5% data which deviates the most from the mean value.
  - Fill them using linear interpolation with their neighboring normal observations.

- Extract baseline
  - Apply moving average with a small sliding window.
  - Baseline extraction removes **anomalies and noises**, while preserving the underlying shape of KPIs.



Shape-based Similarity Measure

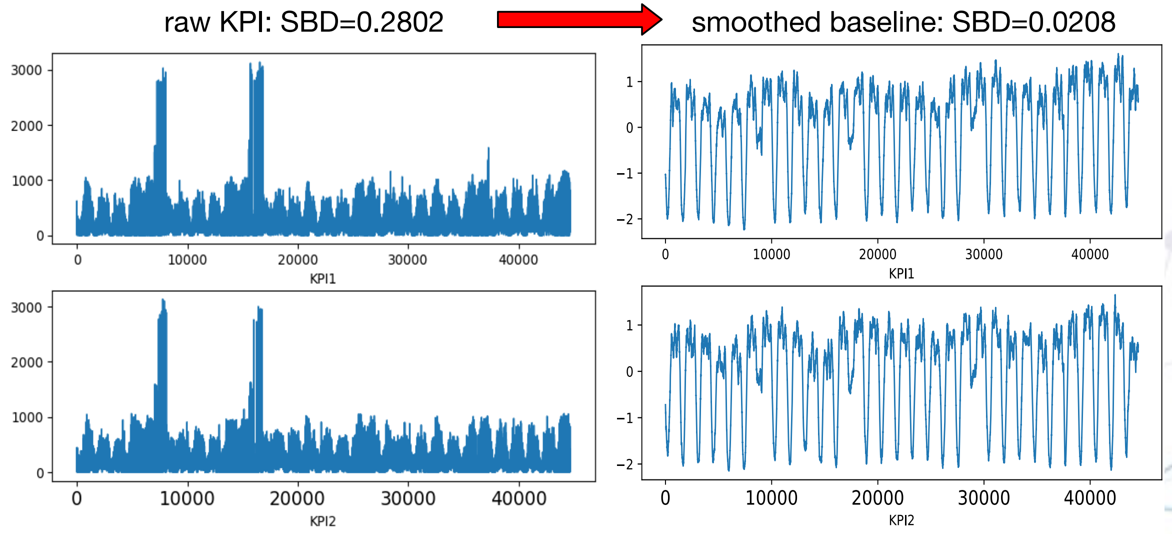
- Normalized version of cross-correlation (NCC) $\in [-1,1]$, robust to **phase shifts**.

$$NCC(\vec{x},\vec{y}) = \max_s\left(\frac{CC_s(\vec{x},\vec{y})}{\|\vec{x}\|_2 \cdot \|\vec{y}\|_2}\right)$$

- Shape-based distance (**SBD**[3]) $\in [0,2]$. Smaller SBD means higher shape similarity.

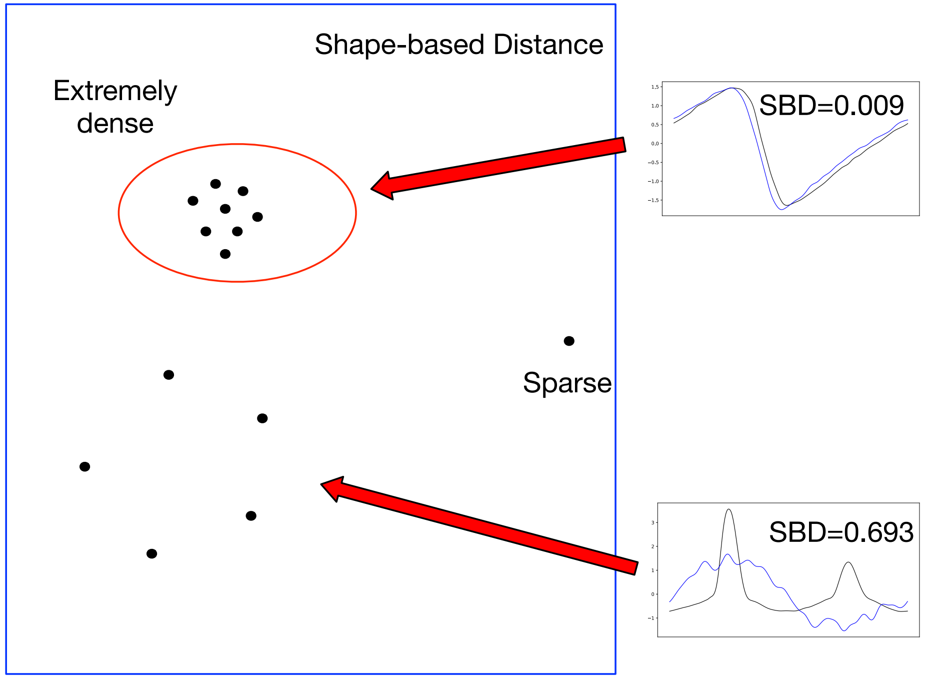
$$SBD(\vec{x},\vec{y}) = 1 - NCC(\vec{x},\vec{y})$$

Baseline extraction step plays an important role in finding the shape similarity between KPIs.

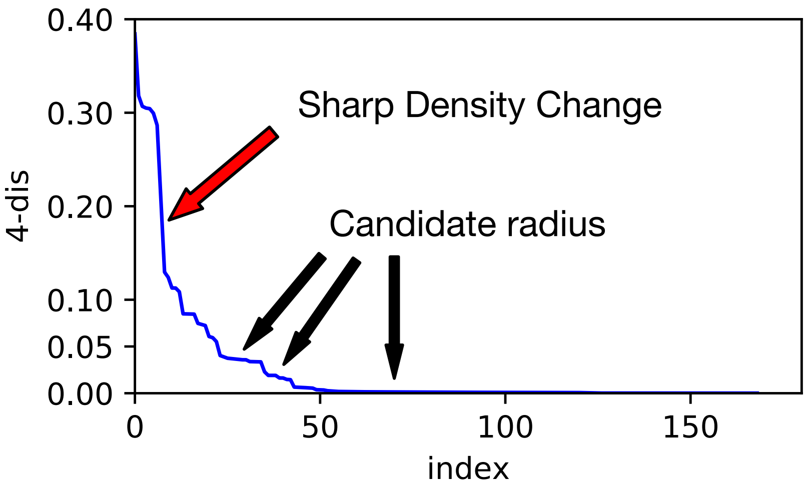


Density-based Clustering

- DBSCAN: find some cores in dense regions, and then expand the cores by transitivity of similarity to form clusters.



With SBD, extremely dense regions contain similar objects that form clusters, while large density radius indicates dissimilar objects.



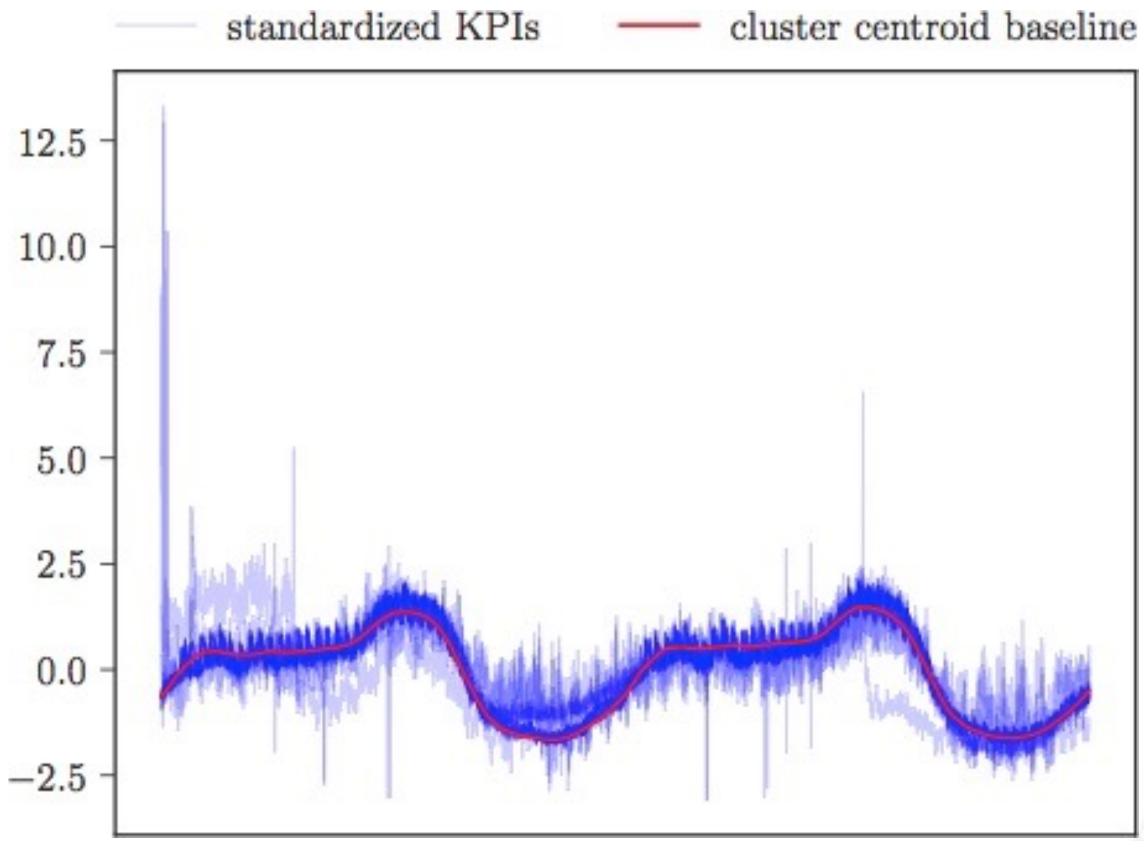
Flat parts on k-dis curve are regarded as candidate radiuses, while steep parts indicate sharp density changes.

Assignment

- Calculate the centroid of each cluster and assign the rest of KPIs based on centroids.

A cluster with 18 standardized KPIs and its centroid capturing the underlying shape of cluster.

$$\text{centroid} = \underset{\vec{x}\in \text{cluster}_i}{\arg\min} \sum_{\vec{y}\in \text{cluster}_i} SBD(\vec{x},\vec{y})^2$$

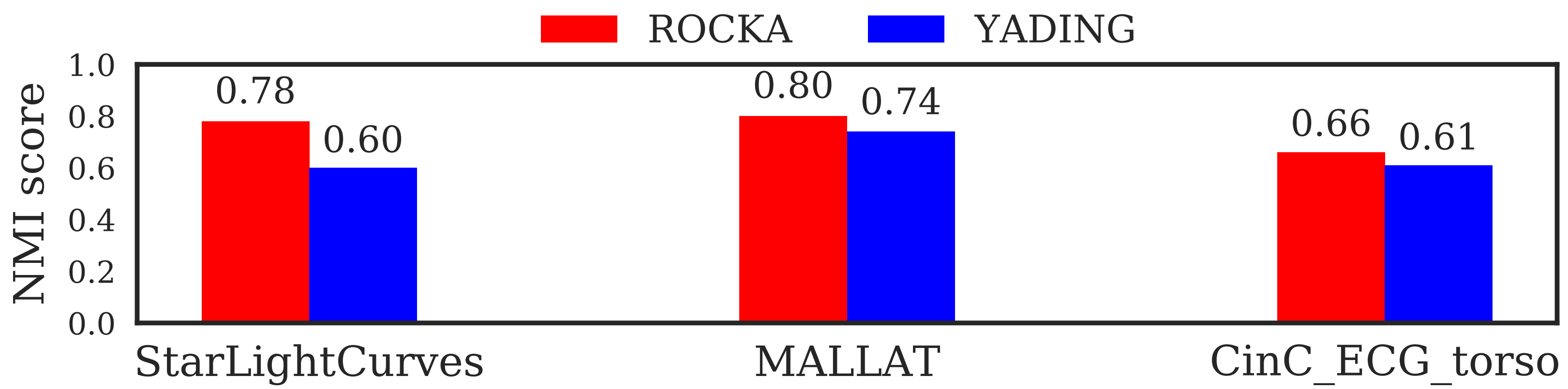


Outline

- Background
- Algorithm
- Evaluation
- Clustering for KPI Anomaly Detection
- Conclusion

Performance on public datasets

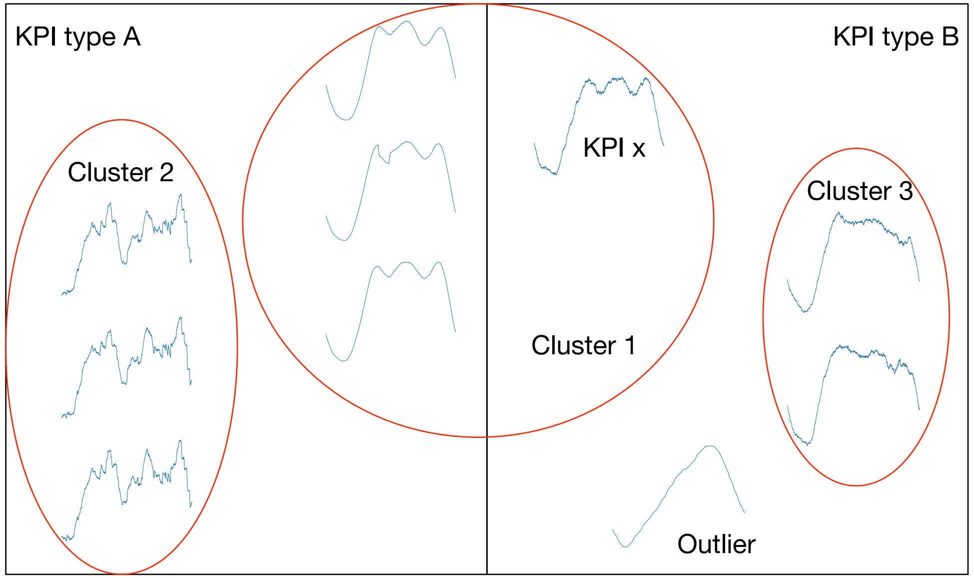
- YADING[4]: a state-of-the-art clustering algorithm for large-scale time series data.



About 1s for clustering, 0.05s to assign each KPI

Performance on real-world KPIs

- Evaluation metrics:

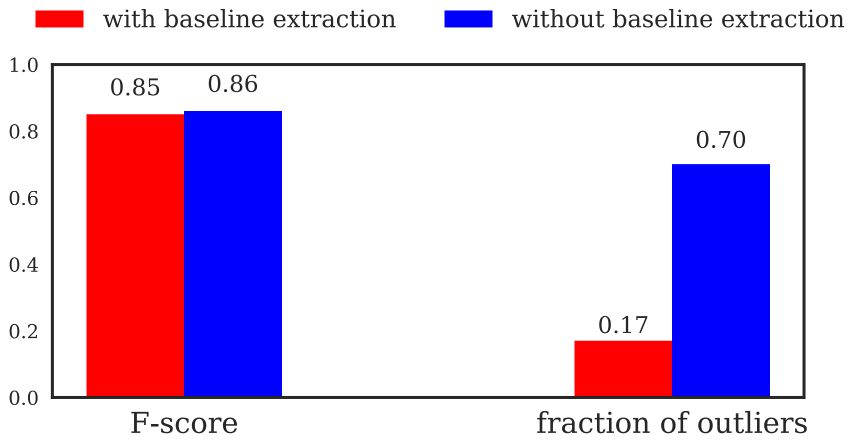


- Each curve is a baseline extracted from the raw KPI

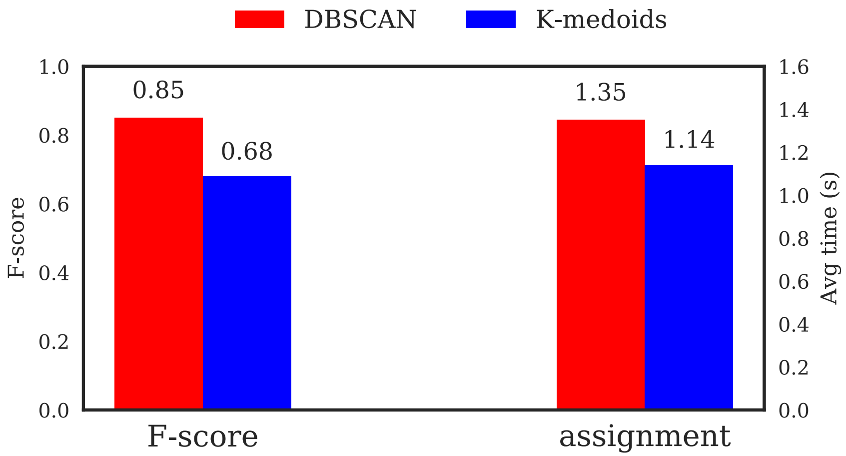
- Performance:

	DS1		DS2	
	ROCKA	YADING'	*ROCKA*	YADING'
F-score	1.00	0.98	0.85	0.99
fraction of outliers	0.04	0.18	0.17	0.49
# clusters	6	7	29	33
avg distance calculation (ms)	53	0.205	58	0.226
avg assignment time (ms)	411	54	1350	99

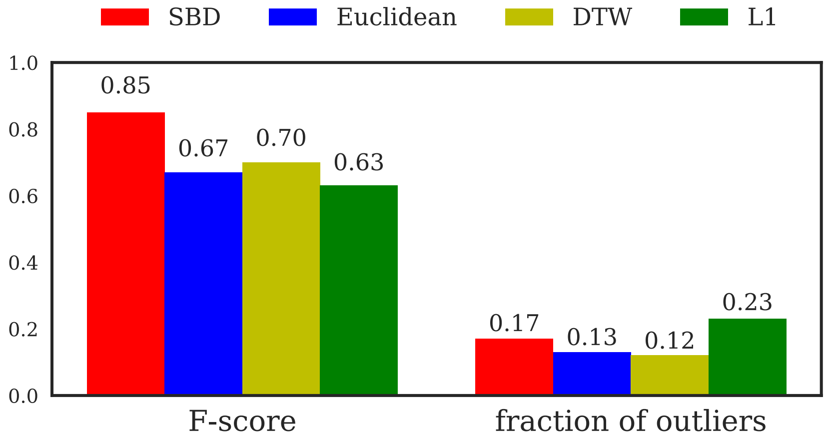
The effects of techniques in ROCKA



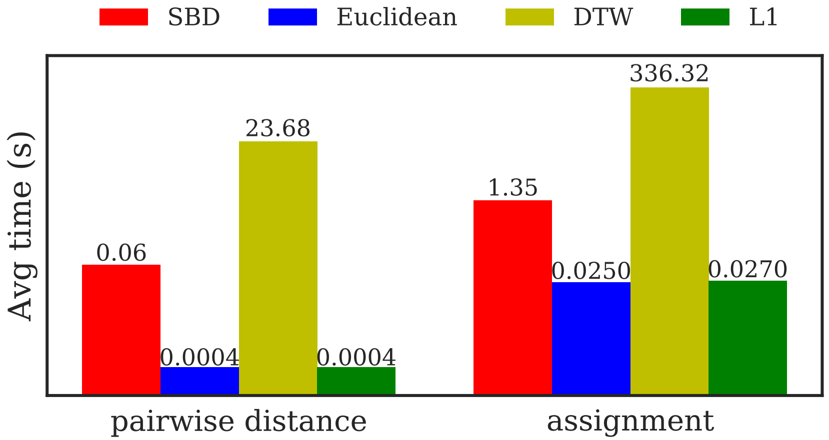
Performance with and without baseline extraction



Performance with different clustering methods



Performance with different similarity measures



Calculation time (log scale) with different similarity measures

Outline

- Background
- Algorithm
- Evaluation
- Clustering for KPI Anomaly Detection
- Conclusion

ROCKA for KPI Anomaly Detection

Prohibitive amount of **model training time**: Anomaly detection algorithms are often designed to have a model trained for each individual time series.

- ROCKA clusters KPIs similar in underlying shapes into a **cluster**.
- Train anomaly detection model on each cluster **centroid**.
- Directly use the **model** to detect anomalies on other KPIs in the **same cluster**.

Simplifying **threshold selection**: in some anomaly detection algorithms, a threshold needs to be fine-tuned by the ground-truth anomaly labels for optimal performance.

- The **threshold** selected for a cluster centroid can be used by other KPIs in the same cluster, reducing the overhead of parameter tuning and anomaly labeling.

Anomaly Detection Experiments Setup

- **DONUT**[2]: a state-of-the-art unsupervised anomaly detection algorithm for seasonal KPIs.

- **Dataset**: 48 6-month-long KPIs collected from different machines in a large Internet company. Experienced operators has labeled anomalies on these KPIs according to their domain knowledge to provide a ground truth for anomaly detection.

- **Experiments**:
  - E1: **DONUT only**. use DONUT to train an anomaly detection **model for each KPI** and fine-tune the **threshold for each KPI** for the best F-score.

  - E2: **ROCKA + DONUT**. First apply ROCKA on 48 KPIs to form **clusters**, then use DONUT to train an anomaly detection model **only on the centroid KPI** in each cluster, and select the best threshold according to the ground-truth labels **on the centroid**. The model and threshold are then used to detect anomalies in other KPIs of the same cluster.

  - E3: ***ROCKA* + DONUT + KPI-specific threshold**. Similar to E2, but reestimate the threshold for each KPI, except centroids, according to its ground-truth anomaly labels to get best performance.

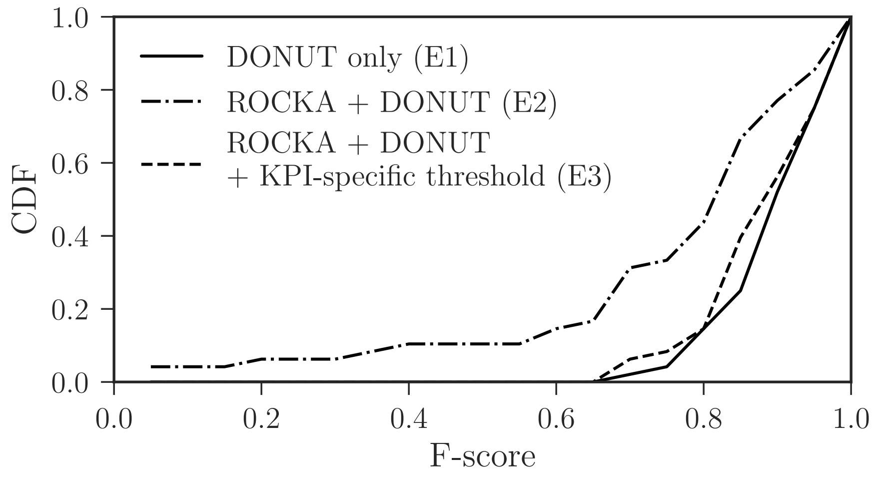
Anomaly Detection Performance

Cluster	E1	E2	E3	# KPIs
A	0.88	0.66	0.86	18
B	0.79	0.78	0.79	6
C	0.95	0.81	0.95	12
D	0.87	0.86	0.87	4
E	0.90	0.83	0.88	8
Overall	0.89	0.76	0.88	

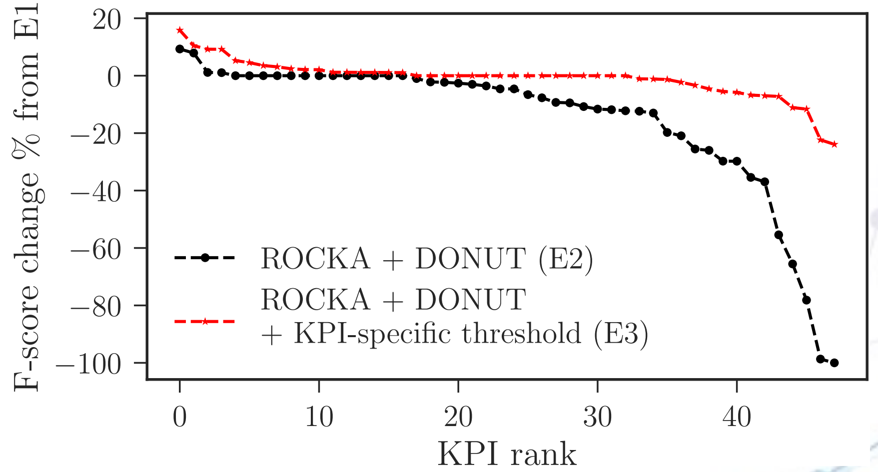
Average F-score for anomaly detection

algorithm	cluster	tot. train	avg. train	avg. test
DONUT only (E1)	—	51621	1075	345
ROCKA+DONUT (E2)	11	5145	1029	345
ROCKA+DONUT+KPI-specific threshold (E3)	11	5145	1029	345

Time Consumption for anomaly detection (seconds)



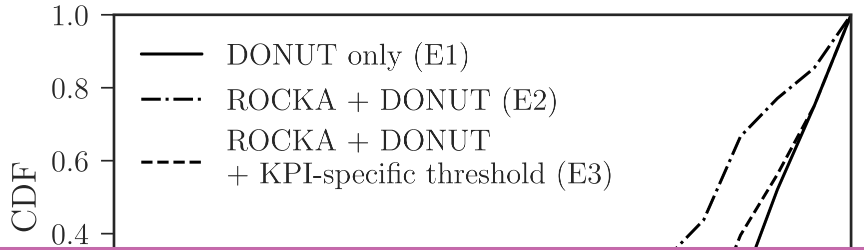
CDF of F-score on each KPI



The F- score change while using *ROCKA*+DONUT, compared to raw DONUT result (E1)

Anomaly Detection Performance

Cluster	E1	E2	E3	# KPIs
A	0.88	0.66	0.86	18

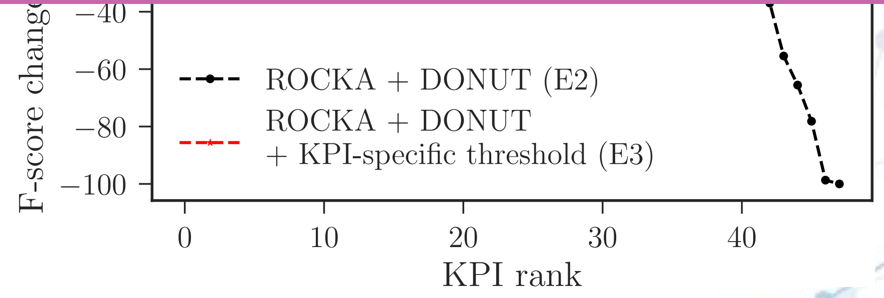


ROCKA reduces the model training time of DONUT by **90%**, with only **15%** performance loss.

When we share model but reestimate the threshold in each cluster, the F-score of most KPIs drop less than **5%** !

ROCKA+DONUT+KPI-specific threshold (E3)	11	5145	1029	345

Time Consumption for anomaly detection (seconds)



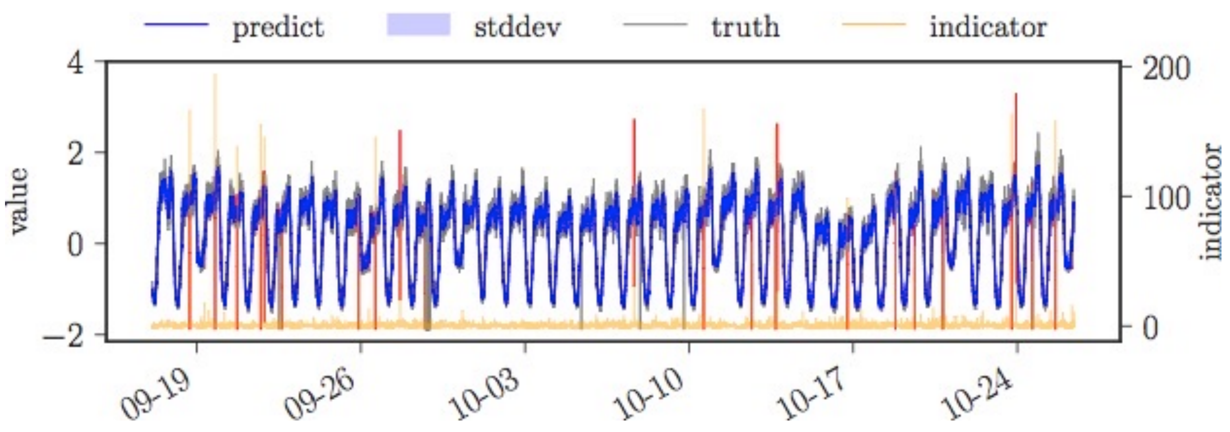
The F- score change while using *ROCKA*+DONUT, compared to raw DONUT result (E1)

Analysis of Results

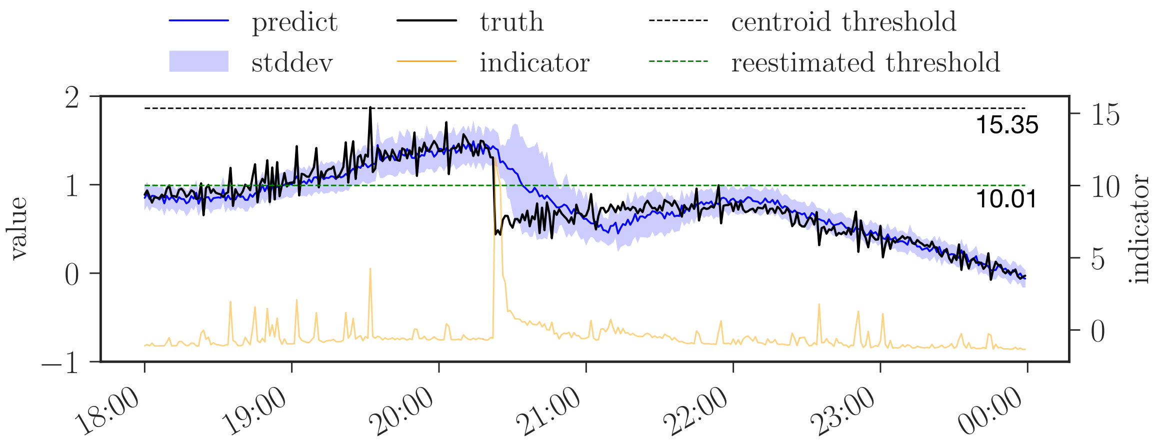
- KPIs with similar underlying shape tend to have implicit associations in practice (*e.g.*, belong to the same cluster of machines). In this way, KPIs in the same cluster also have similar normal patterns. As a result, they can **share an anomaly detection model and threshold**.

- KPIs may share the same anomaly detection model, but they can **vary by their anomaly severity levels**, and a uniform threshold cannot be the optimal for every KPI. This leads to some performance drop when directly applying centroid KPI’s model and threshold on other KPIs in the same cluster.

Analysis of Results



A's cluster centroid model (threshold=15.35)

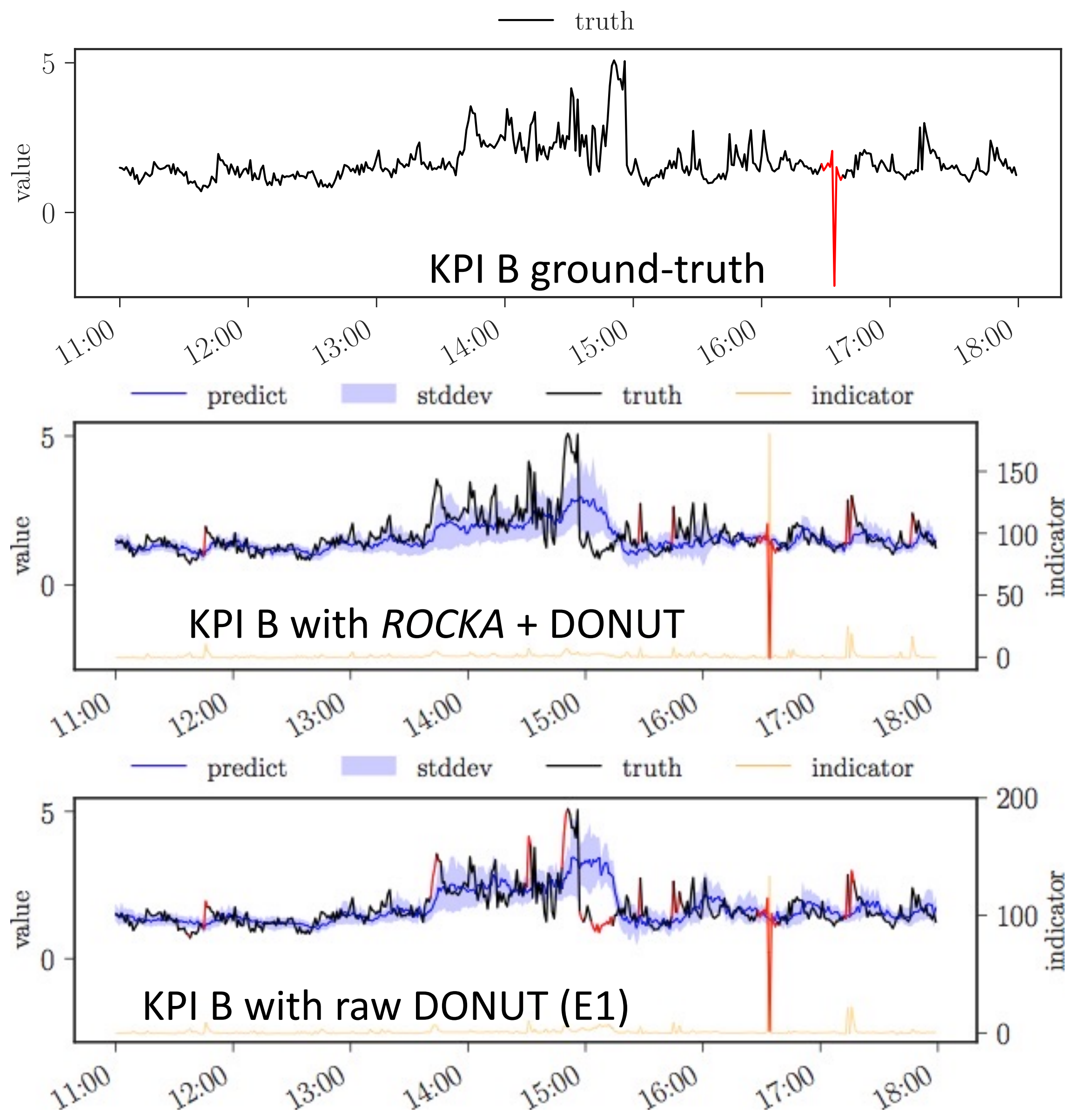


KPI A with ROCKA + DONUT

Orange line is anomaly indicator at each point and red line is the anomalies detected by algorithm. The best threshold on KPI A's centroid is 15.35, larger than the indicator of the most significant anomaly on A (11.90). With the reestimated threshold (10.01), all anomalies on A can be detected.

anomaly detection model can be shared in the same cluster regardless of different anomaly severity levels.

Analysis of Results



The raw DONUT model on KPI B is a bit overfitting and sensitive to small fluctuations. The cluster centroid model is more robust and gets higher F-score.

Outline

- Background
- Algorithm
- Evaluation
- Clustering for KPI Anomaly Detection
- Conclusion

Conclusion

- We propose a **robust** and **rapid** time series clustering algorithm, *ROCKA*, to cluster a large number of KPIs, and assist in anomaly detection.

- *ROCKA* **reduces the model training time** of a state-of-the-art anomaly detection algorithm DONUT by **90%**, with only 15% performance loss. This is the first reported study that uses clustering to reduce the training overhead of KPI anomaly detection.

- *ROCKA* is an important first step towards the direction of using KPI clustering to enable large-scale KPI anomaly detection, a key to ensure service reliability in the Internet.

THANK YOU!

Q&A?

lizhihan17@mails.tsinghua.edu.cn

References

1. D. Liu, Y. Zhao, H. Xu, Y. Sun, D. Pei, J. Luo, X. Jing, and M. Feng, "Opprentice: towards practical and automatic anomaly detection through machine learning," in *Proc. of IMC*. ACM, 2015, pp. 211–224.
2. H. Xu *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proc. of the 27th WWW*. ACM, 2018. [Online]. Available: https://arxiv.org/abs/1802.03903
3. J.Paparrizos and L.Gravano,"k-shape:Efficient and accurate clustering of time series," in *Proc. of SIGMOD*. ACM, 2015, pp. 1855–1870.
4. R. Ding, Q. Wang, Y. Dang, Q. Fu, H. Zhang, and D. Zhang, "Yading: Fast clustering of large-scale time series data," *Proc. of the VLDB Endowment*, vol. 8, no. 5, pp. 473–484, 2015.