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

April 26, 2018

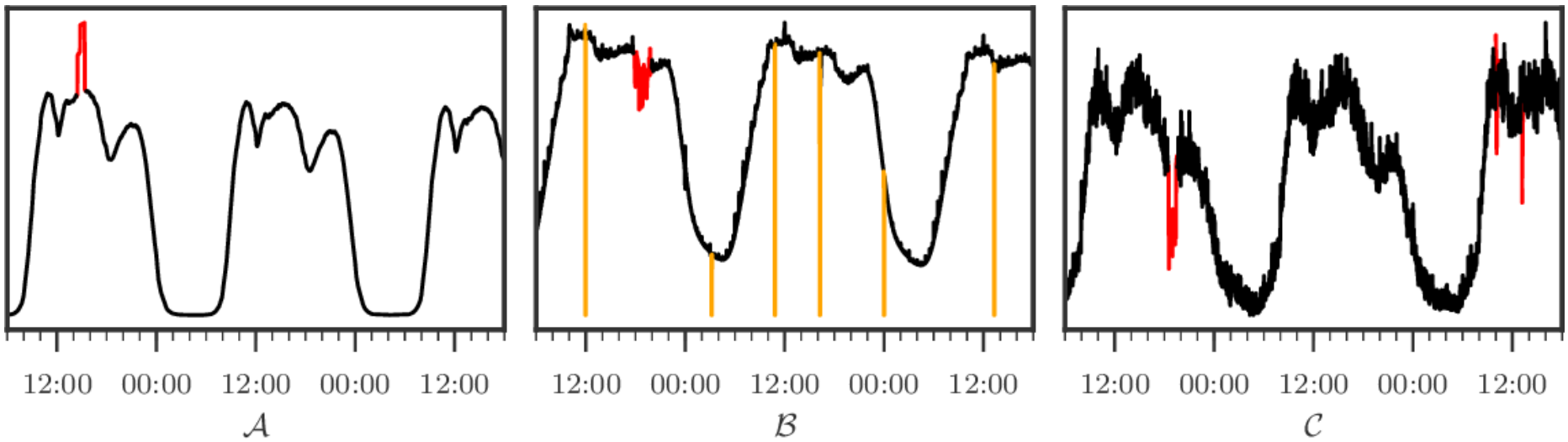
Outline

Outline

Problem Scenario: Anomaly Detection for Seasonal KPIs

KPIs are time sequences, yet one of the most fundamental system monitoring indicators. A failure usually causes more or less anomalies on at least one KPI. Thus anomaly detection for KPIs are very useful in **A**rtificial **I**ntelligence for IT **Op**eration**s (AIOps)**.

For web applications, the user activities are usually seasonal, so are the KPIs, including high level KPIs like the trading volumes, and low level KPIs like the CPU consumptions. We thus focus on **anomaly detection for seasonal KPIs in this work**.



Problem Formulation: Detection of "Abnormal" Patterns

Since KPIs are time sequences, and since in most real cases human operators are willing to see a detection output every time a new observation arrives, the anomaly detection for KPIs can be formulated as:

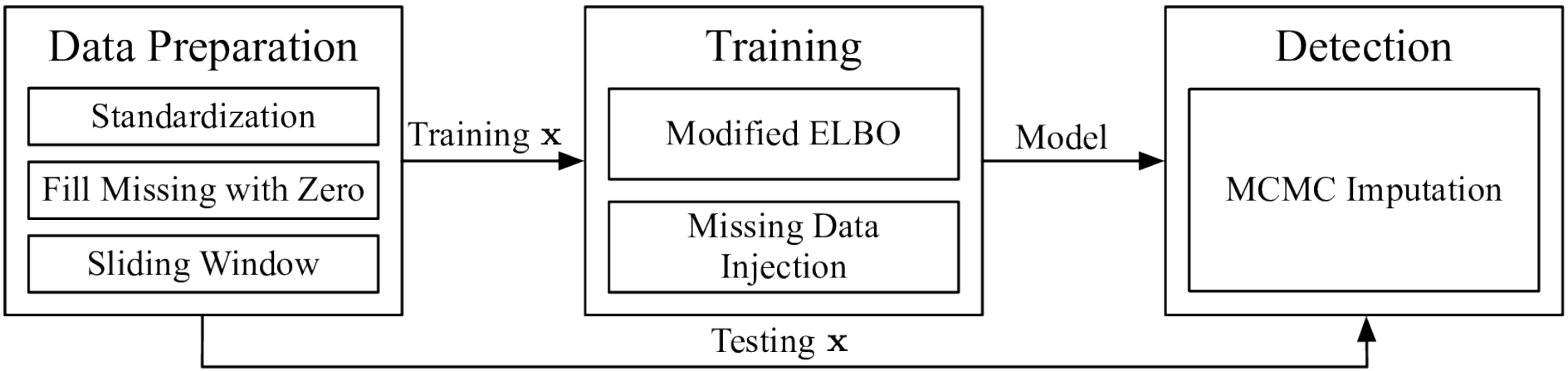
Anomaly Detection for KPIs

For each time t, given the on-time KPI observation x_t and historical observations $x_{t-W+1}, \ldots, x_{t-1}$, determine whether an "abnormal" pattern has occurred (denoted by $y_t = 1$).

Detection algorithms are often designed to compute a **real-valued score** $s(y_t = 1)$ ("**anomaly score**" *hereafter*), *e.g.*, $p(y_t = 1 | x_{t-W+1}, \ldots, x_t)$, leaving the final decision of triggering alerts to the operators.

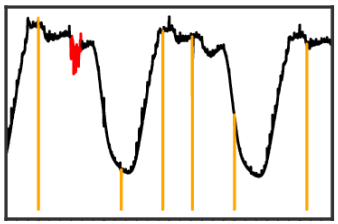
Outline

Overall Architecture of *Donut*



Data Preparation

- Fill Missing with Zero:

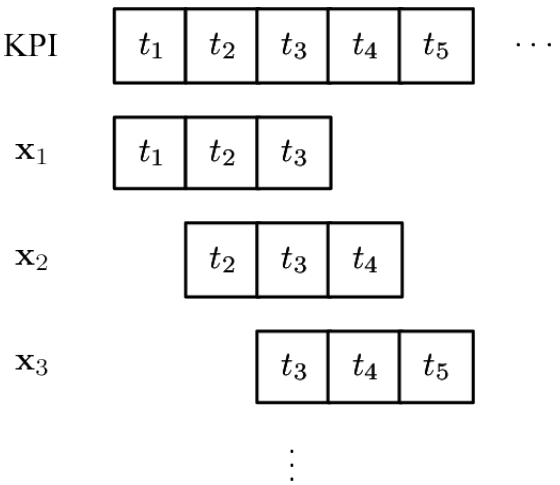


"Missing" are special anomalies, *always* known beforehand. We fill missing points with zeros (orange points in the left figure), and let our model to handle them afterwards.

- Standardization: $\hat{x_t} = (x_t - \mu_x)/\sigma_x$.

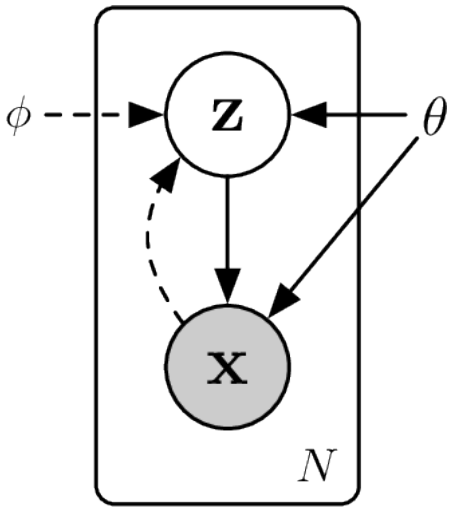
  x_t are the original KPI values, μ_x and σ_x are the mean and std of x_t.
  We shall use x_t to denote $\hat{x_t}$ and neglect the original values x_t *hereafter*.

- Sliding Window:

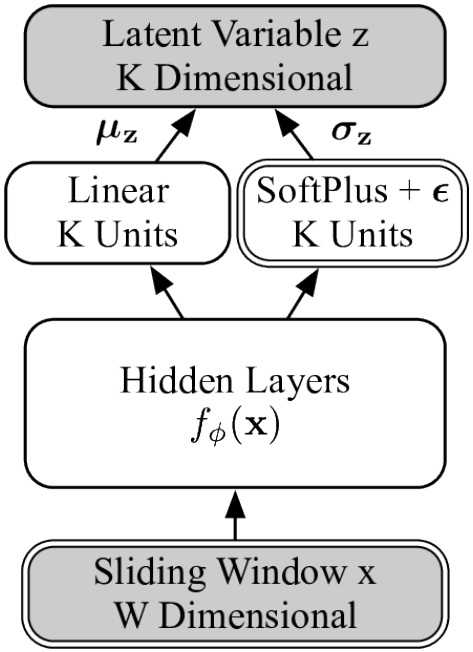


We split the KPIs into fixed-length **sliding windows $\mathbf{x}_t$**, which are assumed to be *i.i.d.*, and are used as **the input $\mathbf{x}$ of VAE** at every time t. For simplicity, we shall omit the subscript t, using $\mathbf{x}$ to denote **the window of "current time"**, and $x_1, \ldots, x_W$ to denote **each point in $\mathbf{x}$** afterwards.

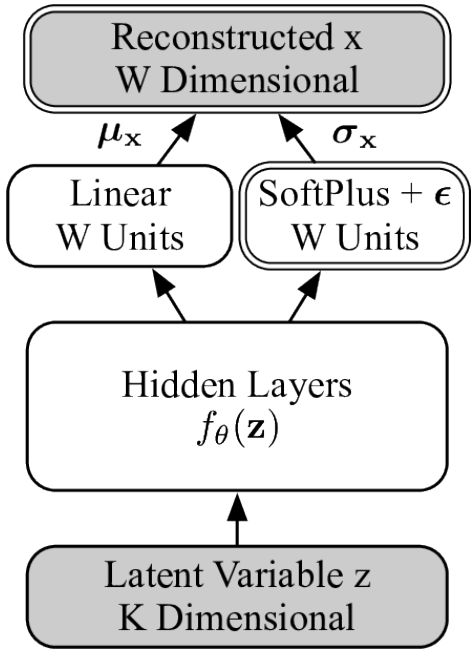
Network Structure



(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)

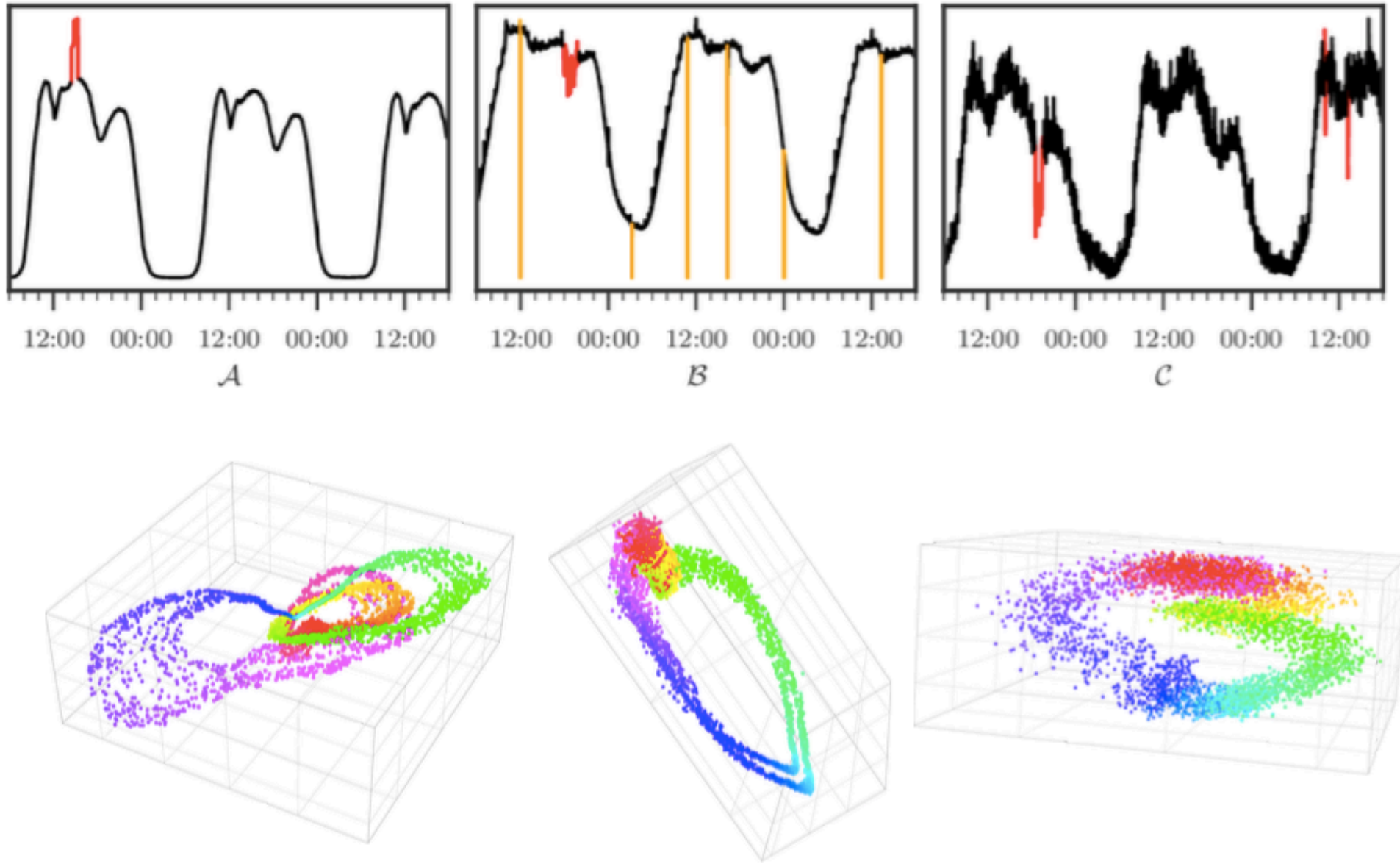


Figure 12: 3-d latent space of all three datasets.

Dealing with Missing and Anomaly

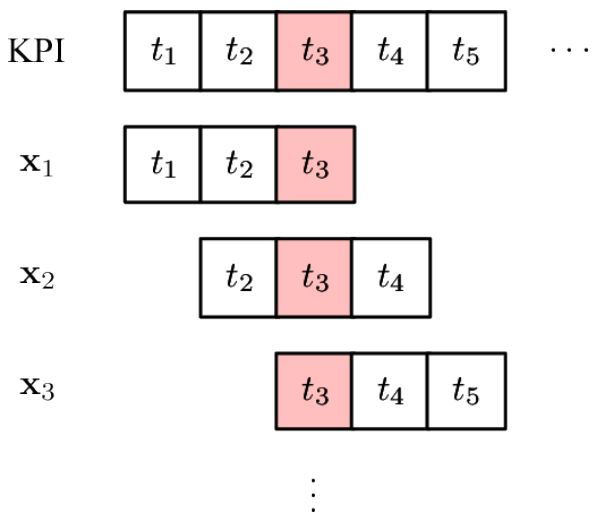


Figure: The anomaly at t_3 shall affect t_4 and t_5, potentially causing trouble in training and detection.

We uses three techniques to handle such "historical anomalies".

1. M-ELBO: We modify the ELBO (objective function) of VAE into **M-ELBO** $\widetilde{\mathcal{L}}(\mathbf{x})$:

$$\widetilde{\mathcal{L}}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{w=1}^{W} \alpha_w \log p_\theta(x_w|\mathbf{z}) + \beta \log p_\theta(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})\right]$$

a_w=0 when anomalous or missing, and a_w = 1 otherwise; beta is the ratio of normal data

2. Missing Data Injection: We randomly set 1% points to be missing at **every epoch** in training, to compensate for having no anomaly labels in the unsupervised scenario.
3. MCMC Imputation (Rezende et al., 2014): In detection, we adopt this technique on known missing points.

The Time Gradient

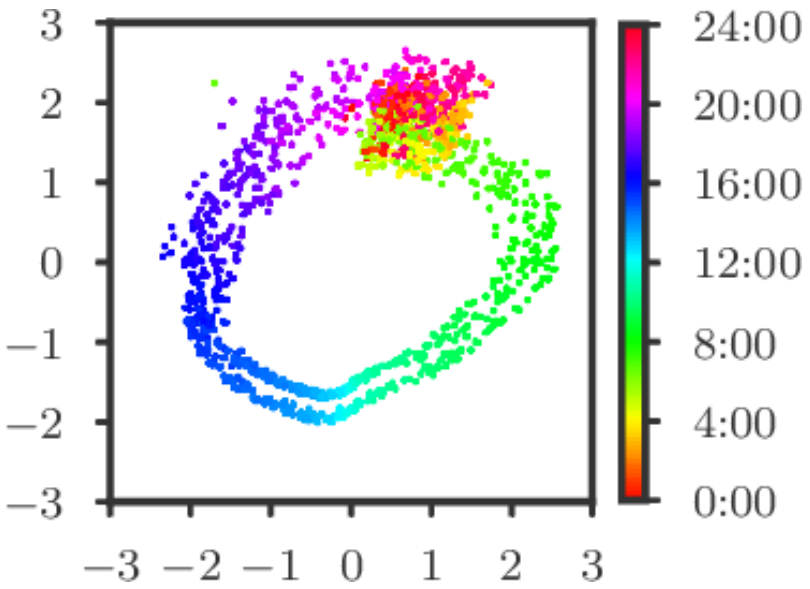


Figure: The $\mathbf{z}$ layout of dataset $\mathcal{B}$. Figure is plotted by sampling $\mathbf{z}$ from $q_\phi(\mathbf{z}|\mathbf{x})$, corresponding to normal $\mathbf{x}$ randomly chosen from the testing set. K is chosen as 2, so the x- and y-axis are the two dimensions of $\mathbf{z}$ samples. The color of a $\mathbf{z}$ sample denotes its time of the day.

- Time gradient: $q_\phi(\mathbf{z}|\mathbf{x})$ are organized in smooth transition: $\mathbf{x}$ at contiguous time are mapped to nearby $q_\phi(\mathbf{z}|\mathbf{x})$.
- Contiguous $\mathbf{x}$ are highly similar in the KPIs of our interest, since they are smooth in general.
- Transition of $q_\phi(\mathbf{z}|\mathbf{x})$ in the shape of $\mathbf{x}$, rather than time, is the cause of time gradient, since *Donut* consumes no time information.
- *Donut* encodes the "shape" or "normal patterns" of $\mathbf{x}$ by $\mathbf{z}$, as shown by the time gradient.
- The time gradient can benefit generalization.

The Causes of Time Gradient

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

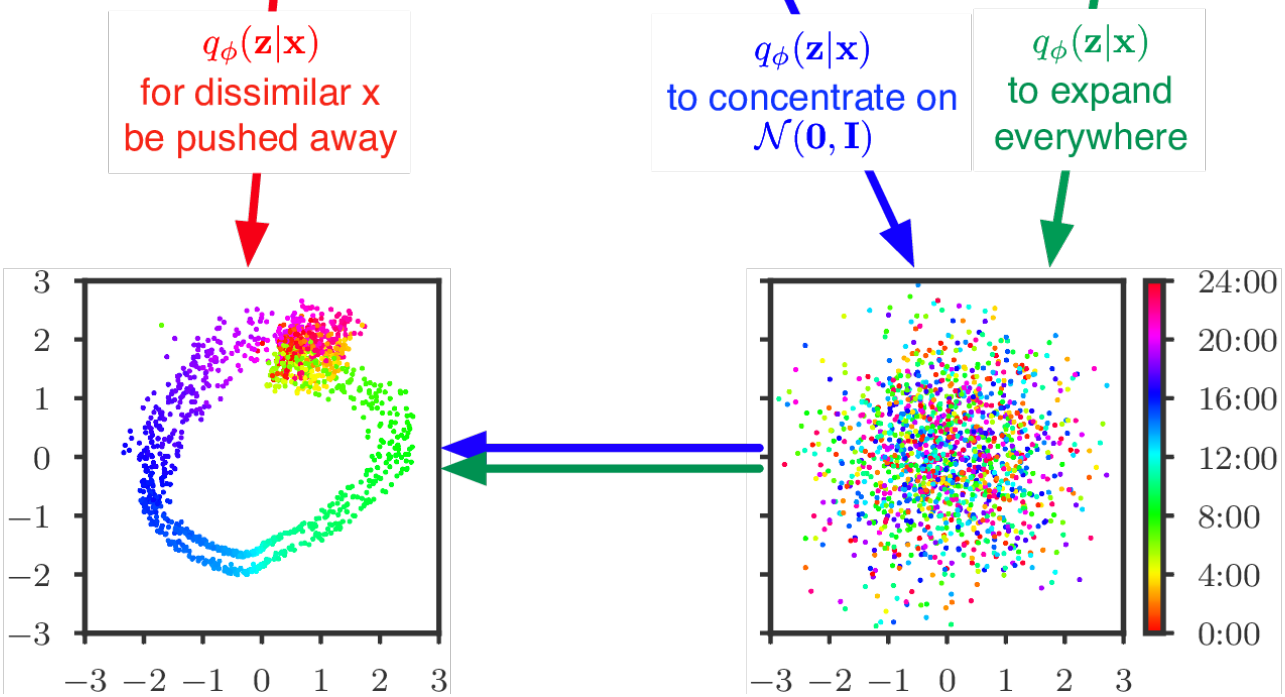


Figure: Causes of the time gradient. Surprisingly, we find no term in ELBO directly pulling $q_\phi(\mathbf{z}|\mathbf{x})$ for similar $\mathbf{x}$ together. The time gradient is likely to be caused mainly by **expansion** ($H(\mathbf{z}|\mathbf{x})$), **squeezing** ($\mathbb{E}[\log p_\theta(\mathbf{z})]$), **pushing** ($\mathbb{E}[\log p_\theta(\mathbf{x}|\mathbf{z})]$), and the **training dynamics** (random initialization and SGVB).

The KDE Interpretation

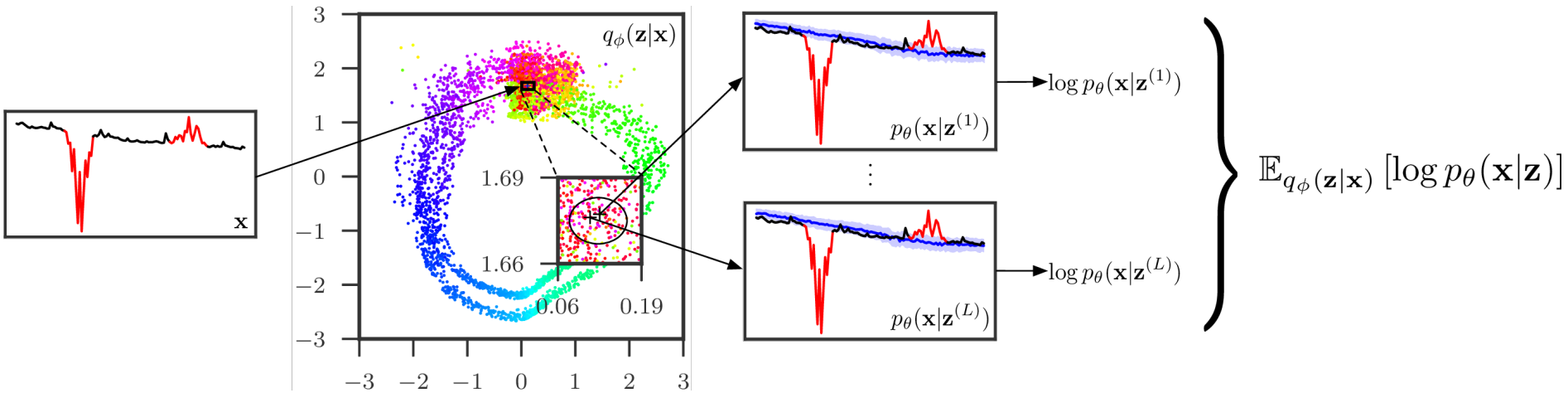


Figure: Illustration of the KDE interpretation. **For a given $\mathbf{x}$ potentially with anomalies, Donut tries to recognize what normal pattern it follows, encoded as $q_\phi(\mathbf{z}|\mathbf{x})$.** The black ellipse in the middle figure denotes the 3-$\boldsymbol{\sigma}_\mathbf{z}$ region of $q_\phi(\mathbf{z}|\mathbf{x})$. **$L$ samples of $\mathbf{z}$ are then taken from $q_\phi(\mathbf{z}|\mathbf{x})$**, denoted as the crosses in the middle figure. **Each $\mathbf{z}$ is associated with a density estimator kernel $\log p_\theta(\mathbf{x}|\mathbf{z})$.** The blue curves in the right two figures are $\boldsymbol{\mu}_\mathbf{x}$ of each kernel, while the surrounding stripes are $\boldsymbol{\sigma}_\mathbf{x}$. Finally, the **values of $\log p_\theta(\mathbf{x}|\mathbf{z})$ are computed from each kernel, and further averaged together as the reconstruction probability.**

The Anomaly Score

An and Cho (2015) has already adopted VAE in anomaly detection tasks of other domain[1]. They use the **reconstruction probability** (1) of truly *i.i.d.* samples $\mathbf{x}$ (*e.g.*, image pixel vectors) as the anomaly score:

$$s(y=1) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \approx \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(\mathbf{x}|\mathbf{z}^{(l)}),\ \mathbf{z}^{(l)} \sim q_\phi(\mathbf{z}|\mathbf{x}) \tag{1}$$

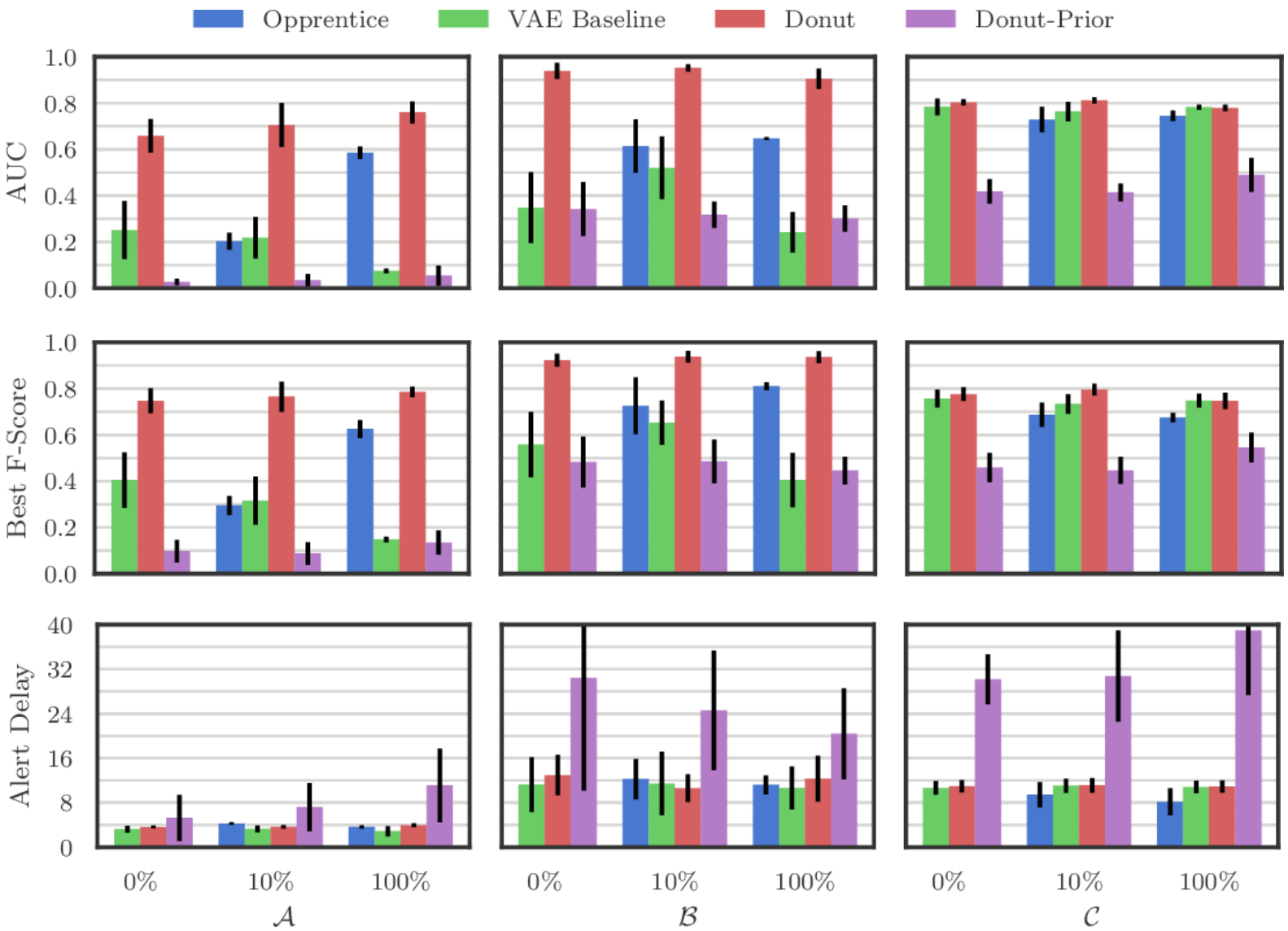
Since the KPIs are time sequences, and the operators are willing to see on-time detection outputs each time a new point arrives, we compute the **element-wise reconstruction probability** (2) for the last point x_W in $\mathbf{x}$, as the anomaly score for the time being:

$$s(y_W=1) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(x_W|\mathbf{z})] \approx \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(x_W|\mathbf{z}^{(l)}),\ \mathbf{z}^{(l)} \sim q_\phi(\mathbf{z}|\mathbf{x}) \tag{2}$$

[1] An and Cho (2015) uses vanilla VAE, without developing techniques like ours to improve performance. We shall compare *Donut* againt their vanilla VAE in evaluation.

Outline

Overall Performance



Effects of *Donut* Techniques

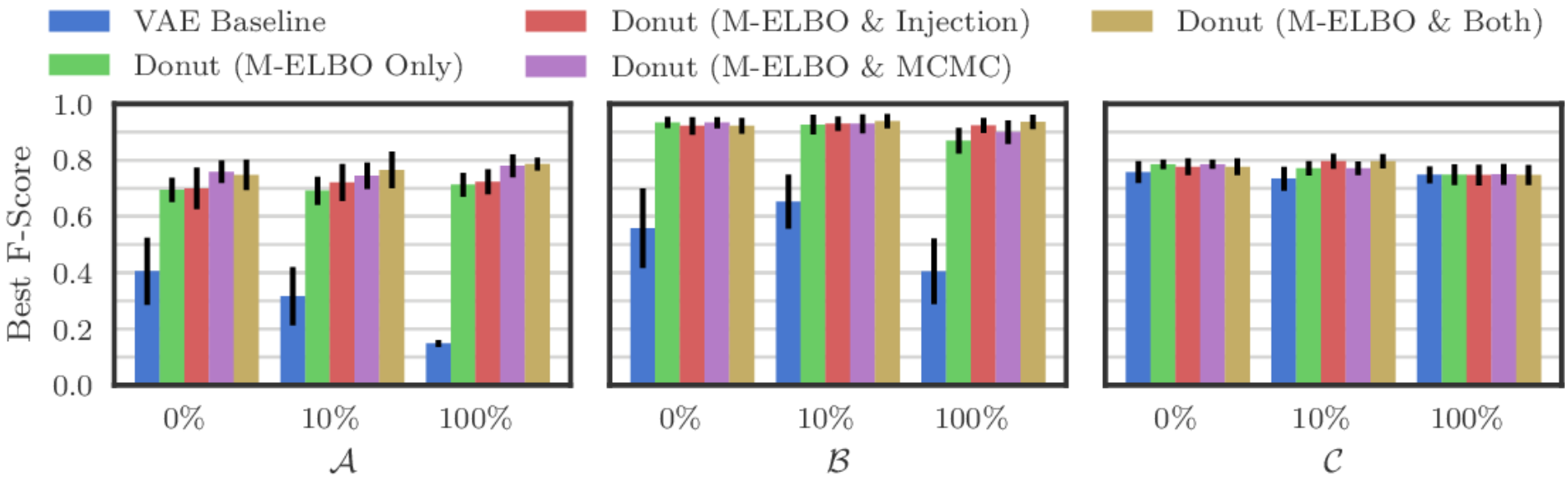


Figure: Best F-score of (1) VAE Baseline, (2) *Donut* with M-ELBO, (3) M-ELBO + missing data injection, (4) M-ELBO + MCMC, and (5) M-ELBO + both MCMC and injection.

The **M-ELBO** alone contributes most of the improvement over VAE Baseline, while the **missing data injection** and the **MCMC imputation** can further benefit the performance.

Impact of Z Dimension Number K

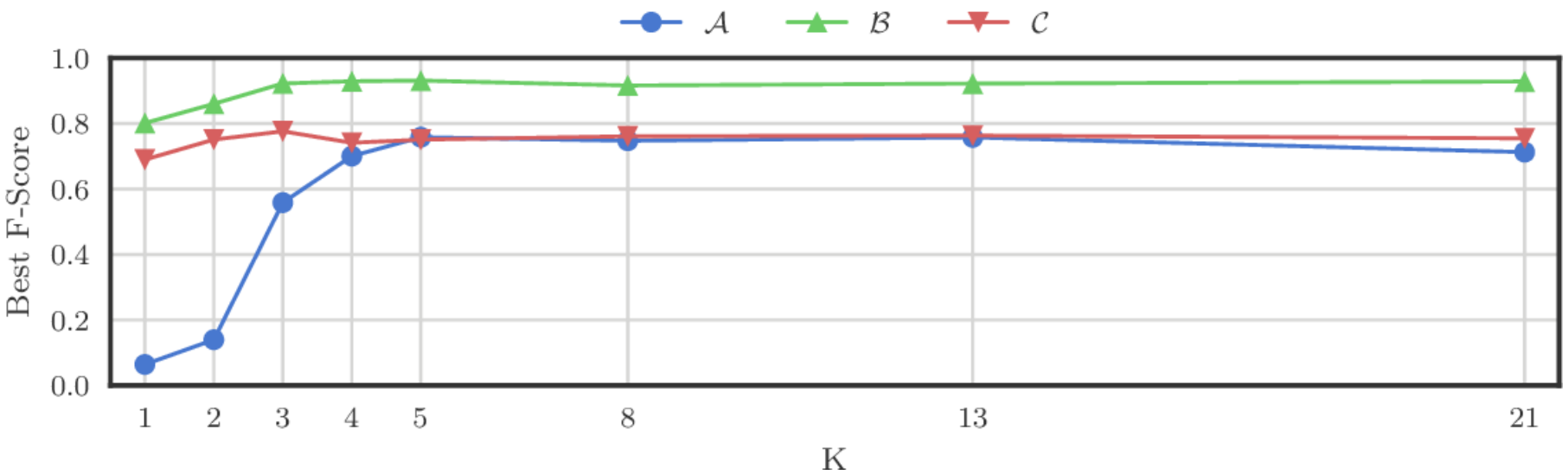


Figure: The best F-score of unsupervised *Donut* with different K on testing set.

- The essential of **Dimension reduction**: W (the dimension of $\mathbf{x}$) is 120, while the best K (the dimension of $\mathbf{z}$) is no larger than 10.
- It should be quite easy to empirically choose K.
  1. The best performance could be achieved with fairly small K.
  2. The performance does not drop too heavily for K up to 21.
- Smoother KPIs seem to demand larger K.

Outline

Conclusion

Our unsupervised anomaly detection algorithm *Donut* for seasonal KPIs, based on VAE, greatly outperforms state-of-art supervised and vanilla VAE anomaly detection algorithms. The best F-scores range from 0.75 to 0.90 for the studied KPIs. The key factors of *Donut* to be successful are:

- Dimension Reduction: forces *Donut* to focus on the overall shape of normal patterns, and gain the ability of resisting abnormal points.
- M-ELBO, Missing Data Injection and MCMC Imputation: further improves *Donut*'s ability to resist abnormal points.

Furthermore, we made the **KDE Interpretation**, which provides a new perspective of VAE-based KPI anomaly detection. All of the above factors can be verified by such interpretation. The KDE Interpretation potentially has more theoretical value in the further development of deep generative models for KPI anomaly detection.

Donut source code published at: `https://github.com/korepwx/donut`.
Full slide: `https://github.com/korepwx/donut/tree/slide`.

Thank you

Q & A

References

An, J. and Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. Technical report, SNU Data Mining Center.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1278–II–1286, Beijing, China. JMLR.org.