Practical Approach to Asynchronous Multivariate Time Series Anomaly Detection and Localization

Ahmed Abdulaal
aabdulaal@ebay.com
eBay Inc
San Jose, California, USA

Zhuanghua Liu
zhualiu@ebay.com
eBay Inc
Shanghai, China

Tomer Lancewicki
tlancewicki@ebay.com
eBay Inc
New York, New York, USA

ABSTRACT

Engineers at eBay utilize robust methods in monitoring IT system signals for anomalies. However, the growing scale of signals, both in volumes and dimensions, overpowers traditional statistical state-space or supervised learning tools. Thus, state-of-the-art methods based on unsupervised deep learning are sought in recent research. However, we experienced flaws when implementing those methods, such as requiring partial supervision and weaknesses to high dimensional datasets, among other reasons discussed in this paper. We propose a practical approach for inferring anomalies from large multivariate sets. We observe an abundance of time series in real-world applications, which exhibit asynchronous and consistent repetitive variations, such as IT, weather, utility, and transportation. Our solution is designed to leverage this behavior. The solution utilizes spectral analysis on the latent representation of a pre-trained autoencoder to extract dominant frequencies across the signals, which are then used in a subsequent network that learns the phase shifts across the signals and produces a synchronized representation of the raw multivariate. Random subsets of the synchronous multivariate are then fed into an array of autoencoders learning to minimize the quantile reconstruction losses, which are then used to infer and localize anomalies based on a majority vote. We benchmark this method against state-of-the-art approaches on public datasets and eBay’s data using their referenced evaluation methods. Furthermore, we address the limitations of the referenced evaluation methods and propose a more realistic evaluation method.

CCS CONCEPTS

• **Computing methodologies** → **Anomaly detection**; Source separation; *Bagging*; *Spectral methods*; *Learning latent representations*; **Neural networks**; • **Applied computing**;

KEYWORDS

anomaly detection, multivariate time series, synchronization, deep learning, representation learning

ACM Reference Format:
Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. 2021. Practical Approach to Asynchronous Multivariate Time Series Anomaly Detection and Localization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3447548.3467174

1 INTRODUCTION

At eBay, site-reliability engineers and technical domain officers depend on real time signal monitoring systems for anomaly detection. They monitor numerous signals from business performance data [29], such as user traffic and activity, to infrastructure data, such as application CPU and memory utilization, in real-time. An anomaly may indicate a potential threat to the business operation. For example, cybersecurity attacks [10, 16] or an internal code bug, which could result in service downtime, immediate tangible business losses, and intangible losses from the loss of customer trust, if not detected and remediated promptly. Therefore, the reliability of the anomaly detection systems is of utmost importance. Systems characterized by high false-positive rates, low true-positive rates, or high latencies would severely impact the system’s reliability.

Traditionally, statistical autoregressive and Gaussian models dominated the time series anomaly-detection field. However, deep learning approaches, such as Long Short Term Memory Recurrent Neural Networks (LSTM-RNN) [10, 16, 18] and Variational Autoencoders (VAE) [14], gained momentum in recent years due to their capability of scaling up to an increasingly dynamic, expanding, and complex multivariate data sets. These data sets produced from the rise of IoT [10, 16], which overpowers the capabilities of traditional methods.

Similarly, the amount of signals being monitored at eBay has been growing substantially year over year, which calls for replacing traditional univariate statistical methods with multivariate neural networks. In particular, unsupervised networks are sought in this research. The reason is that supervised learning approaches face the problem of unobtainable ground truth labels. In contrast, simulated labels are unrepresentative of the true diversity of real-world anomalies [10, 18]. Therefore, unsupervised Deep Learning approaches are undeniably superior in applications where anomaly patterns are inconsistent, scarce, or unlabeled [10, 16, 24].

While recent research approaches achieve competitive performance on public datasets, we experienced concerns regarding their implementation as practical tools for critical decision-making. We concluded that the anomaly inference methodologies, datasets, and evaluation methods reported, had resulted in overestimating the efficacy of those approaches. We discuss the realized flaws in more detail in Section 2. Alternatively, we developed an architecture to improve on the unsupervised deep learning approach methodically. Mainly, our method utilizes priors about the time series and a more

KDD ’21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467174

practical anomaly inference method, with an option to localize anomalies among large feature sets. The theorized improvements in this paper emerge from tackling two limitations for unsupervised approaches in the multivariate context: First, unsupervised methods assume linear dependencies along the spectrum of the series but ignore an often existing non-linearity masked within the inherent correlations of a multivariate time series [16]. Second, time series dimensions are often observed asynchronously in real-world applications, which impedes the learning of appropriate model weights [5]. As a workaround, current solutions often include down-sampling, sliding time-window vectorization, and autoregressive recurrent network approaches. However, such solutions suffer from information loss, low interpretability, high computational costs, increased detection latency, and high false-positive rates, yielding inefficiencies that further inflate as the multivariate dimension increases.

In this paper, we present a deep learning architecture, which prioritizes practical utilization over mathematical complexity. First, we note that we can leverage priors about the time series to improve the learning process; many features oscillate at almost consistent frequencies. Second, we note that reliability engineers and domain officers are more effective at concluding and triaging anomalies when provided with visually interpretable thresholds. For example, the confidence limits in traditional univariate statistical models and providing them with an ability to localize anomalies within the multivariate dataset. Thus we present RANCoders; an architecture for practical anomaly detection with an optional synchronization representation learning phase; RANSynCoders. The latter model utilizes spectral analysis based on the Fourier transform of the latent space representation to identify oscillation frequencies dominant across the raw feature space. If present, the frequencies act as priors that can be leveraged in a separate layer, which learns the synchronized representation of the multivariate input. We hypothesize that this synchronization yields performance gains due to transforming part of the nonlinear dependencies across multidimensional features into linear dependencies. The latter is particularly useful in autoencoders mapping data from real time space to latent space [1, 12, 16, 19, 22–24, 33]. If common oscillations were absent or unidentifiable, then RANSyncoders become identical to RANCoders in the remaining processes. Random feature subsets are fed into multiple autoencoders in a bagging-like manner, each optimizing the quantile reconstruction losses of the full multivariate set. Then, anomaly inference can be obtained via majority voting, and localization is enhanced via feature anomaly-frequency analysis. The intuition behind this approach is that in highly synchronized signals, a small subset holds sufficient information to reconstruct the full dimension of the set during normal operation. We benchmark this method against state-of-the-art approaches from recent research on public datasets and on eBay datasets. Furthermore, we note flaws with the evaluation methods in research and suggest an alternative evaluation method more capable of discriminating among models in practical efficiency. The contributions of this paper can be summarized as follows:

- We propose an architecture of multiple encoders-decoders with random feature selection to infer and localize anomalies through majority voting, where the decoders learn the bounds of the reconstructed signals. In addition, we suggest extracting multivariate signals' priors through spectral analysis on the latent space representation to synchronize the representation of the raw series.
- We analyze the limitations of current widely used evaluation methods, such as the Point adjusted method. We propose a methodology for better evaluating the industry-practical efficacy of anomaly detection models and compare it to commonly used methods in research.
- We experiment with multiple datasets, including real eBay data. Our model achieves improvements over state-of-the-art approaches on most datasets. Furthermore, we investigate the true performance contribution of the underlying methodologies in referenced research through an ablation study. We also make one of our datasets available for public use.

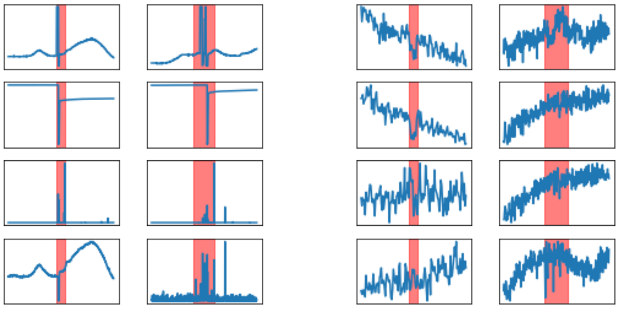
The remainder of this paper proceeds as follows: Section 2 summarizes the findings from reviewed literature on deep learning for time series anomaly detection and highlights the realized weaknesses which motivated this study, Section 3 explains the mathematical formulation and the proposed architecture covering synchronization, anomaly inference, localization, and the proposed evaluation method. Section 4 discusses the experimental setup, the benchmark models, and the datasets. Section 5 discusses the experimental results, and Section 6 summarizes the conclusions from this work and suggests paths for future efforts.

2 BACKGROUND

2.1 Recent SIGKDD trends and limitations

The topic of unsupervised deep learning for multivariate time series anomaly detection has been growing in interest to the SIGKDD community in recent years. In [13], a method based on Long Short Term Memory with unsupervised Nonparametric Dynamic Thresholding (LSTM-NDT) for anomaly inference on predicted results was proposed. The main application targeted spacecraft anomalies, such as the Soil Moisture Active Passive (SMAP) satellite and the Mars Science Laboratory (MSL) Rover. In [28], OmniAnomaly was proposed; a deep Bayesian network utilizing RNNs with stochastic variables to identify anomalies using the reconstruction probabilities. While OmniAnomaly's performance was less superior to LSTM-NDT on the SMAP data set, the method was significantly better on other public datasets, including the Server Machine Dataset (SMD) collected from an internet company and published alongside the OmniAnomaly study. In [2], USAD, an architecture based on adversarial learning in autoencoders, was proposed. Free of RNN layers, USAD has been characterized as one of the fastest learning networks. Continuing the pattern, USAD has outperformed OmniAnomaly on many public datasets and USAD's authors' proprietary data, except for the SMD dataset, published with the OmniAnomaly study and remains a strong proof of its superiority.

There is a noticeable pattern of successive adoption of methods from the latest advances in the language and image domains to the multivariate time series domain, such as LSTM-RNNs inspiring the work in [13, 28], VAEs inspiring the work in [28], GANs inspiring the work in [2]. Another noticeable pattern is the emergence of a new labeled dataset with each study, where the dataset uniquely performs well for the respective study's solution over all other



(a) SMD **(b) BKPI**

Figure 1: Random anomaly samples from a) SMD dataset, and b) eBay business dataset. The shaded region indicates the duration of the labelled anomaly.

solutions. This pattern raises concerns regarding the true robustness of the aforementioned solutions and their extendability to other datasets and industry applications.

Furthermore, we notice that all three methods suffer from a common self-contradictory flaw; While unsupervised anomaly detection is the sought method, they all depend on a posteriori threshold selection for anomaly inference, which requires labeled examples for optimal performance. Some authors were aware of this problem and suggested approaches to approximate threshold selection through dynamic error smoothing over a time window sequence [13], or through applying the principle of Extreme Value Theory (EVT) [26] on the fitted probability distribution for the Bayesian model [28]. However, on experimental analysis, we found that the optimal threshold may fall far from the approximated threshold depending on multiple variables, such as the time window length and the nature of the data.

Another common limitation in these studies is the dependence on ideal time window selection or down-sampling rate for noise reduction, which produces latency in anomaly detection, making these approaches less suitable for application where prompt anomaly detection is critical.

Another concern we raise is the datasets used in the above studies, particularly since one of the main justifications of deep learning over statistical methods is the scalability to higher dimensions. Nevertheless, the datasets used in [13, 28] ranged from 10 to 50 features. The study in [2] included an additional larger dataset of 123 features [20], where the obtained performance was 70% worse across all methods compared to smaller datasets. In comparison, a vanilla isolation forest [17] appeared to be superior to the deep learning solutions on the larger dataset. The second justification of deep learning is the ability to learn complex patterns in the datasets. However, upon closer inspection of the labeled anomalies in the SMD dataset, we note that most anomalies are significant Gaussian outliers, which perhaps explains the superiority of the Bayesian network, OmniAnomaly, on particularly that dataset. Figure 1 compares labeled anomalies picked randomly from SMD and eBay business datasets. Based on the observed nature of anomalies in the SMD dataset, we question the justification of computationally complex deep learning architectures preference over an ensemble of statistical methods of proven performance against Gaussian errors. On the other hand, the anomalies in the eBay business dataset are harder to identify as Gaussian outliers.

The above findings motivated us to present a solution that targets point-anomaly detection rather than the time window approach, does not depend on a posteriori threshold selection, and suitable for a wide range of anomalies.

2.2 Representation learning for time series

Representation learning techniques offer performance gains for high-dimensional time series applications. Different representations unmask desired sources of variation while masking the undesired sources, thus maximize the learning efficiency in neural networks [4]. Autoencoders are a type of representation learning architecture widely used in anomaly detection, novelty detection, denoising on audio signals [19, 22–24] and in text-recognition [33]. In contrast to other architectures, autoencoders are robust to unclean training datasets[30, 33], which makes them ideal for highly dynamic industries with abruptly deviating patterns, such as e-commerce.

2.3 Asynchronicity in multivariate time series

Multi-signal synchrony estimation is well addressed in signal processing literature. Solutions mostly involve bivariate approaches [3] or extrapolation of them to the multivariate domain [21]. Whereas in real time anomaly detection literature, the impact of asynchrony on training and performance is overlooked, albeit being the state of most real-world multivariate time series [1]. In geological monitoring, the presence of unknown correlations among time series measured at different sensors is a known problem [15]. At eBay, asynchrony can be present in series pertaining to different geographical locations of different time zones or due to other spatiotemporal dependencies across features. For example, the user sign-in volume typically lags the cart’s checkout volume. Similar behavior can be observed in machine-level data due to lagged causal dependencies among signals. In other industries, asynchrony can be intended by design, such as in multi-phase electrical systems, sequential operation of machinery in a production system, load-balancing control systems, or due to inconsistent sampling rates, among many other reasons. In a more recent study on noisy multivariate forecast problems [5], the authors tackled asynchronous dimensions by merging them into a single dimension while adding their asynchronous temporal attributes into separate features. They used an architecture combining two sub-CNNs for learning the temporal features mapping with the single dimension and then decompose it back into the original dimension set for forecast results. However, their work is limited to asynchrony in terms of sampling time only, but not in value patterns. Another limitation to their work is the lack of testing on real-life data with relevant characteristics and high dimensionality.

3 METHODS

3.1 Latent spectral density estimation

As indicated in the literature, an exemplary aspect of representation learning is leveraging priors about the data, enabling them to become manifolds guiding other deep learning models [4, 30]. With this opportunity in mind, we train an autoencoder using an extreme latent space of size 1. For the training objective, we minimize the 50th quantile reconstruction loss, denoted by $\min \mathscr{L}_{q=0.5}(\hat{Y}, Y)$. The 50th quantile loss is equivalent to the median of the residuals, which is a suitable metric for dealing with outliers. We then utilize spectral analysis with Fast-Fourier Transform (FFT) [31] on the univariate latent representation to identify a frequency vector W of size S. The W vector contains frequencies theorized to be dominant and commonly present across the series features, where S is the number of sinusoidal components sufficient to approximate the multivariate time series Y. We utilize the determined frequencies for initializing weights in a synchronized representation learning as described in Subsection 3.2. We justify this approach over traditional multivariate synchrony-estimation approaches [3, 21] as follows:

- **Learning efficiency:** Since the autoencoder approach is the underlying methodology for anomaly detection in this paper, the focus is on reducing the amount of information required in the latent space to assist the learning process.
- **adaptiveness:** Typically, anomaly detection models require periodic retraining to adapt to changes in patterns; thus, embedding synchrony-estimation in the network is necessary.
- **Scalability and robustness:** It is faster and more efficient to estimate spectral density from the latent space than from the raw multivariate series, where the presence of anomalies or other impurities may impact the quality of multivariate estimation methods, such as multi-channel Yule-walker [25].

3.2 Asynchronous multivariate signal model

We model an asynchronous multivariate time series of I features and T samples, which have timestamps matrix $\tilde{T}$ and are possibly inconsistent across features, as a combination of S frequency components using the following equation:

$$y_{i,t_i} = \left[\sum_{s=0}^{S} \alpha_{s,i} \sin(\omega_s (t_i + \beta_{s,i})) \right] + \gamma_i + \epsilon_{i,t_i} \quad (1)$$

where $i \in I$ and $t_i \in \tilde{T}$ are the indices of the feature, $y \in Y$, and the feature's timestamp. $s \in S$ is the index of the frequency component, and $\omega \in W$ is the angular frequency in radians. α, β, and γ denote the amplitude, phase, and bias of the feature, and ϵ is the remaining noise in the feature, which is assumed independent for each t_i.

Using Gradient Descent, ω and β cannot be effectively learned due to nonlinearity. However, we initialize ω with the approximate optimal using the method in Subsection 3.1 and further facilitate the learning of β by linearizing the angle part of Equation 1 as $\omega_s t_i + \hat{\beta}_{s,i}$, where $\beta = \frac{\hat{\beta}}{\omega}$ and the linearized form becomes similar to training a neuron with a unit bias and sine activation function. Accordingly, the weights for α, β, and γ are optimized during backpropagation. The noise term ϵ is calculated by subtracting the raw signal from the fitted sinusoidal components. Figure 2 shows an example of one of the features modeled using Equation 1 with two frequency components ($s \in [0, 1]$). This process has a resemblance to statistical time series seasonal-decomposition techniques [8]. However, our process depends on the ith feature timestamp $t_i \in \tilde{T}$ as a model input, which is crucial to manipulate the time series for synchronization, as explained in the following subsection. Finally, we also need to express the error as a function of time, for reasons explained in the following subsection. Thus, we express the noise corresponding to the tth observation of the ith feature as:

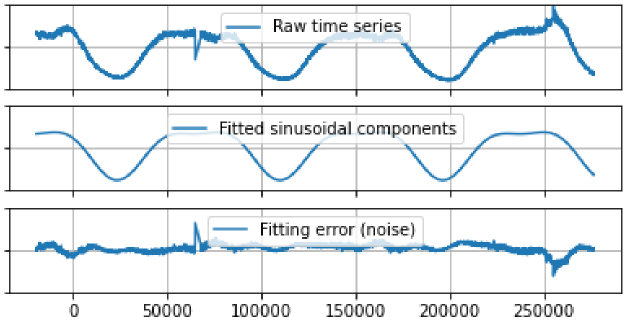


Figure 2: Example of a feature, its fitted sinusoidal components, and the noise. Y-axis is hidden per company policy.

$$\epsilon_{i,t_i} = \alpha_{\epsilon,i,t_i} \sin(\omega_0 (t_i + \beta_{\epsilon,i,t_i})) \quad (2)$$

where ω_0 is the angular frequency of the highest power from the spectral density estimation in Subsection 3.1. β_{ϵ,i,t_i} for the raw asynchronous time series is:

$$\beta_{\epsilon,i,t_i} \approx -t_i + \frac{\nu_0}{4} \quad (3)$$

and $\nu_0 = \frac{2\pi}{\omega_0}$ is the frequency in time steps, thus $\epsilon_{i,t_i} \approx \alpha_{\epsilon,i,t_i}$ from Equations 2 and 3.

3.3 Synchronized representation

Using the formulas presented in Subsection 3.2, we align a multivariate time series of asynchronous feature sample times, t_i, as well as asynchronous phase shifts, $\beta_{s,i}$ and β_{ϵ,i,t_i} to a reference frame, $\bar{t}$, for each observation and across all dimensions, such that:

$$\bar{\beta}_i = \beta_{0,i} + (t_i - \bar{t}) \quad (4)$$

$$y_{i,\bar{t}} = \left[\sum_{s=0}^{S} \alpha_{s,i} \sin(\omega_s (t_i + \beta_{s,i} - \bar{\beta}_i)) \right] + \gamma_i + \epsilon_{i,\bar{t}} \quad (5)$$

and from Equations 2 and 3:

$$\epsilon_{i,\bar{t}} = \epsilon_{i,t_i} \sin(\omega_0 (\frac{\nu_0}{4} - \bar{\beta}_i)) \quad (6)$$

where $\bar{\beta}_i$ is the feature-based phase shift, which shifts asynchronous features into a reference frame for each observation, $y_{i,\bar{t}}$ and $\epsilon_{i,\bar{t}}$ are the synchronized representations of the raw series and the projected noise respectively. Figure 3 shows an example of 4 asynchronous features collected at eBay from users' activity across 4 different time zones and their syncrhonized representation using ($s \in [0, 1]$).

3.4 Embedded representation learning

We embed the synchrony parameter estimation and multivariate synchronization process, described in Subsections 3.2 and 3.3, into the learning process of the network as shown in Figure 4. Different from [5], our approach maintains the input features structure. However, our network takes two inputs; the multivariate time series Y and the features-corresponding timestamp matrix $\tilde{T}$. In no

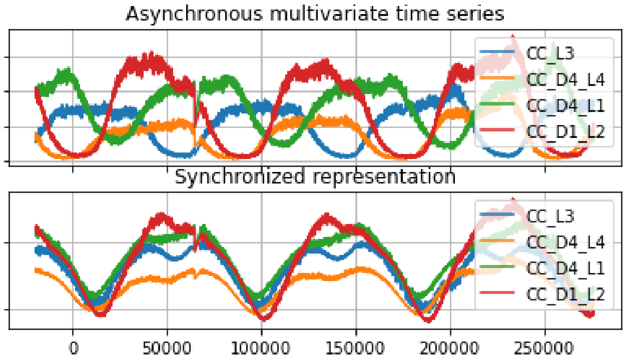


Figure 3: Synchronization of asynchronous time series.

more than the first 5 epochs, the network is set to optimize the frequency components' parameters of Equation 1 independently, by minimizing the noise term ϵ_{i,t_i}. Similar to Subsection 3.1, we use the 50th quantile loss function, denoted by $\min \mathscr{L}_{q=0.5}([\epsilon_{i,t_i}])$, where $\epsilon_{i,t_i} = y_{i,t_i} - [\gamma_i + \sum_{s=0}^{S} \alpha_{s,i} \sin(\omega_s(t_i + \beta_{s,i}))]$ from Equation 1. We continue to train the parameters in series with the remainder of the network for the remaining epochs. We connect the synchronization and anomaly detection parts as follows: the noise term is computed from the fitting error, and the time series is synchronized using Equations 5 and 6, where the sample reference time $\bar{t}$ is either supplied as an input or, in our case, is extracted internally from a reference feature sample time $t_{\bar{i}}$ without supplying additional inputs. The synchronized representation, $y_{i,\bar{t}}$ becomes the input of the RANCoders architecture, described in Subsection 3.5.

3.5 RANCoders: Bootstrapped autoencoders for feature-bounds construction

We propose an alternative approach to utilizing autoencoders for anomaly detection. The method significantly leverages the synchronized multivariate series's enhanced spatial collinearity and performs very well on asynchronous signals. The two main hypotheses, which distinguishes this approach from previous autoencoder models are as follows:

First, a smaller subset of the input could hold enough information to reconstruct the full series at sufficient quality. Thus, we utilize feature bootstrap aggregation (bagging) with a set of N weak deep encoders. Bagging, [6], works by randomly selecting a small subset of the feature set as input for each encoder. The random forests decision tree models inspire this process [7], which have the advantage of significantly decreasing the correlation between trees and increasing the predictive accuracy. However, the purpose of bagging in this paper is not to achieve generalization, which could have been implemented through the dropout technique [27]. Instead, we provide a mechanism for anomaly inference through a majority vote, which we describe in Subsection 3.6.

The second hypothesis is that, for anomaly detection, it is more efficient, in terms of both training and inference, to reconstruct the threshold bounds of the input signal, rather than attempting to rebuild the input followed by an unreliable process of threshold selection for anomaly inference. The former is beneficial since anomaly detection autoencoders ideally require clean input signals to learn the expected behavior, except that clean input signals are not readily available in most industrial applications. Accordingly, we train two sets of deep decoders, AE^{UB}_n and AE^{LB}_n, simultaneously to reconstruct the bounds of the full multivariate from each bootstrapped encoder. The decoders are trained to minimize the quantile reconstruction loss, expressed as:

$$\mathscr{L}_{q=\delta} = \sum_{i=0}^{I} \max\left[\delta(\hat{y}^{LB}_{i,\bar{t},n} - y_{i,\bar{t}}), (\delta - 1)(\hat{y}^{LB}_{i,\bar{t},n} - y_{i,\bar{t}})\right], \forall n \in N \tag{7}$$

for the lower bound, where $\hat{y}$ is the reconstructed bound, LB and UB are short for lower and upper bounds respectively, and n is the index of the decoder pair. δ is the quantile parameter, which is arbitrarily selected as half the user's belief of the amount of noise or anomalies present in the data. The same equation is used for the upper bound decoder as well by setting $q = 1 - \delta$.

Ideally, The number of decoder hidden layers should be at least one layer larger than the encoders' hidden layers since the decoder's output dimension is much larger than the encoders' input. Through the simultaneous training of AE_{UB} and AE_{LB} with opposite loss functions, we observe that the encoders' weights converge faster and more efficiently than other methods, without the need for regularization or other outlier-mitigation methods. It took less than ten epochs for most datasets used to train this architecture.

3.6 Anomaly inference and localization

Two of the highlighted advantages of this approach are inferring anomalies practically and further interpreting potential sources of anomalies within the multivariate dataset. While we discuss both attributes in this subsection, this paper's remainder focuses on the first attribute for performance evaluation purposes since benchmark models do not address localization.

For anomaly inference, binary anomaly votes are collected and aggregated across the spatial dimension by comparing the RANCoders input against the decoded bounds, which can be expressed as follows:

$$P(i_n) \begin{cases} 0 & \text{if } \hat{y}^{LB}_{i,\bar{t},n} < y_{i,\bar{t}} < \hat{y}^{UB}_{i,\bar{t},n} \\ 1 & \text{otherwise} \end{cases}, \forall i \in I, n \in N \tag{8}$$

thus, for each time step, the processed output is a 2-dimensional, $I \times N$, an array of binary values. We, therefore, infer anomalies via majority voting, where the observation is labeled anomalous when there are more 1s than 0s in the output. We conclude this approach as more practical for the industry than a posteriori threshold tuning approach in previous research. The latter requires comprehensive anomaly labeling and frequent revisions for maintaining an ideal threshold, which are unrealistic demands from industry users. Other techniques for unsupervised threshold selection, such as EVT, produce comparable results. However, we noted that their performance is inconsistent across multiple datasets. For example, EVT is less reliable in cases where the training sets are anomaly-free or when the distribution and types of anomalies in the training set are incomparable to those observed post-deployment. Furthermore, the architectures suggested in the reviewed literature generate a continuous, non-Gaussian error score, which disables the option of setting a threshold using a conservative estimate or initial guess,

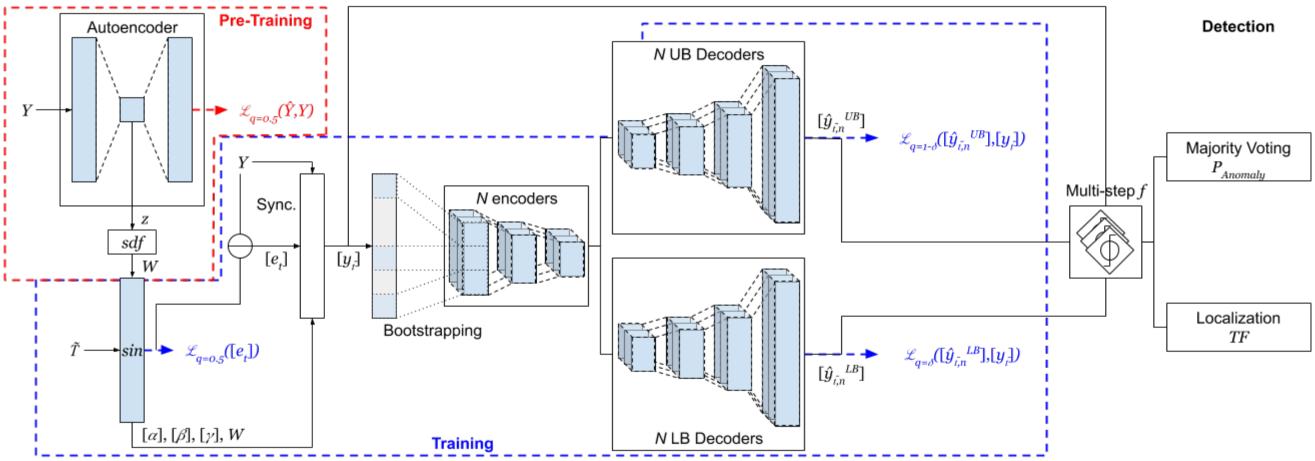

Autoencoder
Pre-Training
Y
$\mathcal{L}_{q=0.5}(\hat{Y},Y)$
z
sdf
W
Y
Sync.
$[e_t]$
$[y_{\tilde{t}}]$
$\tilde{T}$
sin
$\mathcal{L}_{q=0.5}([e_t])$
$[\alpha], [\beta], [\gamma], W$
Bootstrapping
N encoders
N UB Decoders
$[\hat{y}_{\tilde{t},n}{}^{UB}]$
$\mathcal{L}_{q=1-\delta}([\hat{y}_{\tilde{t},n}{}^{UB}],[y_{\tilde{t}}])$
N LB Decoders
$[\hat{y}_{\tilde{t},n}{}^{LB}]$
$\mathcal{L}_{q=\delta}([\hat{y}_{\tilde{t},n}{}^{LB}],[y_{\tilde{t}}])$
Training
Detection
Multi-step f
Majority Voting
$P_{Anomaly}$
Localization
TF

Figure 4: End-to-end network architecture

unlike the majority voting. Nevertheless, our method could utilize threshold tuning, either on top of majority voting or for aggregating the N dimension, to improve performance if comprehensive labeled data were available. However, we note that the majority voting method independently is sufficiently accurate relative to the other architectures.

The second part is anomaly localization, which refers to identifying the feature, attribute, or the set thereof, contributing to the anomaly alert. Localization is essential to facilitating prompt root-cause analysis and, consequently, remediation. For e-commerce business data, localization models have been proposed separately from anomaly detection models [29]. There are two possibilities for anomaly localization enabled by Rancoders' output structure; the first approach, which we had implemented for business data, utilizes a term frequency (TF) analysis of the feature's meta-attributes. For example, for the cart checkout traffic multivariate data, the meta-attributes may include the source traffic location, device, payment method, etc. Thus we compute the frequency of observing each attribute, $attr$, within the out-of-bound features along the I and N dimensions. With this method, we compute the localization score per attribute as: $loc(attr) = \frac{\sum_i \sum_n P(i_n|attr_i)}{n \sum_i attr_i}$, where $attr_i = 1$ if the attribute is present in feature i. The second possibility, which is suitable for different types of features or when when meta-attributes are unavailable, is to utilize a feature-localization score; $loc(i) = \frac{\sum_i \sum_n P(i_n)}{n}$.

3.7 latency and sparsity-aware evaluation

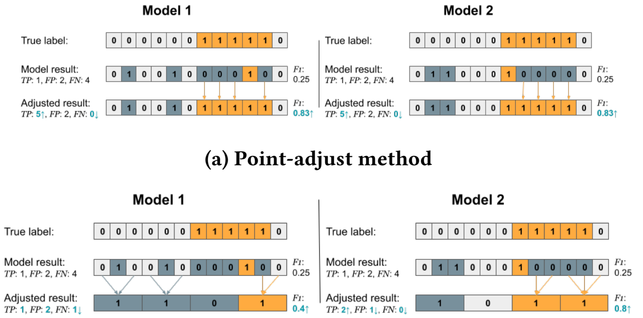
Due to labels imbalance and since their interpretation differs from one user to another, it is common to adjust TP, FP, and FN for detection accuracy evaluation in time series. TP, FP, and FN stand for the True Positive, False Positive, and False Negative anomaly detection counts, respectively. Thus, the authors in [2, 28] adopted a point-adjust approach [32], which adjusts FN into TP by assuming that it is acceptable if an alert is triggered within any subset of a ground truth anomaly segment. We argue that this is not acceptable since it ignores latency and fails to distinguish between two models when they both alert for the same anomaly segments but with different delays. Therefore, we propose an industry-practical approach for adjusting TP, FP, and FN. We justify practicality from the IT-system monitoring view as following: First, we are interested in distinguishing among models in terms of detection latency; thus, only the FN observations after the first TP within the anomaly segment should be adjusted. Second, we recognize that frequent sparse false alerts deem the system unreliable in producing actionable alerts and stresses company resources for triaging and alert-suppression. Thus, we need to discriminate between a model resulting in a continuous stream of FPs within a short time, which are likely caused by a single event, and a model that results in a similar number of FPs but sparsely distributed over a longer period, which indicates that the model is more sensitive to multiple non-anomalous events. To account for sparsity, we utilize an adjustable rolling time window for down-sampling the results. The user selects the down-sampling rate; then, the algorithm temporarily adjusts it in iterations where there is a risk of overlapping distinct segments of ground-truth anomalies. Figure 5 illustrates the proposed method in comparison to point-adjust.

4 EXPERIMENTAL SETUP

4.1 Datasets

We use three publicly available datasets, including a server machine dataset published as part of the contributions of this study and two datasets popular in related research [2, 9, 28]. Additionally, we include results from a business dataset proprietary to eBay. Information regarding the public datasets are provided in Subsections A.1 and A.2 of the appendix. We describe our datasets, labels, production implementation, and other distinct characteristics below:

- **Pooled Server Metrics (PSM):** A dataset collected internally from multiple application server nodes at eBay, anonymized and published with this study. The data consists of 26 features, including localization meta-attributes omitted from the publication due to anonymization requirements. Similar to SMD, the features describe server machine metrics such


Model 1
True label: 0 0 0 0 0 0 1 1 1 1 1 0
Model result: 0 1 0 0 1 0 0 0 0 1 0 0
TP: 1, FP: 2, FN: 4
F1: 0.25
Adjusted result: 0 1 0 0 1 0 1 1 1 1 1 0
TP: 5↑, FP: 2, FN: 0↓
F1: 0.83↑
Model 2
True label: 0 0 0 0 0 0 1 1 1 1 1 0
Model result: 0 1 1 0 0 0 1 0 0 0 0 0
TP: 1, FP: 2, FN: 4
F1: 0.25
Adjusted result: 0 1 1 0 0 0 1 1 1 1 1 0
TP: 5↑, FP: 2, FN: 0↓
F1: 0.83↑
Model 1
True label: 0 0 0 0 0 0 1 1 1 1 1 0
Model result: 0 1 0 0 1 0 0 0 0 1 0 0
TP: 1, FP: 2, FN: 4
F1: 0.25
Adjusted result: 1 1 0 1
TP: 1, FP: 2, FN: 1↓
F1: 0.4↑
Model 2
True label: 0 0 0 0 0 0 1 1 1 1 1 0
Model result: 0 1 1 0 0 0 1 0 0 0 0 0
TP: 1, FP: 2, FN: 4
F1: 0.25
Adjusted result: 1 0 1 1
TP: 2↑, FP: 1↓, FN: 0↓
F1: 0.8↑

(a) Point-adjust method

(b) Latency and sparsity-aware method

Figure 5: An illustration of the difference between a) point-adjust, and b) latency and sparsity-aware method, in discriminating among models of comparable TP and FP counts.

as CPU utilization and memory. The training set consists of 13 weeks, followed by eight weeks for testing. Anomalies are present in both training and testing, with labels prepared only for the latter. The labels were manually created by engineers and application experts, which may include planned anomalies and unplanned anomalies.

- **Business Key Performance Indices (BKPI):** A small subset of important business KPIs related to aggregated users' interaction with specific application features. The dataset was purposefully included to conclude the performance of anomaly detection methodologies due to the distinct challenges it offers above other datasets; First, anomalies often do not constitute Gaussian outliers. Second, user behavior patterns are asynchronous across site features, geographical locations, among other factors. Third, non-anomalous deviations may occur due to external or internal factors such as promotions or auction events. Fourth, with 70 features, this dataset is significantly larger in dimensionality than other datasets, except for the dataset in [20], which proved to be particularly challenging for the benchmark models against classical machine learning approaches [2]. Fifth, this data is of utmost importance in the context of multivariate anomaly detection for identifying and localizing potential interruptions in the service, attacks, or other risks, where anomalies are likely to impact multiple KPIs simultaneously in a non-acute manner. The training set consists of 2 weeks followed by one week of a test set, labeled by Technical Domain Officers and Site-reliability Engineers.

4.2 Benchmark models

To benchmark our model, we selected **OmniAnomaly** [28] and **USAD** [2], which are two state-of-the-art models based on the latest advancements in deep learning architectures from recent literature. The former model adopts variational representation learning [14], while the latter adopts adversarial learning [11] approaches in time series anomaly detection. Furthermore, we created **USAD***, which is a modification of USAD implemented by converting the reconstruction decoder and the discriminator to simultaneous training instead of adversarial training. USAD* enabled us to investigate the true impact of the underlying methodology for IT systems anomaly detection. Additional models' details and their implementations are provided in Section A.3 of the appendix:

4.3 Evaluation metrics and threshold selection

In line with literature, we use Precision, denoted by $Precision = \frac{TP}{TP+FP}$, Recall, denoted by $Recall = \frac{TP}{TP+FN}$, and F1-score, denoted by $F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$, to evaluate all models. However, the scores mentioned above depends on the selected benchmark model thresholds, which is one of the limitations of those models. Thus, we implement two threshold selection strategies to distinguish among models in terms of performance conclusively:

- **Best F-1 ($\mathbf{F1_{best}}$):** Presented in [28] and adopted by [2, 28], this scoring methodology is purely theoretical since it is based on selecting the optimal threshold a posteriori by trying out all possible thresholds on the detection results. While it might be the case that one model is superior when the optimal threshold is known, it is infeasible to achieve this in practice without an exhaustive period of post-deployment anomaly labeling and feedback for each use case. In line with the benchmark literature, we apply the point-adjust method in this strategy by default.
- **Blind F-1 ($\mathbf{F1^*_{blind}}$):** We propose this method to imitate the case of threshold adjustment after deployment in practice; We divide the test set into two halves and identify threshold based on the observed first half, then apply it to the second out-of-sample half. We use the proposed latency-aware and sparsity-aware point-adjust for this strategy by default.

Furthermore, to highlight the functionality of the majority vote system in our architecture over threshold selection approaches, we compute $\mathbf{F1_{init}}$ and $\mathbf{F1^*_{init}}$ as the case of no prior threshold adjustment for the referenced and proposed point-adjust methods, respectively. For USAD and USAD*, this would be assuming a conservative 0.5 threshold since the detection result is in the range 0:1. We omit OmniAnomaly from this part of the comparison since the threshold on stochastic latent units varies on the continuous scale, which is a limitation of OmniAnomaly for some use cases.

4.4 Hyperparameters and settings

One of the observed advantages of the proposed architecture is producing relatively superior performance without requiring hyperparameter tuning. We came to this conclusion since we had not attempted any hyperparameter search and used consistent parameter logic across all datasets. Accordingly, we hypothesize that the performance may considerably increase if we tune some hyperparameters, specifically N, S, and δ. However, since we focus on practical usability, we recognize that hyperparameter tuning can be costly in the real world with label scarcity and thousands of user cases, such as server machines at eBay. See reproducibility Section A for more details.

5 ANALYSIS OF RESULTS

The evaluation results are summarized in Table 1. We note that the proposed architecture yields superior performance for most

Table 1: Performance comparison across 4 deep learning architectures, 4 unique datasets, and 4 evaluation methods.

Sc-ore	Model	SMD			SWaT			PSM			BKPI		
		P	R	F	P	R	F	P	R	F	P	R	F
F1-best	OmniAnomaly [28]	0.92	0.92	**0.91**	0.44	0.92	0.59	0.96	0.81	0.88	0.27	0.70	0.39
	USAD [2]	0.55	0.72	0.50	0.63	0.74	0.68	0.92	0.58	0.71	0.91	0.28	0.43
	USAD*	0.66	0.79	0.62	0.96	0.67	0.80	0.68	0.91	0.78	0.23	0.46	0.30
	RCoders-RSCoders	0.82-0.90	0.80-0.85	0.79-0.83	0.91-0.95	0.77-0.75	0.83-**0.84**	0.99-0.90	0.88-0.96	**0.93**-0.93	0.37-0.38	1.00-0.94	**0.54**-0.54
F1*-blind	OmniAnomaly	0.76	0.58	**0.60**	0.43	0.87	0.58	0.86	0.68	0.76	0.11	0.83	0.19
	USAD	0.22	0.69	0.25	0.19	1.00	0.32	0.00	0.00	0.00	0.11	0.35	0.17
	USAD*	0.28	0.77	0.33	0.19	1.00	0.32	0.55	0.87	0.67	0.06	0.20	0.09
	RCoders-RSCoders	0.29-0.37	0.73-0.80	0.31-0.45	0.51-0.72	0.88-0.75	0.64-**0.73**	0.84-0.76	0.83-0.85	**0.83**-0.80	0.00-0.29	0.00-0.80	0.00-**0.42**
F1-init	USAD	0.37	0.44	0.27	0.18	0.79	0.30	0.00	0.00	0.00	0.00	0.00	0.00
	USAD*	0.44	0.40	0.29	1.00	0.66	**0.80**	1.00	0.00	0.00	0.00	0.00	0.00
	RCoders-RSCoders	0.49-0.54	0.62-0.82	0.44-**0.53**	0.15-0.23	0.97-0.92	0.25-0.36	0.99-0.97	0.88-0.85	**0.93**-0.91	0.00-0.29	0.00-0.28	0.00-**0.29**
F1*-init	USAD	0.39	0.35	0.22	0.18	0.78	0.30	0.00	0.00	0.00	0.00	0.00	0.00
	USAD*	0.46	0.32	0.24	1.00	0.58	**0.73**	0.00	0.00	0.00	0.00	0.00	0.00
	RCoders-RSCoders	0.31-0.37	0.51-0.69	0.30-**0.36**	0.14-0.21	0.94-0.90	0.25-0.34	0.95-0.92	0.77-0.76	**0.85**-0.84	0.00-0.13	0.00-0.13	0.00-**0.13**

datasets and evaluation methodologies. The exception is for the SMD dataset, where OmniAnomaly remains superior. The reported results for SMD are averaged From the 28 datasets. We further discuss the results and learned lessons in the following subsections.

5.1 Impacts of size and synchronization

For all datasets, we apply both RANCoders and RANSynCoders, donated by RCoders and RSCoders in Table 1, where the latter includes the embedded synchronization process. While the overall results in both scenarios were superior to benchmark models, we note that synchronization lifts performance for particularly periodic datasets, such as SWaT, BKPI, and SMD datasets. In particular, RANSynCoders performance was significantly more superior for the BKPI dataset than any other method. Due to the periodic nature of most of the features in the BKPI dataset, we attribute the performance gains to the synchronization process, which transformed temporal covariance among features into spatial covariance, thus reducing the learning load for the autoencoder architecture. An inspection of the covariance matrix before and after synchronization is shown in Figure 6. While the synchronization impact was also noted in other datasets, its impact is more recognizable for higher-dimensional datasets, such as BKPI. Therefore, we conclude that one of the strengths of this architecture is the higher performance for specially higher-dimensional time series. We also want to point out that the suggested implementation by the benchmark models' literature for the SMD data set lacks practical sense. In practice, technology companies, such as eBay, operate thousands of server machines, which challenges the feasibility of deploying and servicing computationally costly VAEs or GANs independently for each server. Therefore, we had designed the proposed architecture to handle pools of interdependent server machines simultaneously per model, albeit their asynchronousity, while taking advantage of the localization feature to enable faster triaging.

5.2 Theoretical versus practical performance

Using various methods for threshold selection and metric adjustment, we highlight discrepancies between the reported performance in literature and the actual performance in practice. Using the latency-aware and sparsity-aware evaluation methods, we note a significant drop in performance for both OmniAnomaly and USAD, relative to the noted drop in RANSynCoders, for 3 of the 4 datasets. Thus, we conclude that the benchmark methods' performance may have been overestimated in the literature. Furthermore, we obtained considerably worse performance for USAD on SMD and SWaT datasets than what the authors reported in [2]. We attribute the discrepancy to two reasons: In contrast to [2], we avoided downs-sampling, which would have reduced the amount of noise. Additionally, we used the entire 38 features of the SMD dataset, including seven problematic features omitted from the referenced study. Finally, the $F1_{init}$ and $F1^*_{init}$ scores prove the practical advantage of bootstrap aggregation and a majority vote in RANSyncoders.

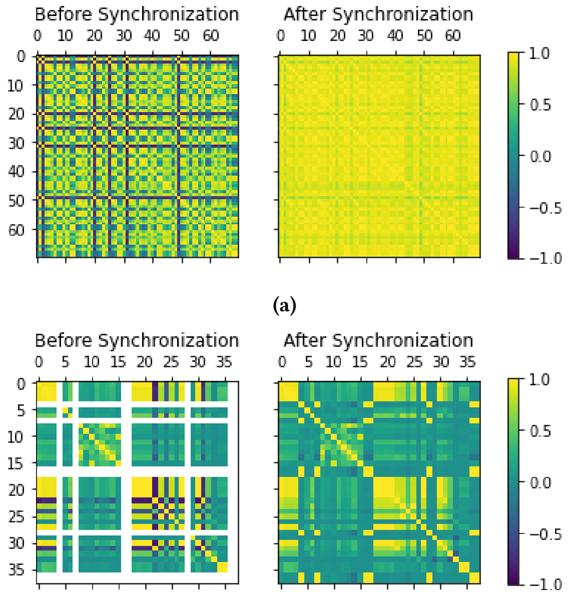
5.3 Notes on VAE and GAN-inspired methods

OmniAnomaly is a state-of-the-art network based on RNNs and VAEs. However, we observed that its superiority on the SMD dataset did not carry over to other datasets. We noted that SMD's labeled anomalies are Gaussian outliers of higher acuteness than labeled anomalies in different datasets. Thus, we question the suitability of variational representation approaches for anomaly detection in the multivariate time series domain. Additional concerns regarding its industry-practical implications are the dependency on ample time windows and high computational costs relative to other approaches.

Alternatively, USAD is based on adversarial learning. Surprisingly, we observed gains in performance after eliminating adversarial training in USAD*, which sums the reconstruction and discrimination losses as a single objective. Thus our findings contradict the assertion that adversarial learning is beneficial in the context of multivariate time series anomaly detection.

6 CONCLUSION

The paper presented an approach for real time anomaly detection in multivariate time series used at eBay. Our method overcomes limitations in previous work, such as dependence on a posteriori threshold identification, time window selection, down-sampling for noise reduction, high computational costs, and inconsistent performance for larger feature dimensions. These are limitations of concern for practical use in industrial applications, such as system monitoring at eBay. The experimental results indicated promising improvements in detection accuracy. The improvements are attributed to the enhanced autoencoder training efficiency via feature



(a)

(b)

Figure 6: Spatial covariance matrix before and after synchronization for a) BKPI and b) Machine 3-1 from SMD datasets.

synchronization, bootstrapping, quantile loss, and majority vote for anomaly inference. The proposed method's additional practical advantages include anomaly interpretability via localization and lower training or parameter setting costs. Furthermore, we proposed an evaluation methodology, considered multiple evaluation scenarios, several public and proprietary datasets of various anomaly types to distinguish between the theoretical and practical performance.

REFERENCES

[1] Ahmed Abdulaal and Tomer Lancewicki. 2021. Real-Time Synchronization in Neural Networks for Multivariate Time Series Anomaly Detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3570–3574.

[2] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. 2020. USAD: UnSupervised Anomaly Detection on Multivariate Time Series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3395–3404.

[3] Selin Aviyente and Ali Yener Mutlu. 2011. A time-frequency-based approach to phase and phase synchrony estimation. *IEEE Transactions on Signal Processing* 59, 7 (2011), 3086–3098.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[5] Mikolaj Binkowski, Gautier Marti, and Philippe Donnat. 2018. Autoregressive convolutional neural networks for asynchronous time series. In *International Conference on Machine Learning*. 580–589.

[6] Leo Breiman. 1996. Bagging predictors. *Machine learning* 24, 2 (1996), 123–140.

[7] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[8] Robert B Cleveland, William S Cleveland, Jean E McRae, and Irma Terpenning. 1990. STL: A seasonal-trend decomposition. *Journal of official statistics* 6, 1 (1990), 3–73.

[9] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. 2016. A dataset to support research in the design of secure water treatment systems. In *International conference on critical information infrastructures security*. Springer, 88–99.

[10] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. 2017. Anomaly detection in cyber physical systems using recurrent neural networks. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 140–145.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc., 2672–2680. https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

[12] James B Heaton, Nick G Polson, and Jan Hendrik Witte. 2017. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry* 33, 1 (2017), 3–12.

[13] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 387–395.

[14] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. arXiv:http://arxiv.org/abs/1312.6114v10 [stat.ML]

[15] Nick Klausner, Mahmood R Azimi-Sadjadi, and Louis Scharf. 2014. Detection of correlated time series in a network of sensor arrays. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3002–3006.

[16] Dan Li, Dacheng Chen, Jonathan Goh, and See-kiong Ng. 2018. Anomaly detection with generative adversarial networks for multivariate time series. *arXiv preprint arXiv:1809.04758* (2018).

[17] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 eighth ieee international conference on data mining*. IEEE, 413–422.

[18] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long short term memory networks for anomaly detection in time series. In *Proceedings*, Vol. 89. Presses universitaires de Louvain, 89–94.

[19] Erik Marchi, Fabio Vesperini, Stefano Squartini, and Björn Schuller. 2017. Deep recurrent neural network-based autoencoders for acoustic novelty detection. *Computational intelligence and neuroscience* 2017 (2017).

[20] Aditya P Mathur and Nils Ole Tippenhauer. 2016. SWaT: a water treatment testbed for research and training on ICS security. In *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*. IEEE, 31–36.

[21] Ali Yener Mutlu and Selin Aviyente. 2011. Multivariate empirical mode decomposition for quantifying multivariate phase synchronization. *EURASIP Journal on Advances in Signal Processing* 2011 (2011), 1–13.

[22] Dong Yul Oh and Il Dong Yun. 2018. Residual error based anomaly detection using auto-encoder in SMD machine sound. *Sensors* 18, 5 (2018), 1308.

[23] Emanuele Principi, Fabio Vesperini, Stefano Squartini, and Francesco Piazza. 2017. Acoustic novelty detection with adversarial autoencoders. In *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 3324–3330.

[24] Ellen Rushe and Brian Mac Namee. 2019. Anomaly detection in raw audio using deep autoregressive networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3597–3601.

[25] Janne M Seppänen, Jukka Turunen, Liisa C Haarla, Matti Koivisto, and Nand Kishor. 2013. Analysis of electromechanical modes using multichannel Yule-Walker estimation of a multivariate autoregressive model. In *IEEE PES ISGT Europe 2013*. IEEE, 1–5.

[26] Alban Siffer, Pierre-Alain Fouque, Alexandre Termier, and Christine Largouet. 2017. Anomaly detection in streams with extreme value theory. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1067–1075.

[27] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[28] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2828–2837.

[29] Yongqian Sun, Youjian Zhao, Ya Su, Dapeng Liu, Xiaohui Nie, Yuan Meng, Shiwen Cheng, Dan Pei, Shenglin Zhang, Xianping Qu, et al. 2018. Hotspot: Anomaly localization for additive kpis with multi-dimensional attributes. *IEEE Access* 6 (2018), 10909–10923.

[30] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. 1096–1103.

[31] Peter Welch. 1967. The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *IEEE Transactions on audio and electroacoustics* 15, 2 (1967), 70–73.

[32] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. 2018. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference*. 187–196.

[33] Chong Zhou and Randy C Paffenroth. 2017. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 665–674.

A REPRODUCIBILITY INFORMATION

A.1 Access to this study's data and files

The anonymized PSM dataset and non-production model files are available on eBay's public GitHub profile: https://github.com/eBay/RANSynCoders. For any access-related problems, please contact the authors to request an email copy of the files. Additionally, the subsections below contain information for reproducing the experiments.

A.2 Public datasets

In addition to eBay's data and production use cases, two publicly available datasets were used in this study. Their descriptions, sources, and implementation details are summarized below:

- **Server Machine Dataset (SMD):** This set is composed of 28 multivariate time series addressed separately. Each series consists of 38 features representing server machine metrics at an internet company, such as memory use and CPU utilization, and are roughly 5-weeks in length. Half the series is used for training, and the other half is labeled for testing. The dataset was published with the study by [28]. There is no information provided regarding anomalies in the training set; however, it is fair to assume unlabeled anomalies in the training set. Furthermore, there were no timestamps provided with this set to address synchronization. However, the publishers' notes indicate that the data were recorded consistently without missing observations or gaps between the training and test splits; thus, we assume timestamps for this data. The apparent sampling frequency in the data is 1 minute.
- **Secure Water Treatment (SWaT) dataset:** A scaled-down version of a water treatment plant, collected from a testbed and published in [9]. The training set consists of 7 days of normal operation, followed by 4 days of intermittent cyber and physical attacks, constituting the labeled anomalies. Therefore, this dataset is ideal for models based on learning the normal behavior. We prioritize this dataset to other public datasets because it contains the timestamp information, with a specified sampling frequency of 1 second.

A.3 Benchmark setup

The selected benchmark models, their sources, and the implementation setup are summarized as following:

- **OmniAnomaly:** The model combined the gated reccurent unit (GRU) variant of RNNS with VAEs and was published with the SMD dataset in 2019 [28]. We used Tensorflow 2.2 to reconstruct the authors' implementation from their shared Github code, written in a less efficient Tensorflow 1.12 version. We validated the updated model version performance using the SMD dataset.
- **USAD:** This model is based on adversarial learning for autoencoders, benchmarked against OmniAnomaly, and published in 2020 [2]. Without a reference to a shared code, we used Tensorflow 2.2 and followed the author's reproducibility instructions from their paper to reconstruct their model.
- **USAD*:** This is a modification of USAD, which we implemented by converting the reconstruction decoder and the discriminator to simultaneous training instead of adversarial training.

A.4 Software packages

Independent of the production implementation at eBay, the authors used an environment with the following packages to benchmark and prototype:

- python 3.7.9
- tensorflow 2.1.0 (gpu build)
- keras 2.3.1 (gpu build)
- cudnn 7.6.5 (cuda 10.1 build)
- numpy 1.19.2
- scipy 1.5.2
- spectrum 0.7.5

A.5 Hyperparameters and settings

Parameters for the presented method were set as following for all datasets: we use 180 for the batch size, 5 pre-training epochs, 10 training epochs, $S = 5$ for frequency components, both N and the bootstrap sample size are selected as one-third of the input dimension I, rounded to the nearest 5, the latent dimensions are selected as $0.5N - 1$. We use $\delta = 0.05$ for system data with Gaussian outliers or $\delta = 0.1$ for business data without Gaussian outliers. For activations, we use Relu and Sigmoid for hidden and output layers respectively. For each encoder n, the input/output layers and dimensions are summarized below:

- Linear: bootstrap sample size (N) -> $\frac{N}{2}$
- Relu
- Linear: size $\frac{N}{2}$ -> $\frac{N}{2} - 1$
- Relu

For each corresponding decoder n, the number of hidden layers are increased since the output dimensions is at least 3 times larger than the encoder input size. This is summarized as following:

- Linear: latent space size $\frac{N}{2} - 1$ -> output size $\frac{I}{4}$
- Relu
- Linear: output size $\frac{I}{4}$ -> output size $\frac{I}{2}$
- Relu
- Linear: output size $\frac{I}{2}$ -> output size I
- Sigmoid

We use the Adam optimizer with the default learning rate of 0.001.

For the benchmark models, we adopt the same parameters suggested by their authors [2, 28], except for downsampling, to maintain a fair comparison with our model. For the eBay datasets, we adopt the authors' parameters from the public datasets closest in dimension.