Journal of Artificial Intelligence Research 1 (1993) 801-824 Submitted 08/2023; published 03/2024

conDENSE: Conditional Density Estimation for Time Series Anomaly Detection.

Alex Moore ALEX.MOORE@HUMA.COM
Huma Therapeutics Ltd,
Millbank Tower, 21-24 Millbank,
London SW1P 4QP, United Kingdom
Corresponding Author

Davide Morelli DAVIDE.MORELLI@HUMA.COM
Huma Therapeutics Ltd &
Institute of Biomedical Engineering,
Department of Engineering Science, University of Oxford,
Oxford OX3 7DQ, United Kingdom

Abstract

In recent years deep learning methods, based on reconstruction errors, have facilitated huge improvements in unsupervised anomaly detection. These methods make the limiting assumption that the greater the distance between an observation and a prediction the lower the likelihood of that observation. In this paper we propose conDENSE, a novel anomaly detection algorithm, which does not use reconstruction errors but rather uses conditional density estimation in masked autoregressive flows. By directly estimating the likelihood of data, our model moves beyond approximating expected behaviour with a single point estimate, as is the case in reconstruction error models. We show how conditioning on a dense representation of the current trajectory, extracted from a variational autoencoder with a gated recurrent unit (GRU VAE), produces a model that is suitable for periodic datasets, while also improving performance on non-periodic datasets. Experiments on 31 time-series, including real-world anomaly detection benchmark datasets and synthetically generated data, show that the model can outperform state-of-the-art deep learning methods.

1. Introduction

Anomaly detection, or outlier detection, is the identification of data points that have deviated from a learnt concept of normality (Ruff, et al., 2021). It is an increasingly active area of machine learning research with applications in a wide range of fields, including: health and medical risk, financial surveillance, risk management, and security (Pang, et al., 2020).

Anomalies can be categorized into three broad categories. Point anomalies are individual data points that are standalone outliers with respect to the rest of the data. Collective anomalies occur when groups of data points appear anomalous with respect to the rest of the dataset. Finally, context anomalies, also known as conditional anomalies, are data points that are only considered anomalous if they occur in particular contexts (Chandola, et al., 2009). Context anomalies are commonly found in seasonal or periodic data (Chalapathy & Chawla, 2019). Considering how widespread periodic phenomena are in biological or behavioural systems (Ushakova, et al., 2021; Ahdesmäki, et al., 2005), catching context anomalies is an important requirement for anomaly detection algorithms.

There are a number of techniques that can be applied to anomaly detection, from distance-based methods (such as K-means) to tree-based methods (such as isolation forests), systematic comparisons have found K-means to be particular effective (Schmidl, et al., 2022). In recent years there has been a focus on applying deep learning architectures to outlier detection, with many reporting state of the art performance (Pang et al., 2020; Chalapathy & Chawla, 2019; Tuor, et al., 2017; Alqurashi, et al., 2021; Deng & Hooi, 2021). Among them, graph neural networks and transformer based architectures have been found to be some of the most powerful. (Deng & Hooi, 2021; Tuli, et al., 2022). However, there is some evidence that deep learning approaches may not justify their increase complexity, with recent work suggesting that traditional anomaly-detection methods perform better (Schmidl et al., 2022; Rewicki, et al., 2023).

The vast majority of unsupervised deep learning methods use reconstruction errors. These methods make predictions about the current state of a time-series and use the difference between their prediction and the observed value, the reconstruction error, as an anomaly score. If the aim is to classify anomalous data points, extreme value analysis is typically used to define a threshold on this anomaly score (Deng & Hooi, 2021; Benecki, et al., 2021; Su, et al., 2019; Tuli et al., 2022; Audibert, et al., 2020; Boniol, et al., 2021; Kingsbury & Alvaro, 2020; Li, et al., 2019; Zhang, et al., 2019; Zhao, et al., 2020).

The reconstruction error approach to anomaly detection is based on a limiting assumption that the greater the distance between an observation and a prediction the lower the likelihood of that observation. Consider the example of a continuous random variable with a bimodal probability density function (PDF) with two distinct peaks, the distance between both peaks will always be greater than the distance between each peak and the local minimum separating them. Accordingly, if a model predicted a value that fell on one peak of the PDF then an observation found at the local minimum would always produce a smaller reconstruction error, and (incorrectly) be judged to have a higher associated likelihood, than an observation found on the other peak. This example highlights the problems with assuming likelihood is proportionate to distance from a prediction. By relying on a point estimate of expected behaviour, reconstruction error models can overlook complexity in the underlying data generating process.

Density estimation algorithms learn the PDF describing the expected range of values a variable can take (Papamakarios, et al., 2018). Historically, they were a popular choice for anomaly detection (Wang, et al., 2019; Breunig, et al., 2000) and a number of light-weight density estimation algorithms are still widely used, such as LODA (Pevný, 2016) and COPOD (Li, et al., 2020). They assume outliers are more likely to be found in areas with lower density, this is a far more flexible assumption than the idea underpinning reconstruction errors. Surprisingly, there have been few studies leveraging the increased power of deep learning by density estimation anomaly detection algorithms (Zong, et al., 2022). Recent work shows that when compared to reconstruction error models deep density estimation models, such as deep autoencoding Gaussian mixture models, tend to perform worse (Tuli et al., 2022).

Our proposed model, conDENSE, does not rely on a reconstruction error but rather uses normalising flows for density estimation. Normalising flows are a class of deep generative models that are highly efficient density estimators (Papamakarios et al., 2018). Their ability to quantify the likelihood of observing unseen data makes them ideal outlier detectors

(Schmidt & Simic, 2019). conDENSE learns a PDF, describing the data generating process, and then uses it to calculate the likelihood of observations. In keeping with previous work, we use extreme value analysis to find a likelihood threshold if required for anomaly classification. conDENSE uses normalising flows for density estimation in anomaly detection. Our primary contribution is conditioning the normalising flows on dense representations of the current trajectory, this generally improves performance and allows the model to handle periodic datasets.

Unlike recent applications of normalising flows to time series, which have focused on estimating the likelihood of a window of observations (Schmidt & Simic, 2019), conDENSE calculates likelihood estimates for single observations, without ignoring short term dynamics.

In this paper we demonstrate that conditional density estimation in normalising flows is well suited to anomaly detection. We highlight how conditioning on a dense representations of the current trajectory allows them to achieve state-of-the-art performance on both periodic and non-periodic datasets. We also visualise the PDFs learnt by our models, illustrating how density estimation can increase the interpretability of anomaly detection algorithms.

2. Related Work

In this section we review the theoretical concepts that underpin our model, namely: normalising flows, and variational autoencoders (VAEs).

2.1 Normalising Flows

Normalising flows are generative models that can be used for sampling and density estimation (Shchur, et al., 2019; Kumar, et al., 2019). When using normalising flows to estimate the probability density function of a random variable $\mathbf{X}$, you can assume $\mathbf{X} = f_\theta(\mathbf{Z})$ where $f_\theta : \chi \to Z$, is a bijective or invertible function and $\mathbf{Z}$ is another random variable with a tractable density function p_z. The change of variables formula can then be used to obtain the PDF of $\mathbf{X}$:

$$p_x(x) = p_z(g(x)) \Big| \det \Big(\frac{\partial g}{\partial x} \Big) \Big| \tag{1}$$

where $g = f_\theta^{-1}$ and $\partial g / \partial x$ is the Jacobian matrix of this inverse function.

Masked autoregressive flows (MAFs) are a type of normalising flow; they use the autoregressive property to ensure a triangular Jacobian, calculating the determinant of a triangular matrix is trivial (Papamakarios et al., 2018). As can be seen in Equation 1, calculating the determinant of the Jacobian matrix is essential to density estimation, consequently MAFs are highly efficient density estimators. MAFs use masking in Masked Autoencoder for Distribution Estimation (MADE) networks to approximate autoregressive functions in a single pass, avoiding the recursion typically associated with autoregressive models (Germain, et al., 2015). Stacking a series of MADE blocks facilitates the learning of more expressive probability densities (Papamakarios et al., 2018).

The inputs to each MADE block can be augmented to include a conditioning variable (Papamakarios et al., 2018). This will extend an unconditional MAF and allow it to estimate conditional densities $p(x|y)$.

2.2 Variational Autoencoders

VAEs are a class of generative model that can be used for dimensionality reduction (Sun, et al., 2018) and anomaly detection (Chatterjee, et al., 2021). Similar to other autoencoder models, VAEs pass data through an encoder, which produces a dense latent representation. A decoder then attempts to accurately recreate the original input from this smaller latent representation (Baldi, 2012). VAEs extend on traditional autoencoders by using a variational Bayesian approach to transform a point estimate in latent space to a distribution in latent space (Kingma & Welling, 2013). VAEs can be trained by maximizing the evidence lower bound (ELBO) on the marginal log-likelihood:

$$\log(p(x)) >= ELBO = E_{q(z|x)}\Big[\log \frac{p(x,z)}{q(z|x)}\Big] \tag{2}$$

Recurrent architectures, such as long short-term memory networks (LSTMs) or gated recurrent units (GRUs), are often combined with VAEs to improve their performance on sequential data (Lin, et al., 2020). VAEs can be used directly for anomaly detection by considering the reconstruction error (Chatterjee et al., 2021; Lin et al., 2020). The latent representations can also be used as a dense representation of the data (Sun et al., 2018).

3. Methods

In this section, we outline conDENSE's model architecture, paying close attention to key sub-components. We also give details on the experiments we ran comparing conDENSE to benchmark anomaly detection algorithms.

3.1 Problem Definition

If we consider the following time series of length N: $T = \{X_1, X_2, ..., X_N\}$. X_t are the observations at time point t, with $X_t \in R^d$ where d is determined by the number of sensors recording at each time point, $d = 1$ for univariate time series. If $\tilde{T}$ is an unseen time series of length $\tilde{N}$ derived from the same system as T , our aim is to predict Y, with $Y = \{y_1, y_2, ..., y_N\}$ where $y_t \in \{0, 1\}$ is a binary variable indicating if the observation at time t in $\tilde{T}$ is anomalous.

3.2 Model Architecture

A summary of the conDENSE model architecture can be seen in Figure 1. There are three core components to conDENSE: a MAF which predicts the conditional likelihood of an observation X_t given the current trajectory of the time series, a GRU VAE that produces a dense representation of that current trajectory (this representation is parameterised by two vectors μ and σ, which are fed into the MAF as conditional variables), and finally a peak over threshold algorithm which is used to set a likelihood threshold (any future observations that fall below this threshold will be considered anomalies). Subsequent subsections will discuss some of these components in more details.

In order to train conDENSE in an end to end fashion we defined the following custom cost function:

$$\mathcal{L} = -\log(p(X_t|\mu, \sigma)) + \lambda_l + \beta_r\Big(\frac{\sum_{i=1}^{n}(Z_i - \hat{Z}_i)^2}{n}\Big) \tag{3}$$

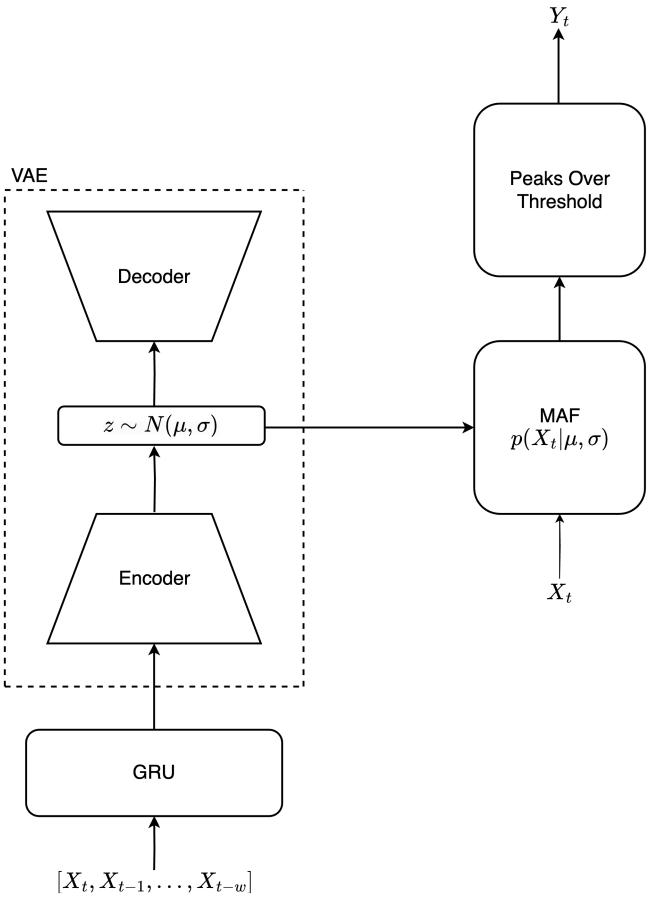


Figure 1: conDENSE model architecture

where $-\log(p(x_t|\mu, \sigma))$ is the negative log likelihood of the training data, λ_l is a regularization term that penalizes the model if the latent representations do not resemble the unit Gaussian, and the final term is the reconstruction error obtained by comparing the input to the GRU VAE with its decoded output, this can be used to approximate the maximizing of the evidence lower bound in Equation 2. The reconstruction term is multiplied by a constant discount parameter $\beta_r \in \{0, 1\}$, this discount term ensures that the loss at each epoch is primarily determined by the output of the likelihood calculation produced by the MAF ($\beta_r = 0.1$ was selected during hyperparameter optimisation).

3.3 GRU VAE

The GRU VAE is an essential component of the conDENSE model. It receives a window of observations, ending with the current observation, as its input and it outputs a dense representation of that window. This dense representation is used as a conditioning variable in the MAF. The GRU VAE performs dimensionality reduction across time and, in the case

of multivariate inputs, space. The tractable representation will describe how the system has been behaving over this window. This can capture a number of crucial pieces of information, such as: the relative position in a cycle of periodic data, the smoothness of the time series over the window, and whether or not different components in multivariate systems are interacting as expected.

Our VAE includes three components: a GRU to capture temporal interactions between inputs, an encoder that produces a dense representation of the input, and a decoder that attempts to reconstruct the original input. This reconstructed input is only used during training, to calculate a reconstruction error, which is included in the global cost function Equation 3. In order to prevent this reconstruction error term from biasing the output of the network too strongly, it is multiplied by a constant discount factor.

The choice of a GRU over other sequential models was primarily motivated by parsimony. Initial experiments found no significant performance difference when using an LSTM or a GRU, so we selected the recurrent architecture with fewer parameters. Our use of relatively short trajectory windows also mitigates the risk of vanishing gradients and reduces the added value of larger architectures, such as transformers, that are better suited to detecting long-term dependencies.

As detailed above, VAEs produce distributions in latent space rather than point estimates. In conDENSE each unit in the latent representation is a Gaussian distribution, implemented in a TensorFlow probabilistic layer, regularization during training ensures that these resemble the unit Gaussian. The mean and standard deviation from each unit in the latent representation are passed as conditional variables to the MAF in a conDENSE model. The number of units in the latent representation is a key hyperparameter for the model. The other key hyperparameters are: the size of the window of previous observations fed into the GRU VAE, the number of units in the GRU, the number of hidden layers in the encoder and decoder, and the dimensionality of each of these layers.

3.4 Peaks Over Thresholds

In keeping with previous work we use the series peak over threshold algorithm to set the threshold for anomaly detection (Deng & Hooi, 2021; Benecki et al., 2021; Su et al., 2019; Tuli et al., 2022; Audibert et al., 2020; Boniol et al., 2021; Kingsbury & Alvaro, 2020; Li et al., 2019; Zhang et al., 2019; Zhao et al., 2020). The method uses extreme value theory over the anomaly scores from the training set to determine the threshold (Siffer, et al., 2017). This method allows us to set an anomaly threshold without labelled anomalies in the training set.

3.5 Data Preprocessing

In order to improve accuracy and increase stability, each feature is normalised independently using min-max scaling:

$$X_t^d \leftarrow \frac{(X_t^d - min(\mathbf{X}_{train}^d))}{(max(\mathbf{X}_{train}^d) - min(\mathbf{X}_{train}^d))} \tag{4}$$

where X_t^d is the value of a given feature at a particular time and $\mathbf{X}_{train}^d$ is the full vector of values used in the training set for a given feature. To avoid information leakage, data in the test set is processed using the min and max values obtained from the training set.

Windows of previous observations are created for the GRU VAE component of conDENSE: $\mathbf{W} = [X_t, X_{t-1}, ..., X_{t-w}]$ where w is the window size. For observations found in the first w steps of a time series a constant value is used for padding. Observations that are included in $\mathbf{W}$ are also scaled according to Equation 4

3.6 Datasets

The performance of our model was evaluated using a combination of real-world and synthetic data. Assessment on commonly used anomaly detection benchmark datasets allows us to see how well our model generalises when applied to real-world time-series, with unknown data generating processes. It also facilitates direct comparisons with similar work. On the other hand, assessment on synthetic datasets with known properties allows us to make more concrete claims about the types of anomalies our model can handle. In total we used 11 real world time-series and 20 synthetic time-series.

3.6.1 Real-World Datasets

Dataset	Train Size	Test Size	Dimensions	Periodic	Number of Time-Series	Anomaly Frequency (%)
UCR	2025	5475	1	True	4	1.06%
SMD	26959	26959	38	True	3	4.31%
SWaT	8250	1441	51	False	1	6.11%
MSL	2110	6248	25 or 55	False	3	8.59%

Table 1: Real-world dataset statistics

The characteristics of the 4 real-world datasets (11 time-series) used to evaluate the performance of our model are summarised in Table 1. Our goal was to evaluate our model in a range of contexts, as a result we have considered both periodic and non-periodic datasets as well as univariate and multivariate datasets.

- *Hexagon-ML* (UCR): we consider four sequences of an internal bleeding dataset (Dau, et al., 2019). Each sequence is univariate. Since the regular beating of the heart drives the recorded values this dataset displays high periodicity. The UCR dataset is also of interest as all of its test set anomalies are context anomalies.

- *Server Machine Dataset* (SMD): a 5-week-long dataset collected from a large internet company, the dataset monitors resource utilisation from 38 different machines (Su et al., 2019). In keeping with (Tuli et al., 2022) we only consider the following non trivial sequences: machine 1-1, machine 2-1, and machine 3-7 (machine 3-2 was excluded as there are no recorded anomalies in the test set). Since human traffic varies over the course of the day this dataset is periodic.

- *Secure Water Treatment* (SWaT): provided by iTrust, this dataset is taken from a scaled-down water treatment testbed (Goh, et al., 2016). We consider data collected

in December 2015, with 6 days of normal activity being used as a training set. The following day, when a number of cyber-attacks were initiated, was used as a test set.

- *Mars Science Laboratory* (MSL): this dataset contains sensor and actuator data from the mars rover. As has been previously established in (Nakamura, et al., 2020; Tuli et al., 2022), we only consider the non-trivial sequences A4, C2, and T1. The P1 sequence was withheld for hyperparameter optimisation.

3.6.2 Synthetic Datasets

In addition to the 11 real-world time-series, our evaluation was supplemented with 20 synthetic time-series. These time-series were generated using GutenTag, an extensible tool for the creation of anomalous time-series (Wenig, et al., 2022).

Synthetic data were grouped into 4 classes, each characterised by a different type of anomaly, 5 time-series were generated per class. We used sine waves as the base oscillations for each time-series with 20% of the dimensions in a given time-series being affected by each anomaly. The underlying characteristics (time-series length, anomaly length, dimensionality, base oscillation frequencies, and anomaly magnitude) of each time-series within a class was randomly sampled without replacement from a range of possible values, full details of the selected values can be seen in Appendices Table 5. More information on the four synthetic anomaly classes is provided below.

- *Point Anomalies*: 5 extremum anomalies were injected into the test set at separate time points. It should be noted that each extremum anomaly only lasts a single time-step. These synthetic anomalies can be considered point anomalies.
- *Variance Anomalies*: A single extended anomaly, characterised by an increase in the variance of 20% of the sine waves, is inserted into the test set. These synthetic anomalies can be considered context anomalies.
- *Frequency Anomalies*: A single extended anomaly, characterised by an increase in the frequency of 20% of the waves, is inserted into the test set. These synthetic anomalies can be considered context anomalies.
- *Mean-Shift Anomalies*: A single extended anomaly, characterised by a shift in the mean of 20% of the waves, is inserted into the test set. These synthetic anomalies can be considered collective anomalies.

3.7 Evaluation Metrics

By only selecting time-series with labelled anomalies in their test sets, we are able to compare models' predictions to ground truth labels. Despite the interest in the field, there is a lack of consensus on the most appropriate metrics for evaluating anomaly detection algorithms (Sørbø & Ruocco, 2023). Most papers tend to report area under the receiver operating curve (ROC AUC) and area under the precision recall curve (PR AUC) (Schmidl et al., 2022; Tuli et al., 2022), even though there is evidence that ROC AUC is poorly suited to imbalanced datasets (Sørbø & Ruocco, 2023). For the sake of comparison, we chose to include both PR AUC and ROC AUC.

We use Autorank, a statistical tool that facilitates the comparison between paired populations (Herbold, 2020), to rank the performance of different models across a number of time-series.

3.8 Benchmark Models

Model	Method	Deep Learning
TranAD	Reconstruction	True
GDN	Reconstruction	True
OmniAnomaly	Reconstruction	True
LSTM	Reconstruction	True
Isolation Forest	Isolation Tree	False
Kmeans	Distance	False
COPOD	Density Estimation	False
LODA	Density Estimation	False

Table 2: Benchmark models

Table 2 provides the full list of models used to benchmark conDENSE's performance. Due to its similarity with conDENSE's GRU VAE component, we implemented the LSTM VAE from scratch. We used publicly available code, provided by the original developers, for TranAD and GDN[1]. All other benchmark models were implemented using TimeEval, an open source evaluation tool for anomaly detection algorithms (Wenig et al., 2022). It should be noted that model performance on a given dataset will differ from previously reported values if different sequences are used for evaluation.

3.9 conDENSE Hyperparameters

Hyperparameter	Value
MAF Bijectors	5
MAF Hidden Units	[64, 64]
Window Size	10
GRU Hidden Units	5
Latent Dimensions	20
VAE Hidden Units	[256, 128]
POT lm	0.993 or 0.97
β_r (see Equation 3)	0.1

Table 3: conDENSE hyperparameters

1. TranAD implementation: https://github.com/imperial-qore/TranAD.
GDN implementation: https://github.com/d-ailin/GDN

Our models were implemented using TensorFlow and optimised using the Adam algorithm with a learning rate of 0.001, and a batch size of 128 (these were the suggested default values). Models were trained for up to 50 epochs, although an early stopping rule terminated training early if performance had not improved over three consecutive epochs. In order to do this, 20% of training data was withheld for use as a validation set.

Model specific hyperparameters are shown in Table 3. conDENSE hyperparameters were optimized using the MSL P1 time-series, with respect to ROC AUC, performance on this dataset is not included in our results. With the exception of POT lm, all conDENSE hyperparameters are shared across all datasets. We use a higher POT lm coefficient for the univariate time-series dataset, in keeping with previous work (Tuli et al., 2022; Su et al., 2019).

4. Results

Critical difference diagrams comparing the performances of conDENSE and the benchmark models are presented in Figure 2 (Demšar, 2006). These diagrams visualise the relative performance of each model across a group of time-series. Models are ordered along the top axis, which reflects their mean rank for a given metric. Models that are connected by a bold line below this axis were not found to be significantly different. Statistical significance was established using: the non-parametric Friedman test, to determine if there are any significant differences between the mean values of populations, followed by the post-hoc Nemenyi test, to infer which differences are significant (Demšar, 2006). For detailed data on the performance of each model on all of the 31 time-series see Appendices Table 6.

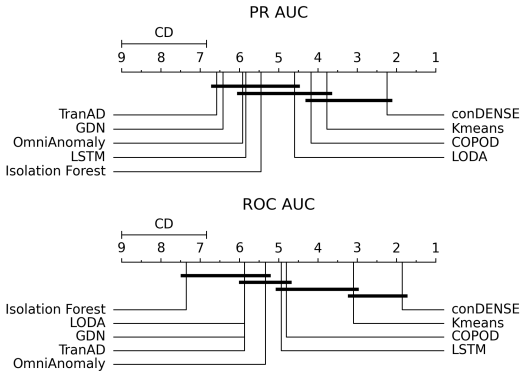
conDENSE ranks well compared to the benchmark models across all datasets. It has the lowest mean rank for both PR AUC and ROC AUC, it is significantly better than all models, with the exception of Kmeans and COPOD. Since ROC AUC and PR AUC scores consider ranked prediction probabilities, rather than binary class labels, conDENSE's strong performance on these metrics demonstrates that it produces scores that are better ordered than the benchmark models'.

conDENSE's performance is also encouraging on real-world datasets, producing the best PR AUC mean rank and the second best ROC AUC mean rank. It should be noted that for both metrics these ranks were not found to be significantly different from most benchmark models.

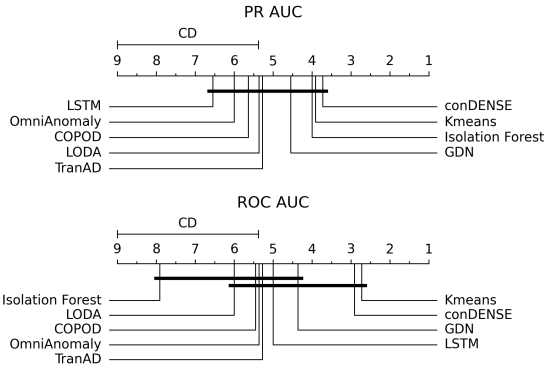
Results are strong across the synthetic anomaly datasets, with conDENSE producing the best mean rank for both metrics in each case, although it was found to be statistically similar to a number of the other benchmark models. It is harder to establish statistical significance for each of the synthetic datasets as models are only being compared on 5 separate time series.

Overall, these results show that conDENSE can compete with state-of-the-art models when detecting anomalies in a range of contexts.

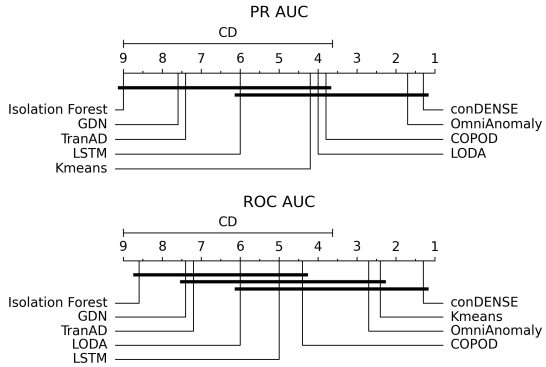
(a) All Datasets



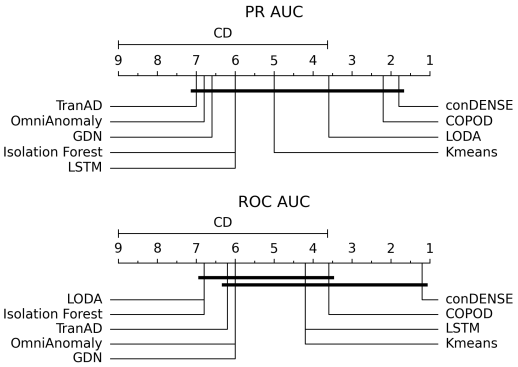
(b) Real-World Datasets



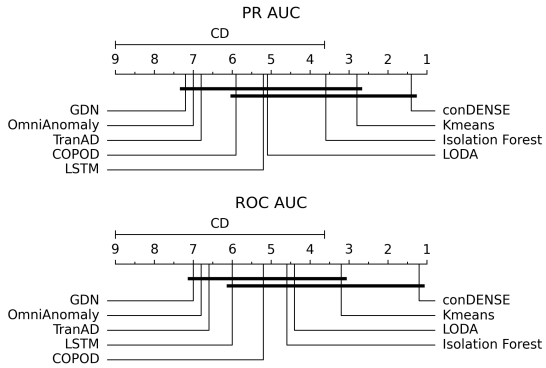
(c) Synthetic Point Anomalies



(d) Synthetic Variance Anomalies



(e) Synthetic Frequency Anomalies



(f) Synthetic Mean-shift Anomalies

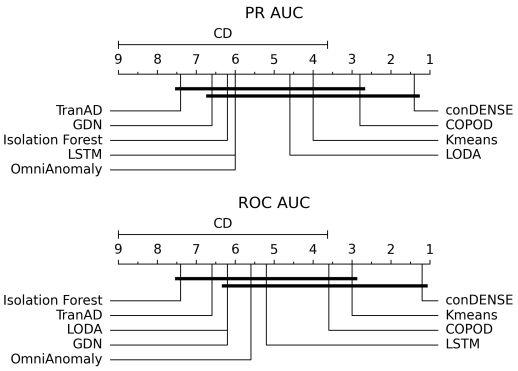


Figure 2: Critical difference diagrams comparing the performance of conDENSE to benchmark models. PR AUC and ROC AUC are compared for different groups of time-series.

4.1 Ablation

In order to assess the impact of our primary contribution, conditioning a MAF on a dense representation of the current trajectory, we performed an ablation analysis. In this analysis we compared the performance of conDENSE to an unconditional MAF.

Statistical analysis shows that conDENSE produces a significantly better PR AUC than an unconditional MAF across all datasets ($p = 0.001 < 0.05$**, Wilcoxon's signed rank test). Similar tests found a significant performance uplift for conDENSE on synthetic frequency anomalies ($p = 0.003 < 0.05$**, Wilcoxon's signed rank test) and a marginal uplift for synthetic variance anomalies ($p = 0.054 < 0.1$**, Wilcoxon's signed rank test). Interestingly, there was no significant difference between an unconditional MAF and conDENSE for synthetic point anomalies ($p = 0.358 > 0.05$, Wilcoxon's signed rank test) and synthetic mean-shift anomalies ($p = 0.203 > 0.05$, Wilcoxon's signed rank test).

Detailed performance breakdowns of an unconditional MAF for each time-series is included in Appendices 6. The unconditional MAF compares favourably to a number of the benchmark models on three of the real-world datasets: MSL, SWaT, and SMD. However, unlike conDENSE, it is totally unable to catch any of the context anomalies in the highly periodic UCR dataset.

These results suggest that the primary advantage of conDENSE over unconditional MAFs is its ability to handle context anomalies.

4.2 Training Time

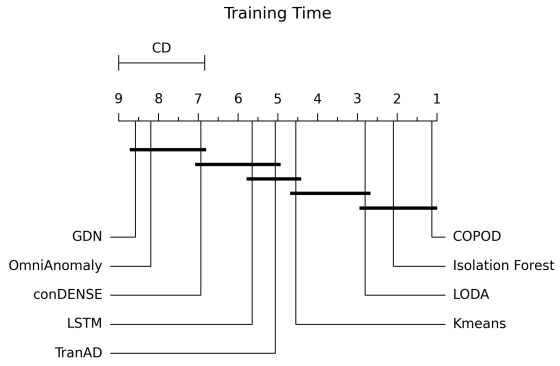


Figure 3: Critical difference diagram comparing the training times of conDENSE and the benchmark models across all time-series.

Figure 3 is a critical difference diagram comparing the training times of conDENSE and the benchmark models. There is no statistical difference between conDENSE and the other deep learning algorithms, although it is found to be significantly slower than the traditional models.

In (Tuli et al., 2022) the training time of TranAD is compared to a large range of other deep learning approaches and it is found to be significantly faster. Table 4, which compares

the training times of deep learning models, reinforces this finding. This table shows how, despite not being the fastest, conDENSE avoids the long training times associated with certain deep learning methods, such as GDN. See Appendices 6 for training times for each individual time-series.

	Training Time (s)		
	Mean (s)	Median (s)	Standard Deviation (s)
TranAD	7.02	3.65	11.06
LSTM	4.75	3.98	2.12
conDENSE	10.20	7.36	7.41
OmniAnomaly	47.18	35.86	43.18
GDN	83.29	45.89	113.10

Table 4: Table comparing the training time of deep learning algorithms.

4.3 Interpretability

Since conDENSE involves direct density estimation, unlike other reconstruction error based approaches, it is possible to visualise the PDF of expected values at a given point in a time series. This greatly increases interpretability. Figure 4a plots the evolution of expected values over a single period, since the observed values remains in the centre of the distribution of the expected values, no anomaly would be flagged for this period. The narrower ranges of expected values in some sections of Figure 4a reflect increased certainty about how conDENSE believes the system will behave. It is unsurprising that uncertainty increases at key inflection points.

The ability of the GRU VAE to capture the relative position in a period is highlighted in Figure 4b. The distribution of expected values for two different data points are plotted. Despite having similar values, these two observations come from different phases of the period. Around observation A the sensor values are rapidly increasing, on the other hand, the sensor values are decreasing around observation B. The position of both PDFs relative to their previous observations captures these dynamics. The PDF of expected values for A shows conDENSE predicts the next observed value to be greater than the previous value. The opposite is true for B, where conDENSE predicts that the next observation will be smaller than the previous value. One can also infer, by calculating the distance between the previous and the most likely value, that conDENSE expects the graph at point A to increase more quickly than it is decreasing at point B.

Figure 4c plots the PDF of expected values over an anomalous period. Several observations fall outside of the expected range and are flagged as anomalies. The expected distribution and observed value for one anomaly are shown in Figure 4d. An interesting observation from 4c is that expected values are strongly impacted by recent instability in the time series. This is particularly obvious for the cluster of anomalies near point Y, where expected values jump around erratically. Contrasting this section of the time series with the smooth evolution of expected values seen in Figure 4a further highlights the effect.

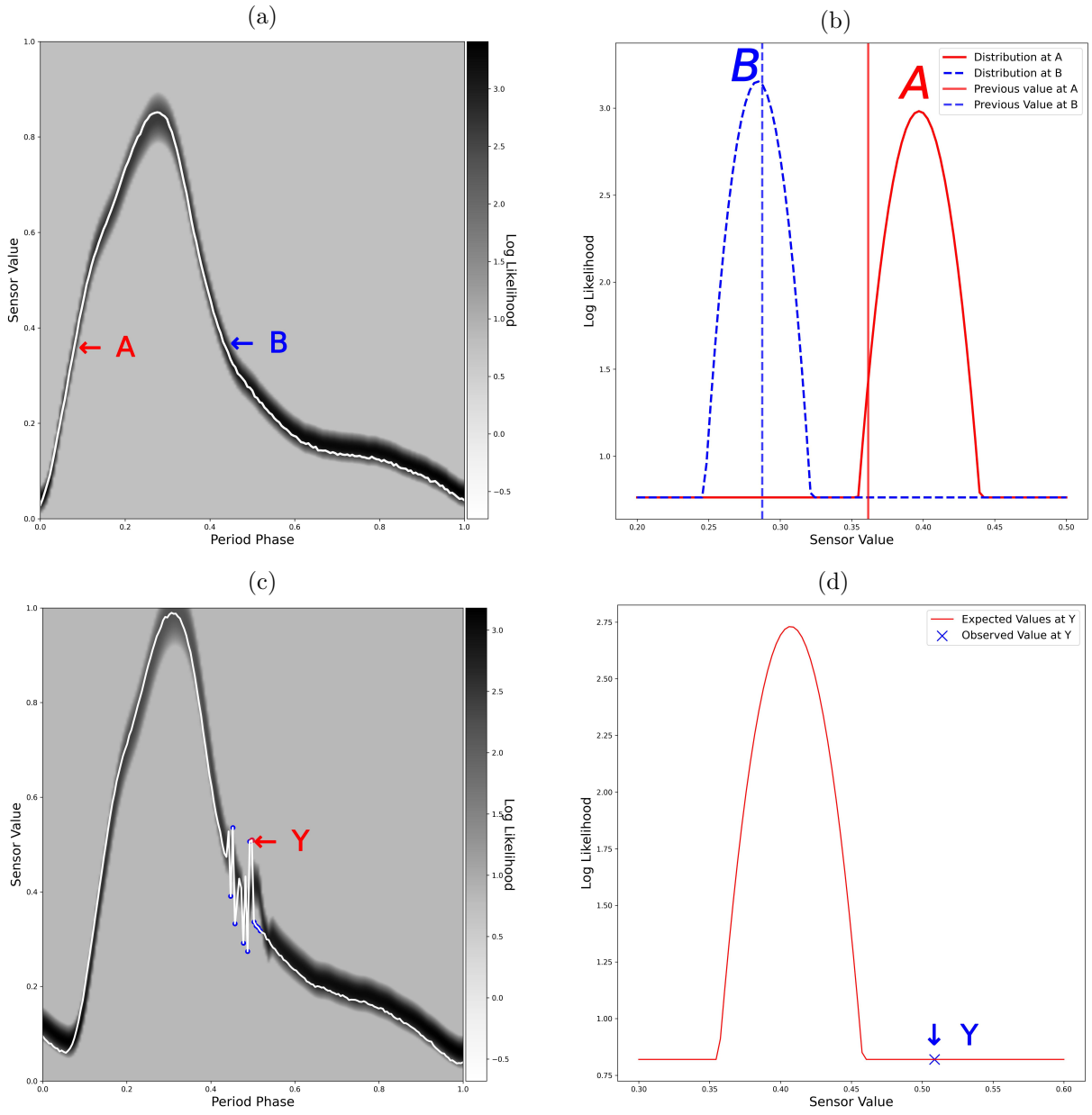


Figure 4: Interpreting conDENSE predictions on the UCR dataset. 4a shows how the PDF of expected values changes over the course of a single non-anomalous period. The solid line plots the observed values at a given time and the heat map shows the PDF of expected values at that time. All values that would be flagged as anomalies are clipped to the same minimum value. 4b plots the distribution of expected values at the time points A and B, marked in 4a. The vertical lines show the observed values at the previous time point for A and B. 4c summarises the change of expected values over a single anomalous period. Any point that would be flagged as anomalous is marked with a circle. 4d plots the distribution of non-anomalous expected values at the point Y marked in 4c. The observed value at time Y is marked with a blue cross.

5. Discussion

In this paper we show that conDENSE can achieve state-of-the-art anomaly detection performance on a range of periodic and non-periodic datasets. It produces results that are statistically better than all bar one of the benchmark models across 31 time-series and it is consistently ranked top for each anomaly group. The model can also handle high-dimensional multivariate datasets. These findings, coupled with a relatively short training time, suggest that this novel approach could be suitable for a large number of real-world anomaly detection problems.

conDENSE achieves state-of-the-art performance by directly estimating the likelihood of observations; it does not rely on a point estimate of expected behaviour but rather learns a PDF of expected values. This is an important conceptual step, relative to reconstruction error algorithms. Relying on a single prediction error is likely to cause problems when detecting anomalies in complex multi-modal datasets or in systems experiencing unpredictable level shifts. Direct density estimation can also improve interpretability, as is shown in Section 4.3. Moreover, the probability densities produced by conDENSE could easily be combined with other distributions in a Bayesian framework. This ability to go beyond anomaly detection and make inferences would not be possible with a reconstruction error model.

Our results also align with recent suggestions that deep learning anomaly detection models do not always justify their increased computational footprint (Schmidl et al., 2022; Rewicki et al., 2023). Although conDENSE compared favourably, other deep learning models often failed to outperform light-weight traditional benchmark models, with Kmeans ranking consistently higher than the benchmark deep learning models.

In many cases, an unconditional MAF is a strong candidate model for anomaly detection tasks. Our ablation study shows that, in certain contexts, it can outperform other state-of-the-art deep learning algorithms. It appears that it is well suited to detecting collective and point anomalies, as shown by its strong performance on the synthetic point and mean-shift time-series as well as the MSL, SMD, and SWaT real-world datasets. However, our ablation study also shows that an unconditional MAF is totally unsuited to detecting context anomalies. It performs poorly on the synthetic frequency and synthetic variance anomalies and is unable to catch outliers in the UCR dataset. An unconditional MAF, with no access to data on relative timings, is unable to detect outliers that fall within the previously observed range of values.

Including the GRU VAE in conDENSE allows it to handle context anomalies, its ability to locate when in a period an observation occurs is highlighted in Figure 4. Moreover, conDENSE outperforms an unconditional MAF when applied to different anomaly types. This implies that the GRU VAE is doing more than simply locating where in a period an observation occurs. Figure 4c offers a potential explanation for this effect. In the figure we can see how recent instability in the time series produces a more erratic distribution of expected values at a particular time point. This points towards the GRU VAE capturing other dynamics, such as smoothness, from the window preceding an observation, which may play a crucial role in the detection of anomalies.

One of the main strengths of our model is it can naturally be extended to incorporate other ideas from time-series analysis or deep learning. For example, when monitoring a

stochastic process one may choose to condition on the first and second moments, instead of using the GRU VAE's dense representation. On the other hand, in some cases it may be necessary to incorporate longer term trends or domain knowledge into the conditioning variable. Ultimately, the most appropriate conditional variable will change dependent on the dataset. With this in mind, conDENSE should be viewed as a flexible and generalisable framework that can be tweaked to suit a particular context, rather than a fixed model architecture.

It is important to note the majority of our analysis has focused on threshold-agnostic metrics, namely ROC AUC and PR AUC. Although these metrics provide useful insight when comparing models, real-world anomaly detection systems tend to operate with defined anomaly thresholds. Where exactly this threshold should be drawn will vary in different contexts, as the relative cost of false positives and false negatives is not fixed. Selecting a threshold is a trade-off and it is typically difficult to find one that can simultaneously maximise both precision and recall. Low PR AUC scores, which are found in much of the literature (Schmidl et al., 2022), do not imply that a model cannot function as an effective anomaly detector. In reality, the optimal anomaly threshold is determined by the intended use of the anomaly detection system.

In this paper we have demonstrated that conDENSE (a conditional MAF) is well suited for anomaly detection. We have shown that using a dense representation of the current trajectory, produced by a GRU VAE, as the conditional variable can significantly improve performance by capturing short-term dynamics. Importantly, as a density estimation algorithm, conDENSE does not rely on the limiting assumptions made by reconstruction error models.

6. Future Work

One shortcoming of our model is that it requires a separate labelled dataset for hyperparameter optimisation. Our goal was to find a general set of hyperparameters that would be suitable for a wide range of datasets. It is likely that the optimal parameters will vary significantly between different datasets and a more tailored approach that doesn't require labelled anomalies could improve performance. There are a number of potential solutions to this: 1) a function that maps certain time series characteristics, such as dimensionality and periodicity, to hyperparameter values might be viable, 2) hyperparameters could be optimised with respect to an unsupervised cost function, minimizing the loss function in Equation 3 on a withheld section of the training data is one option.

Another extension would be adapting conDENSE for unevenly sampled data. Our work has assumed a fixed sampling rate, which could be limiting in many real world situations. Recent work, has explored using a Wiener process as the base distribution for normalising flows when modelling stochastic processes (Deng, et al., 2021). Incorporating these ideas into conDENSE could enable anomaly detection on unevenly sampled data.

Acknowledgments

Both Alex Moore and Davide Morelli were employed by Huma Therapeutics Ltd when this research was conducted. They would like to thank Bastien Orset, Farzin Dadashi and Rheeda Ali for their comments on this manuscript.

7. Appendices

7.1 Synthetic Time-Series Stats

Name	Dimensions	Anomaly Length	Series Length	Anomaly Magnitude
point-0	1	5	4000	NA
point-1	3	5	6000	NA
point-2	5	5	10000	NA
point-3	10	5	8000	NA
point-4	20	5	2000	NA
variance-0	1	100	12000	0.154
variance-1	3	25	6000	0.2
variance-2	5	50	4000	0.151
variance-3	10	75	10000	0.161
variance-4	20	200	8000	0.217
frequency-0	1	100	4000	0.5
frequency-1	3	200	12000	0.1
frequency-2	5	25	8000	1.5
frequency-3	10	50	10000	0.01
frequency-4	20	75	6000	2.0
mean-0	1	100	2000	0.15
mean-1	3	25	4000	0.5
mean-2	5	200	8000	0.1
mean-3	10	75	6000	0.75
mean-4	20	50	10000	0.25

Table 5: Statistics for synthetic time-series. N.B. point anomalies are of fixed magnitude and duration.

7.2 Detailed Results

	SWaT									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.666	0.736	0.732	0.819	0.613	0.410	0.747	0.629	0.661	0.800
PR AUC	0.402	0.331	0.347	0.502	0.275	0.294	0.443	0.297	0.180	0.571
Train Time (s)	161.136	52.208	14.952	11.079	0.046	0.065	4.527	0.496	7.749	7.786
	MSL A-4									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.707	0.738	0.644	0.583	0.494	0.500	0.449	0.511	0.629	0.720
PR AUC	0.021	0.025	0.018	0.015	0.058	0.431	0.013	0.065	0.017	0.023
Train Time (s)	24.175	15.218	2.697	6.233	0.225	0.057	3.130	0.319	3.808	3.281
	MSL C-2									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.513	0.510	0.569	0.624	0.630	0.499	0.634	0.525	0.473	0.612
PR AUC	0.063	0.062	0.161	0.132	0.303	0.491	0.204	0.404	0.057	0.156
Train Time (s)	21.704	4.439	1.432	5.606	0.004	0.053	2.595	0.212	3.197	2.765
	MSL T-1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.738	0.414	0.617	0.491	0.502	0.721	0.223	0.493	0.725	0.258
PR AUC	0.380	0.155	0.205	0.175	0.226	0.559	0.111	0.205	0.307	0.127
Train Time (s)	29.408	16.250	2.937	6.535	0.006	0.058	3.178	0.404	3.317	2.936
	SMD Machine 1-1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.581	0.856	0.909	0.938	0.668	0.205	0.915	0.799	0.596	0.950
PR AUC	0.328	0.419	0.473	0.568	0.392	0.103	0.543	0.585	0.323	0.635
Train Time (s)	459.685	168.595	39.585	35.330	0.138	0.099	5.617	0.547	6.479	16.421
	SMD Machine 2-1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.395	0.916	0.933	0.913	0.765	0.321	0.982	0.687	0.517	0.920
PR AUC	0.045	0.201	0.225	0.198	0.378	0.280	0.551	0.293	0.079	0.153
Train Time (s)	389.656	139.663	34.952	33.114	0.120	0.085	4.860	0.506	8.477	16.965
	SMD Machine 3-7									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.481	0.546	0.699	0.686	0.735	0.342	0.788	0.522	0.671	0.685
PR AUC	0.015	0.022	0.434	0.109	0.330	0.203	0.514	0.124	0.030	0.090
Train Time (s)	373.891	176.568	42.134	20.437	0.141	0.098	5.749	0.576	12.556	17.116
	UCR 135									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.481	0.546	0.699	0.686	0.735	0.342	0.788	0.522	0.671	0.330
PR AUC	0.015	0.022	0.434	0.109	0.330	0.203	0.514	0.124	0.030	0.001
Train Time (s)	373.891	176.568	42.134	20.437	0.141	0.098	5.749	0.576	12.556	2.486
	UCR 136									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.891	0.496	0.368	0.888	0.494	0.398	0.971	0.479	0.633	0.496
PR AUC	0.609	0.019	0.032	0.697	0.062	0.183	0.581	0.041	0.024	0.019
Train Time (s)	11.660	9.676	0.541	3.939	0.001	0.056	1.725	0.010	2.949	2.236
	UCR 137									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.680	0.637	0.285	0.836	0.448	0.399	0.727	0.480	0.594	0.468
PR AUC	0.111	0.025	0.012	0.417	0.010	0.204	0.038	0.010	0.024	0.018
Train Time (s)	16.720	13.773	0.761	5.627	0.001	0.056	0.908	0.012	3.538	2.767

	UCR 138									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.894	0.675	0.233	0.966	0.449	0.247	0.987	0.452	0.785	0.688
PR AUC	0.399	0.004	0.001	0.841	0.001	0.001	0.194	0.001	0.005	0.004
Train Time (s)	21.752	17.802	1.014	5.799	0.001	0.057	0.526	0.015	3.904	2.953
	Synthetic Point 0									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.756	0.918	0.661	1.000	0.850	0.319	0.997	0.500	0.700	0.840
PR AUC	0.004	0.708	0.004	1.000	0.405	0.001	0.355	0.501	0.064	0.800
Train Time (s)	30.034	22.921	1.139	4.425	0.001	0.058	0.973	0.019	2.688	1.893
	Synthetic Point 1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.609	1.00	0.639	1.000	0.950	0.486	0.999	0.751	0.993	1.000
PR AUC	0.002	1.00	0.002	0.906	0.504	0.000	0.580	0.303	0.293	1.000
Train Time (s)	45.892	35.21	2.066	8.548	0.004	0.062	2.698	0.436	4.026	3.211
	Synthetic Point 2									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.717	0.987	0.770	0.994	0.948	0.474	0.996	0.851	0.784	0.999
PR AUC	0.001	0.801	0.001	0.803	0.502	0.000	0.346	0.402	0.058	0.820
Train Time (s)	75.691	59.632	4.116	15.292	0.010	0.069	3.282	0.582	4.829	5.567
	Synthetic Point 3									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.504	1.000	0.561	1.000	0.951	0.482	0.982	0.851	0.907	1.000
PR AUC	0.002	0.918	0.001	1.000	0.503	0.000	0.572	0.403	0.089	1.000
Train Time (s)	59.954	48.068	4.534	10.536	0.013	0.064	3.312	0.403	5.549	4.555
	Synthetic Point 4									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.322	0.977	0.327	1.000	0.844	0.492	0.971	0.849	0.963	1.000
PR AUC	0.002	0.804	0.002	0.963	0.409	0.001	0.298	0.410	0.055	1.000
Train Time (s)	16.748	12.567	1.685	6.984	0.007	0.057	2.933	0.304	3.991	3.025
	Synthetic Variance 0									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.503	0.490	0.497	0.804	0.536	0.492	0.998	0.489	0.597	0.511
PR AUC	0.070	0.070	0.069	0.313	0.096	0.032	0.948	0.032	0.118	0.097
Train Time (s)	87.208	72.027	3.716	5.184	0.003	0.070	2.881	0.516	3.708	2.290
	Synthetic Variance 1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.454	0.422	0.409	0.706	0.569	0.499	0.497	0.530	0.510	0.679
PR AUC	0.004	0.004	0.004	0.192	0.126	0.002	0.004	0.085	0.015	0.019
Train Time (s)	43.666	35.857	2.457	7.095	0.005	0.061	3.094	0.320	5.011	3.142
	Synthetic Variance 2									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.592	0.612	0.617	0.823	0.602	0.521	0.488	0.501	0.571	0.457
PR AUC	0.018	0.021	0.020	0.287	0.174	0.057	0.013	0.062	0.016	0.012
Train Time (s)	29.343	23.910	2.023	5.808	0.004	0.062	2.827	0.247	3.001	3.079
	Synthetic Variance 3									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.562	0.544	0.582	0.781	0.555	0.503	0.667	0.496	0.538	0.713
PR AUC	0.009	0.009	0.009	0.129	0.117	0.022	0.014	0.054	0.008	0.125
Train Time (s)	74.488	59.698	6.407	11.567	0.016	0.070	3.314	0.413	5.030	5.944
	Synthetic Variance 4									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.520	0.500	0.529	0.678	0.568	0.511	0.610	0.523	0.568	0.570
PR AUC	0.028	0.027	0.030	0.072	0.156	0.071	0.143	0.101	0.047	0.037
Train Time (s)	69.666	48.232	7.390	7.361	0.024	0.064	3.940	0.359	3.787	4.919

	Synthetic Frequency 0									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.378	0.467	0.470	0.976	0.501	0.495	0.998	0.495	0.464	0.509
PR AUC	0.021	0.024	0.034	0.763	0.074	0.083	0.820	0.053	0.244	0.033
Train Time (s)	28.873	23.822	1.266	4.507	0.002	0.059	2.277	0.020	2.608	1.845
	Synthetic Frequency 1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.431	0.353	0.428	0.990	0.468	0.456	0.786	0.524	0.584	0.428
PR AUC	0.015	0.012	0.014	0.904	0.031	0.054	0.594	0.092	0.156	0.014
Train Time (s)	86.699	72.999	5.116	9.998	0.007	0.071	3.024	0.315	3.976	3.621
	Synthetic Frequency 2									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.421	0.369	0.386	0.939	0.489	0.499	0.912	0.492	0.461	0.533
PR AUC	0.003	0.002	0.002	0.637	0.043	0.043	0.101	0.043	0.003	0.003
Train Time (s)	58.031	49.896	3.654	10.715	0.007	0.064	2.762	0.348	4.369	3.744
	Synthetic Frequency 3									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.007	0.001	0.004	0.992	0.454	0.950	0.000	0.452	0.001	0.638
PR AUC	0.003	0.003	0.003	0.348	0.002	0.524	0.003	0.002	0.003	0.006
Train Time (s)	74.037	60.582	6.224	13.129	0.017	0.068	3.147	0.240	7.377	6.356
	Synthetic Frequency 4									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.521	0.504	0.524	1.000	0.492	0.504	1.000	0.505	0.482	0.496
PR AUC	0.013	0.012	0.013	0.979	0.058	0.028	0.978	0.066	0.011	0.012
Train Time (s)	50.893	36.405	5.574	11.709	0.019	0.062	3.589	0.744	4.350	4.748
	Synthetic Mean-Shift 0									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.501	0.534	0.493	0.880	0.552	0.499	0.970	0.490	0.535	0.516
PR AUC	0.047	0.049	0.045	0.535	0.170	0.154	0.901	0.067	0.062	0.148
Train Time (s)	14.498	12.225	0.582	6.036	0.002	0.056	2.419	0.012	3.181	2.303
	Synthetic Mean-Shift 1									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.521	0.357	0.446	0.823	0.571	0.486	0.608	0.570	0.605	0.649
PR AUC	0.014	0.004	0.005	0.134	0.130	0.025	0.011	0.130	0.008	0.154
Train Time (s)	29.196	24.162	1.658	6.365	0.003	0.059	3.210	0.361	3.290	2.785
	Synthetic Mean-Shift 2									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.489	0.427	0.477	0.860	0.571	0.497	0.488	0.530	0.476	0.641
PR AUC	0.024	0.020	0.022	0.281	0.159	0.068	0.030	0.108	0.036	0.089
Train Time (s)	58.073	47.524	3.670	9.051	0.007	0.064	3.123	0.408	3.757	3.430
	Synthetic Mean-Shift 3									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.624	0.738	0.684	0.968	0.816	0.488	0.934	0.749	0.830	0.948
PR AUC	0.160	0.234	0.126	0.934	0.409	0.015	0.844	0.337	0.599	0.898
Train Time (s)	44.615	35.674	3.720	9.272	0.011	0.063	2.611	0.404	6.427	4.389
	Synthetic Mean-Shift 4									
	GDN	OmniAnomaly	TranAD	condense	copod	iso_f	kmeans	loda	lstm	MAF
ROC AUC	0.533	0.545	0.533	0.937	0.709	0.492	0.751	0.499	0.690	0.962
PR AUC	0.006	0.006	0.005	0.813	0.274	0.002	0.194	0.055	0.015	0.821
Train Time (s)	86.028	59.750	9.111	13.882	0.033	0.070	4.035	0.417	6.802	6.420

Table 6: Detailed model performance for each time-series.

References

Ahdesmäki, M., Lähdesmäki, H., Pearson, R., Huttunen, H., & Yli-Harja, O. (2005). Robust detection of periodic time series measured from biological systems. *BMC Bioinformatics*, *6*(1), 117. DOI: 10.1186/1471-2105-6-117.

Alqurashi, S., Shirazi, H., & Ray, I. (2021). On the Performance of Isolation Forest and Multi Layer Perceptron for Anomaly Detection in Industrial Control Systems Networks. In *2021 8th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pp. 1–6. DOI: 10.1109/IOTSMS53705.2021.9704986.

Audibert, J., Guyard, F., Marti, S., & Zuluaga, M. (2020). *USAD: UnSupervised Anomaly Detection on Multivariate Time Series*. DOI: 10.1145/3394486.3403392, Pages: 3404.

Baldi, P. (2012). Autoencoders, Unsupervised Learning, and Deep Architectures. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pp. 37–49. JMLR Workshop and Conference Proceedings. https://proceedings.mlr.press/v27/baldi12a.html, ISSN: 1938-7228.

Benecki, P., Piechaczek, S., Kostrzewa, D., & Nalepa, J. (2021). Detecting anomalies in spacecraft telemetry using evolutionary thresholding and LSTMs. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '21, pp. 143–144, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3449726.3459411.

Boniol, P., Paparrizos, J., Palpanas, T., & Franklin, M. J. (2021). SAND: streaming subsequence anomaly detection. *Proceedings of the VLDB Endowment*, *14*(10), 1717–1729. DOI: 10.14778/3467861.3467863.

Breunig, M., Kriegel, H.-P., Ng, R., & Sander, J. (2000). *LOF: Identifying Density-Based Local Outliers.*, Vol. 29. DOI: 10.1145/342009.335388, Journal Abbreviation: ACM Sigmod Record Pages: 104 Publication Title: ACM Sigmod Record.

Chalapathy, R., & Chawla, S. (2019). Deep Learning for Anomaly Detection: A Survey.. DOI: 10.48550/arXiv.1901.03407, arXiv:1901.03407 [cs, stat].

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly Detection: A Survey. *ACM Comput. Surv.*, *41*. DOI: 10.1145/1541880.1541882.

Chatterjee, S., Sciarra, A., Dünnwald, M., Agrawal, S., Tummala, P., Setlur, D., Kalra, A., Jauhari, A., Oeltze-Jafra, S., Speck, O., & Nürnberger, A. (2021). *Unsupervised reconstruction based anomaly detection using a Variational Auto Encoder*.

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., & Keogh, E. (2019). The UCR Time Series Archive.. http://arxiv.org/abs/1810.07758, arXiv:1810.07758 [cs, stat].

Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, *7*(1), 1–30. http://jmlr.org/papers/v7/demsar06a.html.

Deng, A., & Hooi, B. (2021). *Graph Neural Network-Based Anomaly Detection in Multivariate Time Series*.

Deng, R., Chang, B., Brubaker, M. A., Mori, G., & Lehrmann, A. (2021). Modeling Continuous Stochastic Processes with Dynamic Normalizing Flows.. DOI: 10.48550/arXiv.2002.10516, arXiv:2002.10516 [cs, stat].

Germain, M., Gregor, K., Murray, I., & Larochelle, H. (2015). MADE: Masked Autoencoder for Distribution Estimation..

Goh, J., Adepu, S., Junejo, K., & Mathur, A. (2016). *A Dataset to Support Research in the Design of Secure Water Treatment Systems.*

Herbold, S. (2020). Autorank: A Python package for automated ranking of classifiers. *Journal of Open Source Software*, *5*(48), 2173. DOI: 10.21105/joss.02173.

Kingma, D., & Welling, M. (2013). Auto-Encoding Variational Bayes. *ICLR.*

Kingsbury, K., & Alvaro, P. (2020). Elle: inferring isolation anomalies from experimental observations. *Proceedings of the VLDB Endowment*, *14*(3), 268–280. DOI: 10.14778/3430915.3430918.

Kumar, M., Babaeizadeh, M., Erhan, D., Finn, C., Levine, S., Dinh, L., & Kingma, D. (2019). *VideoFlow: A Flow-Based Generative Model for Video.*

Li, D., Chen, D., Jin, B., Shi, L., Goh, J., & Ng, S.-K. (2019). MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. In *Artificial Neural Networks and Machine Learning – ICANN 2019: Text and Time Series: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part IV*, pp. 703–716, Berlin, Heidelberg. Springer-Verlag. DOI: 10.1007/978-3-030-30490-4$_5$6.

Li, Z., Zhao, Y., Botta, N., Ionescu, C., & Hu, X. (2020). COPOD: Copula-Based Outlier Detection. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 1118–1123. DOI: 10.1109/ICDM50108.2020.00135, arXiv:2009.09463 [cs, stat].

Lin, S., Clark, R., Birke, R., Schönborn, S., Trigoni, N., & Roberts, S. (2020). Anomaly Detection for Time Series Using VAE-LSTM Hybrid Model. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4322–4326. DOI: 10.1109/ICASSP40776.2020.9053558, ISSN: 2379-190X.

Nakamura, T., Imamura, M., Mercer, R., & Keogh, E. (2020). MERLIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives. *ICDM 2020.* DOI: 10.1109/ICDM50108.2020.00147.

Pang, G., Shen, C., Cao, L., & Hengel, A. v. d. (2020). Deep Learning for Anomaly Detection: A Review.. DOI: 10.1145/3439950, arXiv:2007.02500 [cs, stat].

Papamakarios, G., Pavlakou, T., & Murray, I. (2018). Masked Autoregressive Flow for Density Estimation.. DOI: 10.48550/arXiv.1705.07057, arXiv:1705.07057 [cs, stat].

Pevný, T. (2016). Loda: Lightweight on-line detector of anomalies. *Machine Learning*, *102*(2), 275–304. DOI: 10.1007/s10994-015-5521-0.

Rewicki, F., Denzler, J., & Niebling, J. (2023). Is It Worth It? Comparing Six Deep and Classical Methods for Unsupervised Anomaly Detection in Time Series. *Applied Sciences*, *13*(3), 1778. DOI: 10.3390/app13031778, Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., & Müller, K.-R. (2021). A Unifying Review of Deep and Shallow Anomaly Detection. *Proceedings of the IEEE*, *109*(5), 756–795. DOI: 10.1109/JPROC.2021.3052449, Conference Name: Proceedings of the IEEE.

Schmidl, S., Wenig, P., & Papenbrock, T. (2022). Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, *15*(9), 1779–1797. DOI: 10.14778/3538598.3538602.

Schmidt, M., & Simic, M. (2019). Normalizing flows for novelty detection in industrial time series data.. DOI: 10.48550/arXiv.1906.06904, arXiv:1906.06904 [cs, stat].

Shchur, O., Biloš, M., & Günnemann, S. (2019). *Intensity-Free Learning of Temporal Point Processes*.

Siffer, A., Fouque, P.-A., Termier, A., & Largouet, C. (2017). Anomaly Detection in Streams with Extreme Value Theory. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pp. 1067–1075, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3097983.3098144.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., & Pei, D. (2019). *Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network*. DOI: 10.1145/3292500.3330672, Pages: 2837.

Sun, J., Wang, X., Xiong, N., & Shao, J. (2018). Learning Sparse Representation With Variational Auto-Encoder for Anomaly Detection. *IEEE Access*, *6*, 33353–33361. DOI: 10.1109/ACCESS.2018.2848210, Conference Name: IEEE Access.

Sørbø, S., & Ruocco, M. (2023). Navigating the Metric Maze: A Taxonomy of Evaluation Metrics for Anomaly Detection in Time Series.. http://arxiv.org/abs/2303.01272, arXiv:2303.01272 [cs].

Tuli, S., Casale, G., & Jennings, N. (2022). TranAD: deep transformer networks for anomaly detection in multivariate time series data. *Proceedings of the VLDB Endowment*, *15*, 1201–1214. DOI: 10.14778/3514061.3514067.

Tuor, A., Kaplan, S., Hutchinson, B., Nichols, N., & Robinson, S. (2017). Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data Streams.. http://arxiv.org/abs/1710.00811, arXiv:1710.00811 [cs, stat].

Ushakova, A., Taylor, S., & Killick, R. (2021). *Multi-level Changepoint Inference for Periodic Data Sequences*. DOI: 10.13140/RG.2.2.35838.51526.

Wang, H., Bah, M. J., & Hammad, M. (2019). Progress in Outlier Detection Techniques: A Survey. *IEEE Access*, *7*, 107964–108000. DOI: 10.1109/ACCESS.2019.2932769, Conference Name: IEEE Access.

Wenig, P., Schmidl, S., & Papenbrock, T. (2022). TimeEval: a benchmarking toolkit for time series anomaly detection algorithms. *Proceedings of the VLDB Endowment*, *15*(12), 3678–3681. DOI: 10.14778/3554821.3554873.

Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., & Chawla, N. V. (2019). A Deep Neural Network for Unsupervised Anomaly Detection

and Diagnosis in Multivariate Time Series Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 1409–1416. DOI: 10.1609/aaai.v33i01.33011409, Number: 01.

Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., Xu, B., Bai, J., Tong, J., & Zhang, Q. (2020). Multivariate Time-series Anomaly Detection via Graph Attention Network.. DOI: 10.48550/arXiv.2009.02040, arXiv:2009.02040 [cs, stat].

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., & Chen, H. (2022). Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection.. https://openreview.net/forum?id=BJJLHbb0-.