Time Series Anomaly Detection using Prediction-Reconstruction Mixture Errors

by

Lawrence C. Wong

S.B. Computer Science and Molecular Biology
Massachusetts Institute of Technology, 2020

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Master of Engineering in Computer Science and Molecular Biology

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022

Author .....
Department of Electrical Engineering and Computer Science
May 6, 2022

Certified by .....
Kalyan Veeramachaneni
Principal Research Scientist
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Time Series Anomaly Detection using Prediction-Reconstruction Mixture Errors

by

Lawrence C. Wong

Submitted to the Department of Electrical Engineering and Computer Science
on May 6, 2022, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science and Molecular Biology

Abstract

Anomaly detection on time series data is increasingly common across various industrial domains that require monitoring metrics to prevent potential accidents and economic losses. The complications of anomaly detection revolve around a scarcity of labeled data and the need to learn temporal correlations between multiple variables. Most successful unsupervised methods either use single-timestamp prediction or reconstruct entire time series. However, these methods are not mutually exclusive and can each offer complementary perspectives. This work first explores the successes and limitations of prediction-based and reconstruction-based methods. Next, it compares the effect of attention-based architectures with LSTM-based architectures on existing models. Finally, this research proposes a novel autoencoder architecture capable of producing bi-directional predictions while simultaneously reconstructing the original time series by optimizing a joint objective function. An ablation study using a mixture of prediction and reconstruction errors demonstrates that this simple architecture outperforms other state-of-the-art models for anomaly detection on both univariate and multivariate time series.

Thesis Supervisor: Kalyan Veeramachaneni
Title: Principal Research Scientist

Acknowledgments

I want to express my deepest gratitude to my thesis advisor Dr. Kalyan Veeramachaneni from the Laboratory for Information and Decision Systems (LIDS) at MIT for his guidance throughout this project. Without his ongoing support and academic advice, this research project and my level of personal growth would not have been possible.

I also wish to acknowledge the following individuals for their input and help in my research project. Thank you to Dr. Dongyu Liu, a postdoctoral researcher at the Data-to-AI (DAI) group, for being a fantastic advisor and providing feedback on this research project. Thank you to Sarah Alnegheimish, M.S. candidate at the DAI group, for her input on the implementation details of this project. I want to thank everyone in the DAI group for being wonderful lab mates.

Finally, I would like to thank my family and friends for always being there. Thank you for the sanity checks and ensuring I stay both healthy and happy in stressful times.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Time series data is consistently generated and collected across various industries – examples include stock price in finance, heart rate in biomedicine, and retail sales in business. Effective monitoring, utilization, and analysis of time series data often increases efficiency and productivity. Anomaly detection is a subset of time series analysis that aims to identify unexpected events. Since early anomaly detection is crucial to prevent issues like bank fraud, medical problems, and structural defects, this form of research is broadly and increasingly relevant.

1.2 Background

1.2.1 Common Characteristics of Anomalies in Time Series

While criteria for anomalies in time series differ across domains, there are common identifiable patterns. Point anomalies are singular data points that suddenly deviate from the normal range of the series. Contextual anomalies are data points that lie within the normal range of the series but do not follow the expected patterns. Finally, collective anomalies are data points that are not anomalous individually but are when considered in the context of the entire dataset.

Choi et al. [6] further broke down the patterns exhibited by anomalies in time series. Assume that the baseline time series has symmetric amplitude and frequency over time. The “missing" anomalies are long segments of observations that are missing or default to one constant value. The “minor" anomalies are segments of observations with relatively small amplitudes. The “outlier" anomalies are the same as point anomalies. The “square" anomalies describe a sudden shift in oscillation patterns (e.g., from sine wave to square wave). The “trend" anomalies introduce a linear trend to the previously stationary time series. Finally, the “drift" anomalies introduce a random drift to the once stationary time series.

1.2.2 Properties of Time Series

Time series also exhibit unique properties like temporality, dimensionality, noise, and non-stationarity that complicate anomaly detection. First, temporality refers to the temporal correlations and dependencies between consecutive observations in a time series [12]. Second, dimensionality refers to the number of channels in each observation. Multivariate time series datasets with more than one channel have the added complexity of capturing both temporal dependencies and correlation between observations simultaneously [5]. Third, noise refers to the unwanted changes to signals during their capture, storage, transmission processing, or conversion [17]. Finally, non-stationarity refers to having statistical properties that change over time, like seasonality, concept drift, and change points.

1.3 Related Work

Existing methods for anomaly detection on time series are categorized into prediction-based and reconstruction-based approaches as summarized by Geiger et al. [7]. While these methods are often based on LSTM networks, alternative networks based on the attention mechanism provide a different perspective for learning hidden representations.

1.3.1 Prediction-based Approaches

Prediction-based methods learn to forecast future observations through previous patterns in the time series. An observation is anomalous when the predicted value deviates significantly from the actual value. Examples of prediction-based methods include Autoregressive Integrated Moving Average (ARIMA) [15] and Long Short Term Memory Recurrent Neural Network with Non-parametric Dynamic Thresholding (LSTM-NDT) forecasting models [9]. However, these models are sensitive to outliers and often fail to find contextual anomalies.

1.3.2 Reconstruction-based Approaches

Reconstruction-based methods learn a latent low-dimensional representation to reconstruct the original input. This method assumes the latent space prioritizes capturing common patterns within the dataset. Rare events like anomalies are not captured in the latent representation and are less likely to be accurately reconstructed. Examples of reconstruction-based methods include Principal Component Analysis (PCA) [16], LSTM Auto-Encoders (LSTM-AE) [8], and LSTM Variational Auto-Encoders (LSTM-VAE) [14]. PCA is limited to linear reconstructions, while other methods fail to leverage spatial-temporal correlation and other dependencies in multivariate settings. Generative Adversarial Networks (GAN) is another reconstruction-based method that strives to address these issues. For example, MAD-GAN [13] uses spatial-temporal correlation and other dependencies among multiple variables to capture non-linear latent interactions by considering the entire variable set concurrently. TadGAN [7] is a similar model trained with cycle consistency loss to address model instability issues and to allow for better reconstruction of time series data.

1.3.3 Attention Mechanisms

Transformers [18] are a viable alternative to LSTM networks. LSTM networks quickly lose information from earlier hidden states since they rely on sequential processing to store past information in one final hidden state. Instead, transformers use multi-

head attention to parallelize computation, avoid recursion, and increase tolerance in capturing long-term dependencies. Parallelization also has the added benefit of reducing training time and overcoming computation limitations faced by TadGAN and MAD-GAN with high-resolution inputs. For example, TransGAN [10] proposed a simple GAN structure built entirely of memory-efficient transformer blocks along with a training recipe to handle model instability issues. While TransGAN was developed for high-resolution image data, similar ideas can be adopted to improve the LSTM-based GAN models.

1.4 Outline

- Chapter 2 provides information about univariate and multivariate time series datasets used in this study. This chapter also states the problem definition and evaluation criteria of unsupervised anomaly detection in time series.

- Chapter 3 explores the successes and limitations of baseline prediction-based methods (ARIMA, LSTM-NDT) and reconstruction-based methods (LSTM-AE, LSTM-VAE, TadGAN).

- Chapter 4 proposes several ideas to leverage successes and overcome the limitations of baseline methods.

- Chapter 5 investigates the effect of replacing LSTM-based networks with attention-based networks on the performance of existing methods.

- Chapter 6 summarizes the study and discusses potential future works.

1.5 Contributions

- Documented four successes and six limitations of baseline methods with visualized examples across multiple datasets, as summarized in Chapter 6.

- Proposed the idea of masking anomaly scores to address false positive predictions and bi-directional regression to address missing forecasts at the start of the time series.

- Provided a simple autoencoder with regression (AER) framework that leverages the successes of prediction-based and reconstruction-based methods. Ablation studies show that AER achieved an average contextual F1 score of 0.7383, a 20% improvement compared to the baseline ARIMA model.

- Explained the instability of critic scores and the duality of the critic and generator sub-components of the TadGAN model.

Chapter 2

Experimental Setup

2.1 Data Description

The models are evaluated on both univariate and multivariate datasets spanning various domains to test generalizability and adaptability.

2.1.1 Data Sources

The National Aeronautics and Space Administration (NASA)[1] provided two spacecraft telemetry datasets: Soil Moisture Active Passage (SMAP) and Mars Science Laboratory (MSL), acquired from a satellite and a rover respectively [9]. The type and timing of the measurements have been anonymized. All telemetry values are min-max scaled to between `[-1,1]` according to the train split. Each measurement is accompanied by one-hot encoded information about commands sent or received by specific spacecraft modules in a given time window. MSL consists of 54 while SMAP consists of 24 of these one-hot encoded commands. MSL and SMAP can be treated as either univariate or multivariate datasets.

Yahoo Webscope Program[2] provided the S5 datasets, which consist of one dataset of real production traffic to Yahoo properties (A1) and three synthetic datasets (A2, A3, A4) with varying trend, noise, and pre-specified or random seasonalities. The A2

[1] `https://github.com/khundman/telemanom/`

[2] `http://research.yahoo.com/Academic_Relations/ydata-labeled-timeseries-anomalies-v1`

and A3 datasets only contain outliers inserted at random positions, while the A4 has outliers and change points. The interval between each observation is one hour for all signals from Yahoo. A1, A2, A3, and A4 are all univariate datasets.

Numenta Anomaly Benchmark (NAB)[3] provides time series datasets from various domains: artificialWithAnomaly (Art), realAdExchange (AdEx), realAWSCloudwatch (AWS), realTraffic (Traffic), realTweets (Tweets). Art consists of artificially generated signals with varying types of anomalies. AdEx is a dataset for online advertisement clicking rates, where the signals include cost-per-clicks (CPC) and cost per thousand impressions (CPM). AWS is a dataset for metrics collected by the AmazonCloudWatch service and consists of signals from CPU utilization, Network Bytes In, and Disk Read Bytes. Traffic is a dataset for real-time traffic data from the twin cities metro area in Minnesota collected by the Minnesota Department of Transportation. It includes specific sensors' signals from occupancy, speed, and travel time. Tweets is a dataset for Twitter mentions of large publicly-traded companies (e.g., Google and IBM), measured by the number of mentions for a given ticker symbol every 5 minutes. Art, AdEx, AWS, Traffic, and Tweets are univariate datasets.

The UCR time series anomaly archive [4] is an alternative to the NASA, Yahoo, and NAB datasets. Wu et al. [19] mentioned that these popular datasets suffer from one or more of the following four flaws: triviality, unrealistic anomaly density, mislabeled ground truth, and run-to-failure bias. Hence, the UCR time series anomaly archive was created as a standardized dataset to compare anomaly detection algorithms. UCR consists of one univariate dataset with 250 signals.

OmniAnomaly [5] collected data over five weeks from a large internet company and created the Server Machine Dataset (SMD). The dataset is already partitioned into train and test splits and consists of signals from 28 different machines. SMD is a multivariate dataset.

[3] https://numenta.com/machine-intelligence-technology/numenta-anomaly-benchmark/

[4] https://www.cs.ucr.edu/~eamonn/time_series_data_2018/UCR_TimeSeriesAnomalyDatasets2021.zip

[5] https://github.com/NetManAIOps/OmniAnomaly

Datasets		Properties		# Anomalies		# Points	
Source	Name	Synthetic	Signals	Point	Contextual	Anomaly	Data
NASA	MSL	✗	27	0	36	7766	132046
	SMAP	✗	53	0	67	54696	562800
Yahoo	A1	✗	67	68	110	1669	94866
	A2	✓	100	33	167	466	142100
	A3	✓	100	935	4	943	168000
	A4	✓	100	833	2	837	168000
NAB	Art	✓	6	0	6	2418	24192
	AdEx	✗	5	0	11	795	7965
	AWS	✗	17	0	30	6312	67644
	Traffic	✗	7	0	14	1560	15662
	Tweets	✗	10	0	33	15662	158511
UCR	UCR	✗	250	0	250	49113	19353766
Source	Name	Channels	Signals	Point	Contextual	Anomaly	Data
NASA	MSL	55	27	0	36	7766	132046
	SMAP	25	53	0	67	54696	562800
INET	SMD	38	28	0	14	58940	1416825

Table 2.1: Summary of the benchmark datasets.

2.1.2 Exploratory Analysis

Table 2.1 provides a high-level overview of each dataset, including the properties, total number of anomalies, and total number of data points. The total number of anomalies is sub-divided into point anomalies identified by a single value and contextual anomalies identified by an interval. The total number of data points consists of the total number of anomalous points and observations in the dataset. A synthetic indicator is provided for univariate datasets, while the number of channels is provided for multivariate datasets.

2.1.3 Preprocessing Steps

Figure 2-1 visualizes the pre-processing steps. Each dataset is divided into train and test splits according to the data source and detrended, as necessary, by fitting a linear model. Then, the values of each signal are min-max normalized to between `[-1, 1]` according to the train split. Missing values are imputed with the mean. Let T be the number of observations in the split, n be the input size to the model (defaults to

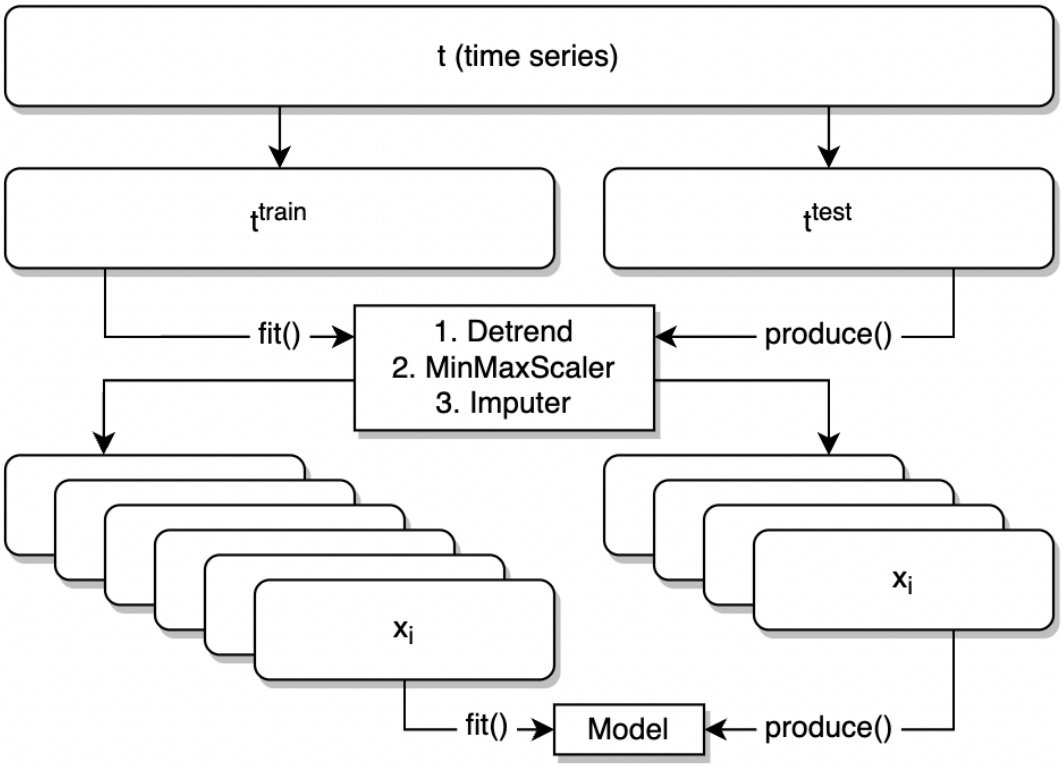

Figure 2-1: Visualization of the preprocessing steps from the original time series t to model inputs x_i. The train split uses the `fit` function while the test split uses the `produce` function based on the `MLPrimitives` library.

$n = 100$ for reconstruction-based models and $n = 250$ for prediction-based models), and d be the dimensionality. $T - n$ number of inputs $x_i \in \mathbb{R}^{n \times d}$ are then created using a stride of one. The output differs between prediction-based and reconstruction-based models. In the case of prediction-based models, the prediction for index i of the original time series is $f_i \in \mathbb{R}$. In the case of many-to-one reconstruction-based models, the reconstructed sequence starts at index i is $y_i \in \mathbb{R}^n$.

2.2 Evaluation

2.2.1 Problem Definition

Unsupervised time series anomaly detection aims to find a set of anomalous intervals from time series t with d observations per time step. Ideally, each anomalous interval

captures an unexpected behavior that deviates from the pattern in the signal.

2.2.2 Identifying Anomalous Sequences

Chapter 3 documents the method for constructing anomaly scores from prediction-based and reconstruction-based models. Given the anomaly scores, a locally adaptive thresholding function from Hundman et al. [9] is used to identify anomalous intervals. It uses a sliding window to compute local thresholds, join continuous observations to create anomalous sequences, and mitigate false positives by pruning anomalies.

Let α be the sequence of anomaly scores with a maximum size of length T (one score for each observation). The window size defaults to $\frac{T}{3}$ with a step size of $\frac{T}{3*10}$ to optimally identify both point and contextual anomalies. The adaptive threshold for each sliding window is four standard deviations from the window's mean. Observations with scores that exceed that threshold are identified as anomalous. Consecutive anomalous time steps are joined together to create K anomalous sequences. A pruning method is used to reduce the number of false positives. Let the maximum anomaly score $K_{max}^{(i)}$ represent each anomalous sequence $K^{(i)}$. The maximums are sorted in descending order, and the decrease percentage change $p^{(i)}$ is calculated between $K_{max}^{(i)}$ and $K_{max}^{(i+1)}$. At the sequence $K^{(j)}$ whose percentage change $p^{(j)}$ does not exceed an empirically defined threshold θ (defaults to 0.13), that sequence and all subsequent sequences are reclassified as normal, i.e., all sequences between `[j, K]`.

2.2.3 Evaluation Metrics

Similar to Hundman et al. [9] and Geiger et al. [7], the metric used in this study is unweighted contextual F1 scores for each dataset. The motivation is that anomalies are rare and window-based in many real-world application scenarios. The end-users' goal is to detect timely true alarms without receiving many false positives. This evaluation metric prioritizes finding any part of the anomalies.

Anomaly scoring is based on overlapping segments: a true positive (TP) if a known anomalous window overlaps any detected windows, a false negative (FN) if a known

anomalous window does not overlap any detected windows, and a false positive (FP) if a detected window does not overlap any known anomalous region. Two final metrics summarize the F1 scores across datasets into one number. The first method gives equal importance to all datasets by averaging the F1 scores (Avg. F1). The second method emphasizes datasets with more signals by tallying up all TP, FN, and FP values across all datasets to compute the F1 score (Total F1).

Chapter 3

Baseline Models

The baseline models for time series anomaly detection are divided into prediction-based and reconstruction-based approaches. All baseline models are implemented using TensorFlow 2.0 [1] under Orion [7], a machine learning library for unsupervised time series anomaly detection.

3.1 Prediction-based Methods

Prediction-based methods learn patterns from a given time series to forecast future values. An observation is classified as anomalous if the difference between the forecasted value and the original value exceeds a certain threshold. The prediction-based models used in this study are AutoRegressive Integrated Moving Average (ARIMA) [15] and LSTM with Non-parametric Dynamic Thresholding (LSTM-NDT) [9].

3.1.1 Prediction-based Anomaly Scores

Given the input $x_i \in \mathbb{R}^{n \times d}$ with length n and dimensionality d starting at index i, prediction-based models produce an one-step forecast in the forward direction f_{n+i} at index $n + i$. Let t be values of the time series and T be the total number of observations. Only forecasts f_{n+i} for $i \in [n+1, T)$ can be calculated since the models require at least n observations to forecast the first value at the index $n + 1$. The

prediction-based anomaly score α_p takes the absolute mean error between the time series t and the sequence of forward predictions f.

$$\alpha_p(t, f) = \begin{cases} 0 & i \in [1, n+1) \\ |t_i - f_i| & i \in [n+1, T) \end{cases} \tag{3.1}$$

An exponentially weighted moving average with a smoothing window of $0.1 * T$ is applied to the anomaly score to reduce noise.

3.1.2 Autoregressive Integrated Moving Average (ARIMA)

ARIMA is a popular statistical model that uses lags and lagged forecast errors to predict future values. Three terms characterize the model: the order of the autoregressive term p, differencing d, and moving average term q. The auto-regressive model $AR(p)$ uses a weighted sum of lagged observations to capture different temporal structures. The integrated differencing uses a d order differencing to ensure stationarity. Finally, the moving average model $MA(q)$ uses the weighted sum of lagged prediction errors to capture the relationship between observations and residual errors. However, ARIMA has several drawbacks. First, it is sensitive to parameter selection and requires extensive domain knowledge about the time series. Second, it cannot model seasonal or cyclic variations. Finally, it cannot model multivariate time series and learn correlations between exogenous variables.

ARIMA is implemented with the `StatsModels` library. The hyperparameters are empirically set to `p=1, d=0, q=0`. The inputs are $x_i \in \mathbb{R}^{n \times d}$ such that $n = 250$ and d depends on dimensionality of the dataset. The outputs are predictions in the forward direction $f_{n+i} \in \mathbb{R}$.

3.1.3 LSTM Non-parametric Dynamic Threshold (LSTM-NDT)

LSTM-NDT [9] uses LSTMs and a novel non-parametric dynamic thresholding technique to detect anomalies in telemetry time series data. Hundman et al. proposed maintaining a single model per channel to facilitate more granular system control and

mitigate errors from high dimensionality outputs. Their research also promoted an exponentially-weighted average function to smooth anomaly errors, dynamic thresholds to produce anomalous intervals, and a pruning methodology to reduce false positives.

Similar to ARIMA, the inputs are $x_i \in \mathbb{R}^{n \times d}$ such that $n = 250$ and d depends on the dimensionality of the dataset. The outputs are predictions in the forward direction $f_{n+i} \in \mathbb{R}$. The original LSTM-NDT uses two LSTM layers with 80 units and dropout rate of 0.3. The training hyperparameters were 35 epochs, batch size of 64, and Adam optimizer. Simplifying the architecture, a one-layer version LSTM-NDT-1 is also benchmarked in this study. The specifics of the models are located in the appendix figure A-1.

3.1.4 Results & Discussion

Appendix table B.2 documents the results for each prediction-based model. LSTM-NDT and LSTM-NDT-1 outperformed ARIMA with higher average F1 scores. They also had lower average training times and did not encounter convergence issues. RNN layers can model non-linear relationships and better forecast future values. The fact that LSTM-NDT-1 perform similarly to LSTM-NDT for both univariate and multivariate datasets suggests that a simpler model with fewer parameters might be beneficial.

Visualizations of the time series t and the corresponding prediction-based anomaly scores α_p for each model are plotted to understand the successes and limitations in figures 3-1, 3-2, and 3-3. The top graph in each figure shows the time series with the ground truth anomalous intervals highlighted in red. The bottom graph shows the prediction-based anomaly score for ARIMA (green), LSTM-NDT (blue), and LSTM-NDT-1 (purple).

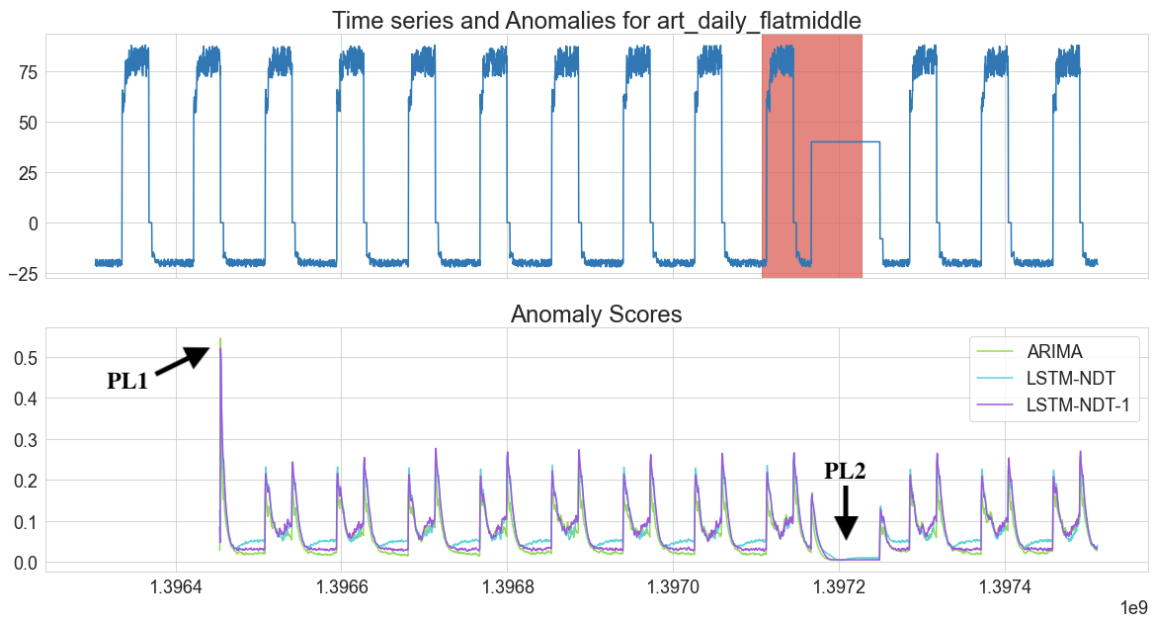

Figure 3-1: `art_daily_flatmiddle` signal from the `artificialWithAnomaly` dataset shows that prediction-based models produce high prediction-based anomaly scores near the beginning (PL1) and low anomaly scores for contextual anomalies with simple patterns (PL2).

PL1 - High anomaly scores at the start result in false positives

Figure 3-1 shows that high prediction-based anomaly scores towards the beginning resulted in false-positive predictions. This behavior occurs in many signals across multiple datasets. The error is the byproduct of the exponential weighted moving average function used to smooth the anomaly scores. The function requires at least the same number of observations as the size of the smoothing window before producing stable anomaly scores.

PL2 - Low anomaly scores for contextual anomalies with simple patterns

Figure 3-1 also shows that prediction-based models produce low anomaly scores at the contextual anomaly with a simple pattern. The cyclic pattern in prediction-based anomaly scores suggests that the models could not fully capture the structure, especially at the change point in the time series. However, in this case, the contextual anomaly is a simple pattern. Therefore, the models can easily forecast the pattern, resulting in nearly zero anomaly scores at the interval. Hence, the adaptive threshold

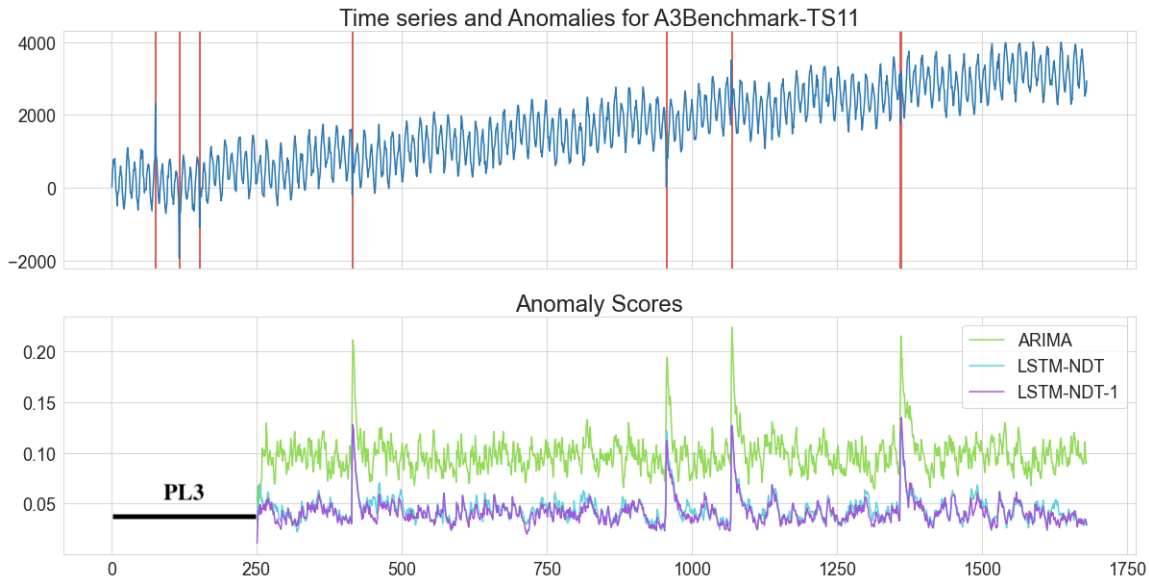

Figure 3-2: `A3Benchmark-TS11` signal from the `YAHOOA3` dataset shows that prediction-based methods fail to find anomalies at the start of the time series (PL3).

failed to find the contextual anomaly.

PL3 - Missing predictions at the start of the time series

Figure 3-2 shows false negative predictions for anomalies located at the beginning of the time series. These false negatives occur because prediction-based models require at least n observations to forecast the first value at index $n+1$. This behavior mainly impacts datasets like `YAHOOA3` and `YAHOOA4` with a decent amount of point anomalies at the start of the time series. False-negative predictions will artificially decrease the F1 scores for those datasets.

PL4 - Short-sightedness produces many false positives

Figure 3-3 shows prediction-based models producing a false positive prediction for every spike in the signal without anomalies. This behavior occurs when the model fails to capture the long-term pattern or when input context n is not sufficiently long enough. In such cases, the false positives significantly decrease the F1 scores of datasets such as `realAWSCloudwatch`.

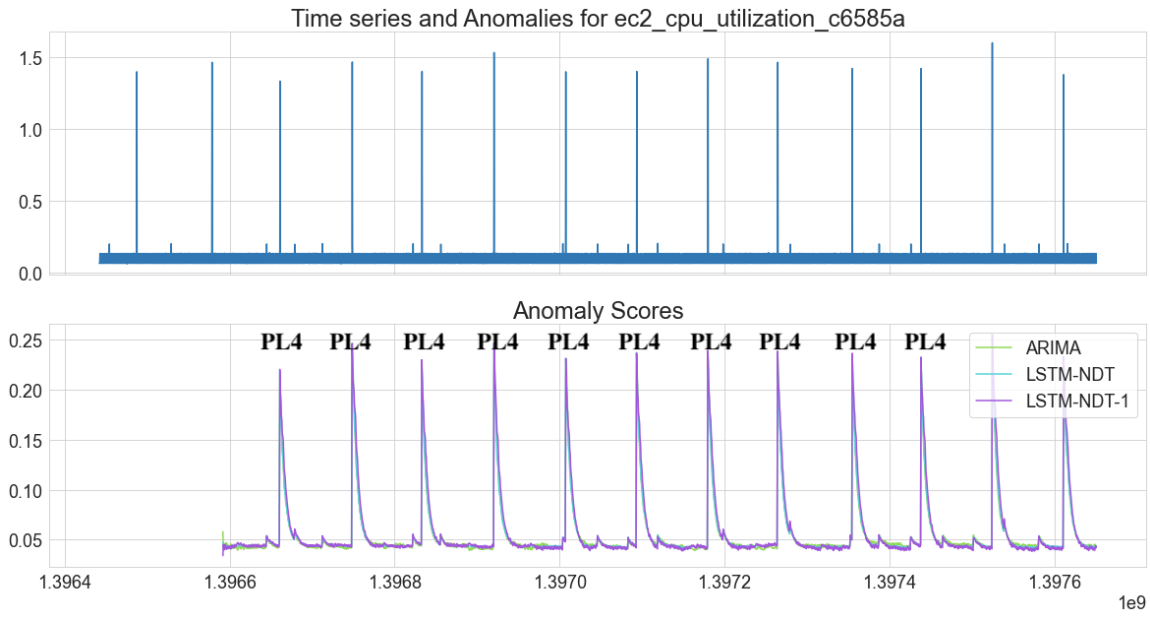

Figure 3-3: `ec2_cpu_utilization_c6585a` signal from the `realAWSCloudwatch` dataset that shows prediction-based models producing many false positives for a signal without anomalies (PL4).

3.2 Reconstruction-based Methods

Reconstruction-based methods capture a low-dimensional latent representation using an encoder function and reconstruct the input using the representation via a generator function. Rare events like anomalies are not captured in the latent representation and are less likely to be accurately reconstructed, leading to high reconstruction-based anomaly scores. Similar to Geiger et al. [7], this research explores three methods for calculating anomaly scores: point-wise differencing, area differencing, and Dynamic Time Warping (DTW). The reconstruction-based models used in this study are LSTM autoencoders (LSTM-AE) [8], LSTM variational autoencoders (LSTM-VAE) [14], and generative adversarial networks (TadGAN) [7].

3.2.1 Reconstruction-based Anomaly Scores

The input of reconstruction-based models is $x_i^{1\cdots n} \in \mathbb{R}^{n\times d}$ with length n and dimensionality d starting at index i of time series t. The output $y_i^{1\cdots n} \in \mathbb{R}^n$ is the reconstructed sequence of values of one channel in t. The models consist of an encoder

E and a generator G function such that $y_i^{1\cdots n} = G(E(x_i^{1\cdots n}))$. Each index i in the time series t_i for $i \in [1, T]$ has multiple reconstructed values since that index occurs in multiple sequences of $y_i^{1\cdots n}$. Hence, the final value for index i is the median of the collection of reconstructed values for that index. Given the full sequence of t and y, the reconstruction-based anomaly scores can be calculated in the three different ways laid out above. Similar to prediction-based anomaly scores, an exponential weighted moving average with a smoothing window of $0.1 * T$ is applied to the anomaly score to reduce noise.

Point-wise differencing (PD)

The reconstruction-based anomaly score $\alpha_{r,p}$ takes the absolute mean error between the time series t and the reconstructed value y every index i.

$$\alpha_{r,p}(t, y) = |t_i - y_i| \qquad i \in [1, T] \tag{3.2}$$

Area differencing (AD)

The reconstruction-based anomaly score $\alpha_{r,a}$ is created using a fixed length window size that measures the similarity between local regions.

$$\alpha_{r,a}(t, y) = \frac{1}{2l} \left| \int_{i-l}^{i+l} t_i - y_i \, dt \right| \qquad i \in [1, T] \tag{3.3}$$

The similarity is measured by the average difference between areas beneath two curves of length $2l$ calculated using the trapezoidal rule (defaults to $l = 10$).

Dynamic Time Warping (DTW)

The reconstruction-based anomaly score $\alpha_{r,d}$ created with dynamic time warping allows for nonlinear alignment between two sequences that are locally out of phase [4]. DTW creates a cost matrix $C \in \mathbb{R}^{2l \times 2l}$ such that each (i, j) coordinate represents the

distance c_k between t_i and y_j.

$$\alpha_{r,d}(t, y) = \min_{C} \left[\frac{1}{K} \sqrt{\sum_{k=1}^{K} c_k} \right] \tag{3.4}$$

Dynamic programming solves for the optimal wrap path C^* with the minimum warp distances between the t and y series.

3.2.2 LSTM Autoencoders (LSTM-AE)

LSTM-AE is an autoencoder with LSTM layers that reconstructs the output y using a latent representation z of the input x. The size of the latent space balances the tradeoff between information loss and the accuracy of reconstructed values.

The inputs are $x_i \in \mathbb{R}^{n \times d}$ such that $n = 100$ and d depends on the dimensionality of the dataset. The outputs are the reconstructed values of one channel $y_i \in \mathbb{R}^n$. The model uses one LSTM layer with 60 units for the encoder and generator. A time-distributed layer with a dense one-unit layer is used to create the output. The specifics of the models are located in the appendix figure A-2a.

3.2.3 LSTM Variational Autoencoders (LSTM-VAE)

LSTM-VAE introduces regularization in the latent space using a probabilistic encoder $q(z|x)$ and decoder $p(x|z)$. The encoder learns the mean and standard deviations to reparameterize the prior distribution to create the latent vector z. This method avoids overfitting and ensures that the latent space has good properties that enable a generative process. Unlike autoencoders, variational autoencoders optimize the evidence lower bound (ELBO) criterion. ELBO consists of two terms: maximization of the expected log-likelihood of the observation and minimization of the KL-divergence between the posterior distribution $q(z|x)$ and the prior distribution $p(z) \sim N(0, I)$.

The inputs are $x_i \in \mathbb{R}^{n \times d}$ such that $n = 100$ and d depends on the dimensionality of the dataset. The outputs are the reconstructed values of one channel $y_i \in \mathbb{R}^n$. The encoder uses one shared LSTM layer with 60 units and separate dense layers, each

with 60 units, to create the mean and standard deviation vector. The decoder uses a repeat vector layer, a LSTM layer with 60 units, and a time-distributed layer with a dense one-unit layer. The specifics of the models are located in the appendix figure A-2b.

3.2.4 TadGAN

TadGAN is a generative adversarial network that also uses a cycle consistency loss [20] to resolve mode collapse issues in regular GANs. The model has an encoder E and generator G, each with its critic, C_z and C_x. Let x be the input and z be the corresponding latent vector. Also, let the prior be a white noise matrix that follows a standard multivariate normal distribution $N(0, I)$. C_x tries to distinguish the real input sequences x from the reconstructed output $G(z)$. Likewise, C_z tries to distinguish the real latent noise z from the generated latent representation $E(x)$. The objective function consists of the Wasserstein loss with Wasserstein-1 distance [2] and the cycle consistency loss [20]. The Wasserstein loss trains each output distribution of E and G to match the target distribution of z and x. This loss motivates both encoder and generator to produce new samples that fool their respective critics C_z and C_x. The loss for each generator-critic pair is as follows:

$$\min_{G} \max_{C_x} V_x(C_x, G) = \mathbb{E}_x[C_x(x)] - \mathbb{E}_z[C_x(G(z))] \tag{3.5}$$

$$\min_{E} \max_{C_z} V_z(C_z, E) = \mathbb{E}_z[C_z(z)] - \mathbb{E}_x[C_z(E(x))] \tag{3.6}$$

The cycle consistency loss motivates the model to reconstruct samples $\hat{x}$ that are similar to the input x. Similar to autoencoders, this loss function minimizes the L2 norm between the input x and the reconstructed output $G(E(x))$; that is,

$$V_{L2}(E, G) = \mathbb{E}[||x - G(E(x))||_2] \tag{3.7}$$

The full objective combines equations 3.5, 3.6, and 3.7.

$$\min_{E,G} \max_{C_x,C_z} V_X(C_x, G) + V_Z(C_z, E) + V_{L2}(E, G) \tag{3.8}$$

The inputs are $x_i \in \mathbb{R}^{n \times d}$ such that $n = 100$ and d depends on the dimensionality of the dataset. The outputs are the reconstructed values of one channel $y_i \in \mathbb{R}^n$. The latent space is $z_i \in \mathbb{R}^z$ (defaults to $z = 20$). The encoder and generator each use bi-directional LSTM layers, while both critics use 1D convolution layers. The specifics of the models are located in the appendix figure A-2c.

Incorporating critics into anomaly scores

Both the reconstruction score between input x_i and reconstructed output $G(E(x_i))$ and the critic score from $C_x(x_i)$ can be used to create the final anomaly scores. Critic C_x could offer another perspective, since it is optimized to distinguish between real and fake reconstructed outputs. Geiger et al. reported an ablation study merging these scores using summation, product, critic-only, and reconstruction-only combinations. This study will report the optimal combination of critic and reconstruction scores for each dataset.

TensorFlow Implementations and GPUs

TadGAN is originally implemented in TensorFlow 1.x using Keras Layers. Since then, TensorFlow has released the 2.x version allowing for tighter integration with Keras and eager execution to enhance model development. With the intention of future-proofing, this study will re-implement TadGAN in TensorFlow 2.x. The main difference between the two environments is that eager execution evaluates operations and returns actual values of tensors as they occur in the code without building graphs. Hence, the most significant change is the model compilation and the implementation of gradient penalty loss. In TensorFlow 1.x, the gradients are obtained from the Keras backend to calculate the gradient penalty loss. However, TensorFlow 2.x handles these gradients using the `tf.GradientTape` function and requires a second-order derivative

to calculate the gradient penalty loss.

Graphics processing units (GPUs) are used to significantly reduce the training time of TadGAN and other baseline models. However, this requires some minor changes to the layers of TadGAN. TensorFlow 1.x offers a separate LSTM Keras layer backed by cuDNN in `tf.keras.layers.CuDNNLSTM`. On the other hand, TensorFlow 2.x has the cuDNN LSTM layer built directly into the standard LSTM layer in `tf.keras.layers.LSTM`. TensorFlow 2.x automatically uses the cuDNN-backed version during runtime if all requirements are met. One such requirement is that the recurrent dropout rate (fraction of the units to drop for the linear transformation of the recurrent state) must be 0.0, but that directly conflicts with the original implementation, which uses 0.2.

3.2.5 Results & Discussion

Appendix table B.3 documents the results for LSTM-AE and LSTM-VAE models using various reconstruction-based anomaly scoring methods. Appendix table B.4 documents the results for all reconstruction-based baseline models. The results of LSTM-AE and LSTM-VAE using DTW to calculate the anomaly scores are similar to those of TadGAN. These results suggest that the LSTM-AE architecture alone is sufficient to capture the same information as TadGAN with less parameters. The results of TadGAN implemented in TensorFlow 2.x are similar to those in TensorFlow 1.x. However, TadGAN implemented in TensorFlow 2.x took significantly longer to train. This could be because second-order derivatives might be more computationally intensive in eager execution or because the model was not compiled optimally.

As with the prediction-based models, visualization of the time series t and the corresponding reconstruction-based anomaly scores for each model are plotted. The top graph in figures 3-4, 3-5, and 3-7 shows the time series with the ground truth anomalous intervals highlighted in red. The middle three graphs correspond to point-wise differencing $\alpha_{r,p}$, area differencing $\alpha_{r,a}$, and DTW $\alpha_{r,d}$ anomaly scores for LSTM-AE (dark green) and LSTM-VAE (purple) models. The anomaly scores for TadGAN (light green) are plotted on a separate graph since they combine recombination and

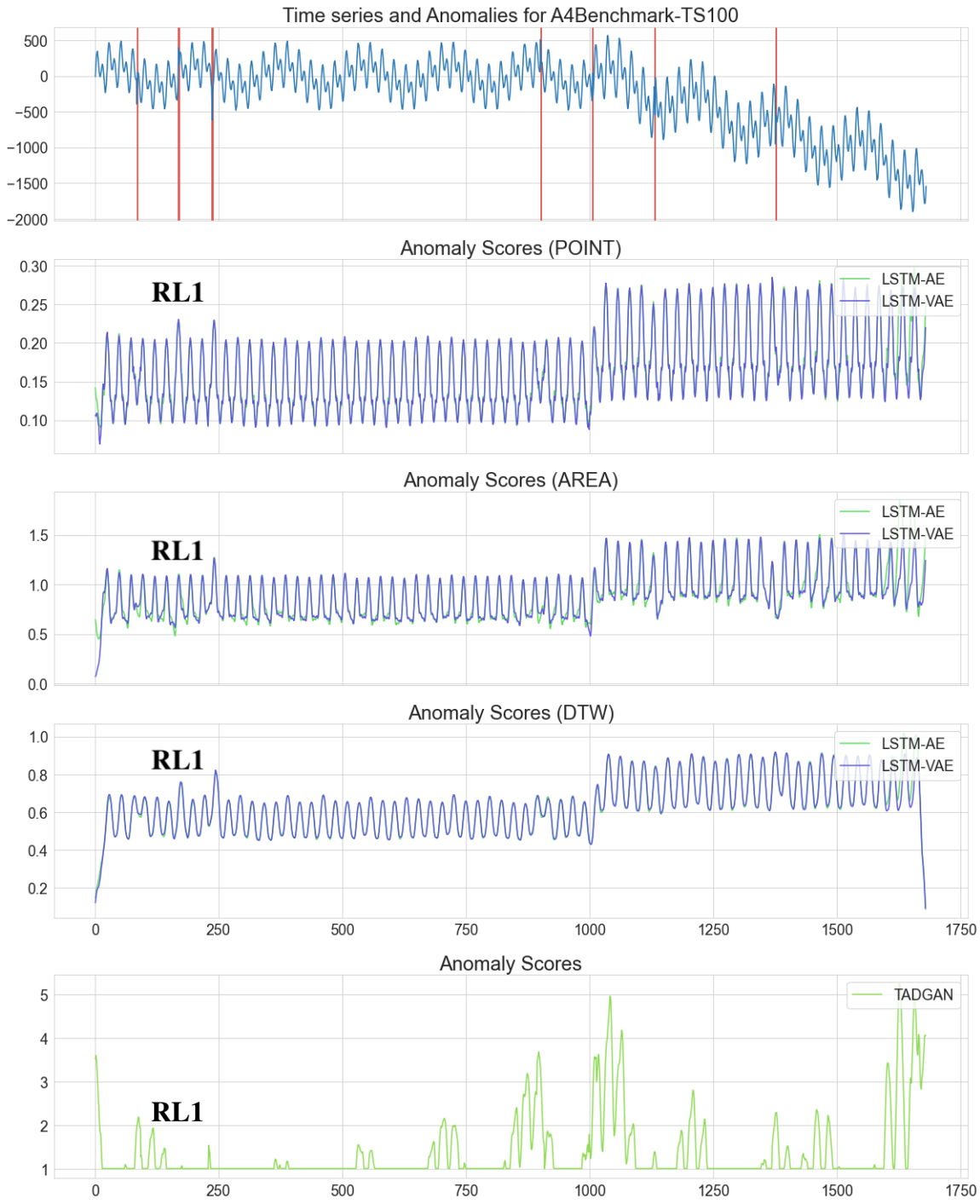

Figure 3-4: `A4Benchmark-TS100` signal from `YAHOOA4` dataset shows that reconstruction-based anomaly scores fail to capture point anomalies (RL1).

critic scores. The principal component analysis (PCA) of the reconstructed output y_i is also plotted in figures 3-6 and 3-8 to visualize the output in 2D space for each model.

RL1 - Reconstruction-based scores reduce peaks for point anomalies

Figure 3-4 shows how reconstruction-based methods like LSTM-AE and LSTM-VAE fail to capture point anomalies. The prediction-based anomaly scores are calculated from the median of all predicted values for index i. Since some reconstructed outputs y_i are better at capturing the point anomalies than others, the median value is closer to the true value at index i. This lowers the anomaly scores such that the window-based threshold no longer captures those point anomalies since the scores are now closer to the window's mean. The anomaly score created from combining critics and reconstruction errors for TadGAN was inaccurate in this signal and produced many false positives.

RL2 - Bad train-test splits result in mode collapse

Figure 3-5 shows the train-test split of a signal in the univariate benchmark dataset leading to mode collapse of outputs from reconstruction-based methods. The train split consists of only -1 values, while the test split consists of non-trivial values. Therefore, any reconstruction-based methods fitted on the train split will likely reconstruct the sequences of values close to -1. Figure 3-6 confirms this idea since the same color dots for each reconstruction-based method are close to one another rather than being close to the true output (red). The pattern of the anomaly scores also closely resembles the test split of the time series. This behavior is the byproduct of information loss since the signal originates from a multivariate dataset and calls for the re-evaluation of univariate benchmark datasets in the `Orion` library.

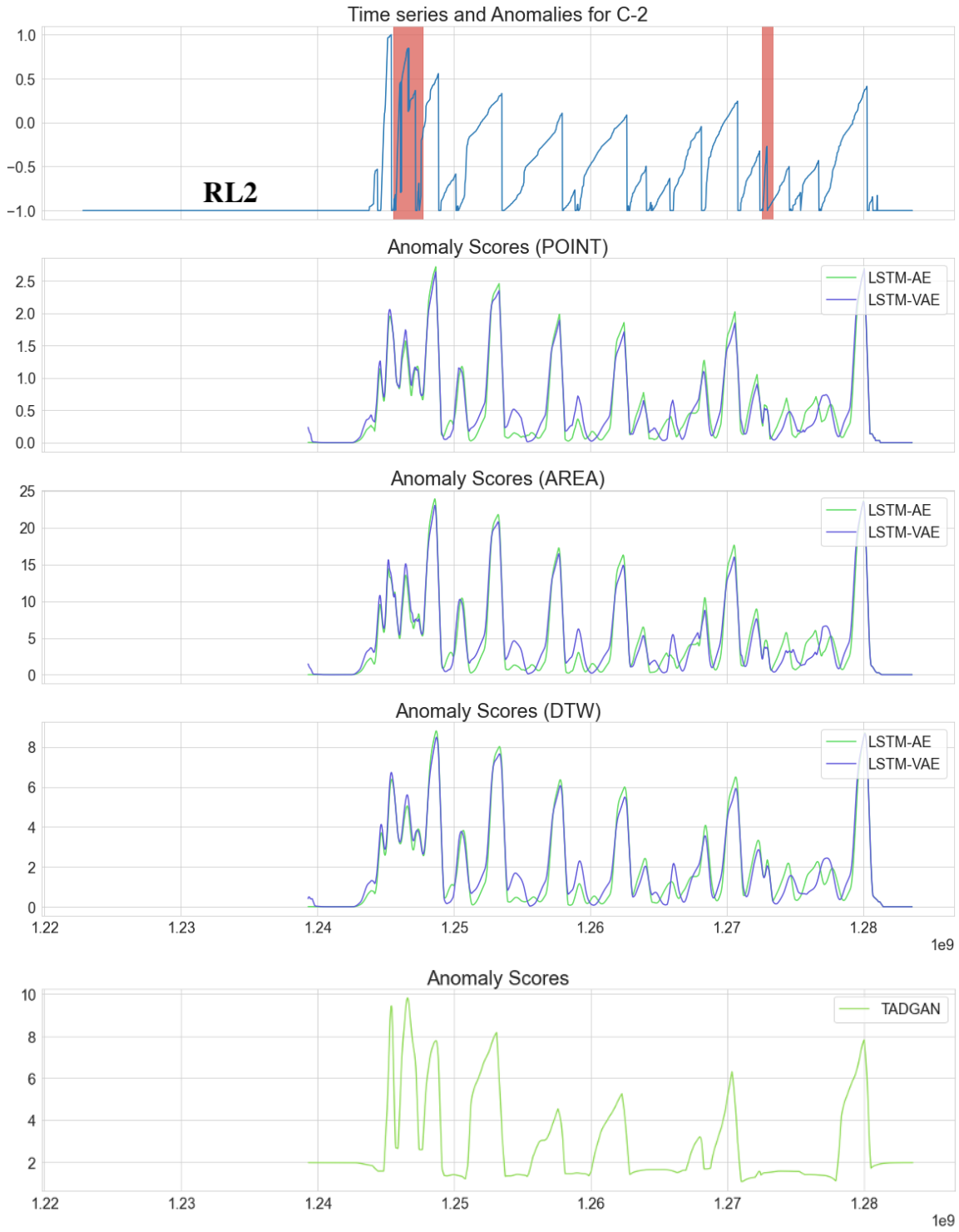

Figure 3-5: `C-2` signal from `MSL` dataset demonstrates a bad train-test split with the train split consisting of only one constant number.

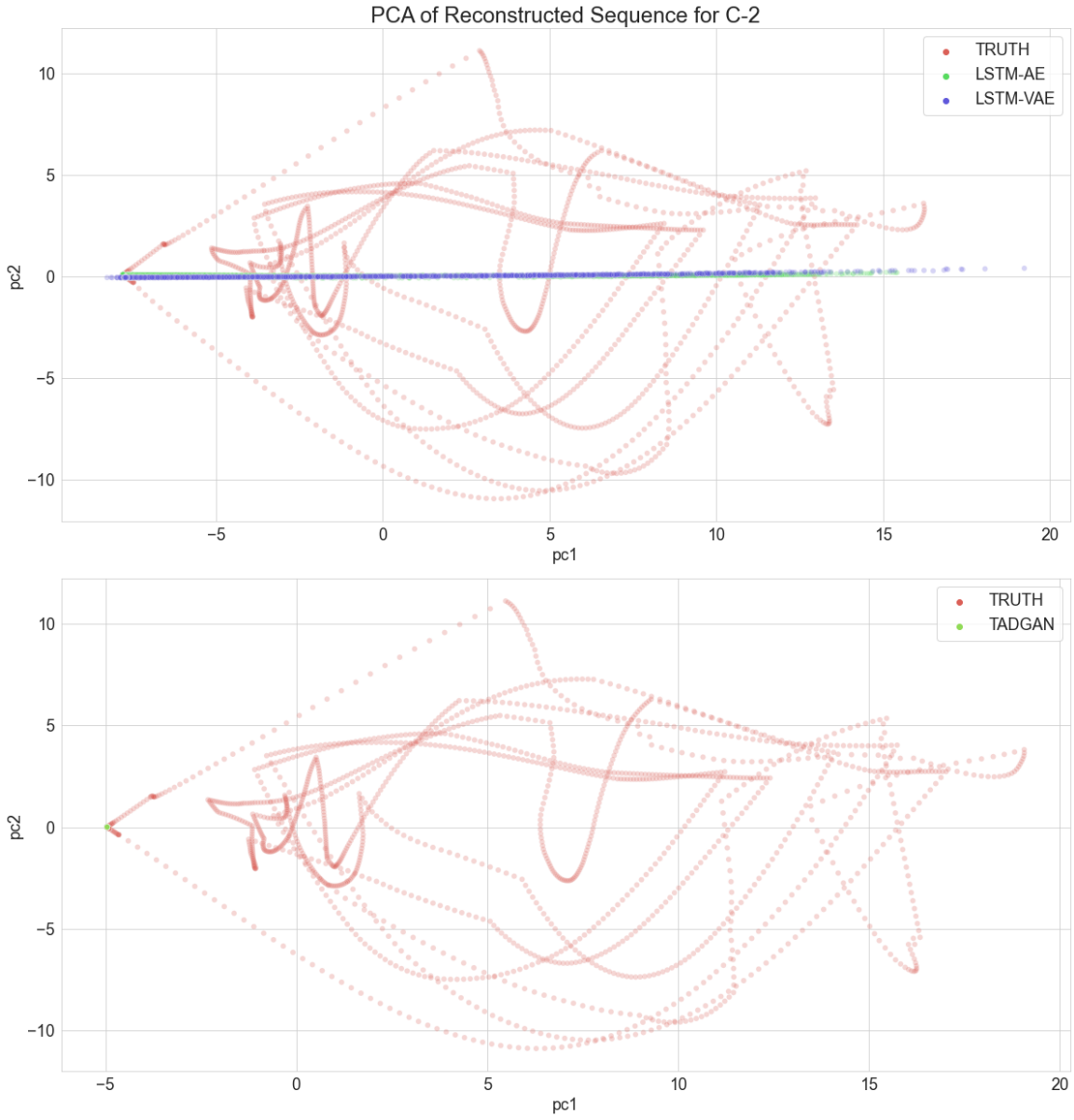

Figure 3-6: PCA of outputs from the `C-2` signal from `MSL` dataset confirms mode collapse of outputs from reconstruction-based methods.

RS1 - DTW scores are better at capturing anomalies

Figure 3-7 shows that anomaly scores created from point-wise and area differencing are noisy compared to those created from DTW at the point anomaly. The success of DTW scores is attributed to the method's ability to handle shifts in the alignment of two series. The anomaly scores created from the reconstruction and critic scores in TadGAN also seem to capture similar information. However, those scores are still relevantly noisy and produce many false positives, leading to lower F1 scores for the `YAHOOA3` dataset. Moreover, in figure 3-8, the PCA of the outputs of TadGAN for this signal suggests that the model seems to reconstruct a fuzzier version of the input compared to the autoencoder architectures.

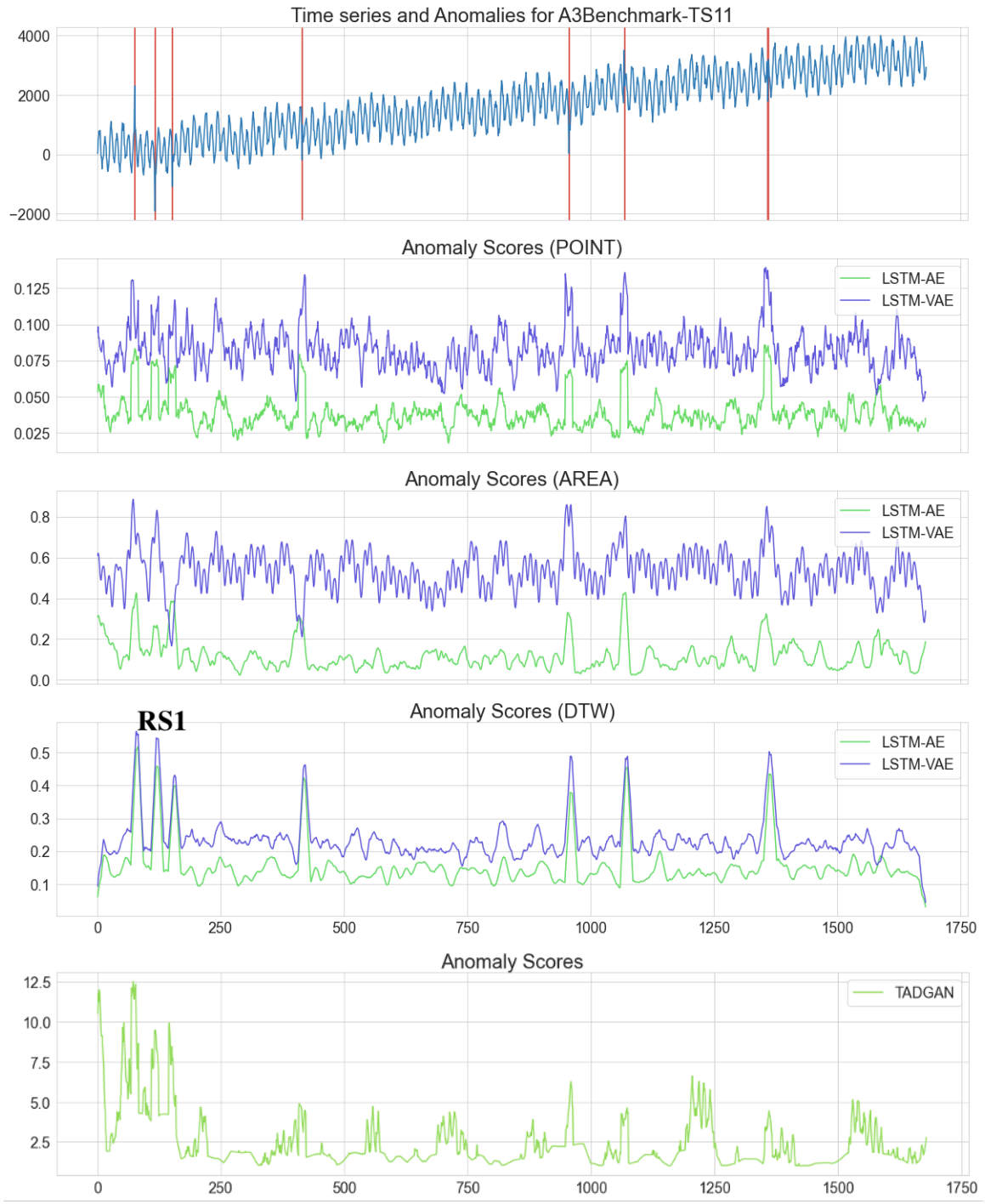

Figure 3-7: `A3Benchmark-TS11` signal from `YAHOOA3` dataset demonstrates the effectiveness of each anomaly scoring method.

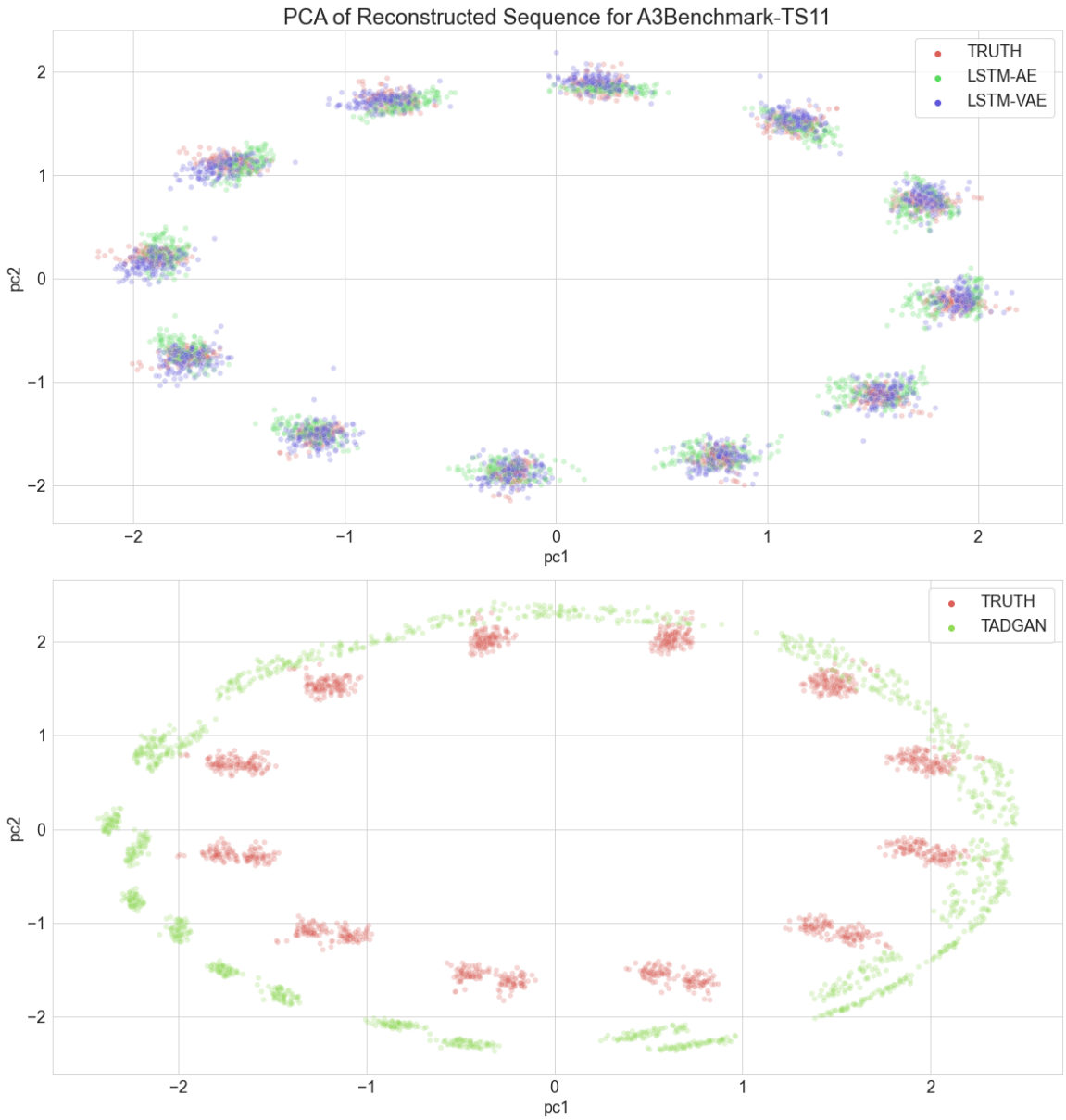

Figure 3-8: PCA of outputs from the `A3Benchmark-TS11` signal from `YAHOOA3` dataset demonstrates that LSTM-AE and LSTM-VAE models are better than TadGAN at reconstructing the input.

Chapter 4

Prediction-Reconstruction Models

4.1 Optimizing Existing Models

4.1.1 Smoothing Function Masking

The first optimization targets the start of sequence false-positive predictions created from the exponential weighted moving average smoothing function (PL1). A solution is to mask m indices from the start of the sequence with some values. Using the minimum of anomaly scores as the masking value provided the best results. By default, m is equal to $0.01 * T$ (size of smoothing window) such that T is the length of the time series.

Masking improves average F1 scores for all models

Appendix tables B.5 and B.6 show the results of masking prediction-based and reconstruction-based anomaly scores. The average F1 score increased by ~ 0.02 for prediction-based methods and ~ 0.005 for reconstruction-based methods. Prediction-based methods saw a greater increase since those methods tend to make more false-positive predictions. In general, masking had more of a negative effect on prediction-based methods than reconstruction-based methods for the `YAHOOA3` dataset. The `YAHOOA3` dataset has many point anomalies at the start of the signal. Hence, removing extra anomaly scores on top of the anomaly scores that prediction-based models

cannot produce further exaggerates the false-negative issues in the dataset. Furthermore, masking does not seem to affect the combined anomaly scores from TadGAN.

4.1.2 Bi-directional Regression

The second optimization targets the missing start of sequence anomaly scores from prediction-based methods (PL3). This issue occurs because prediction-based methods require at least n observations to make the first prediction at the $n + 1$ index. A solution is to perform regression in both directions to produce anomaly scores in the forward f and reverse r directions. The reverse r direction scores fill in the missing anomaly scores at the beginning. Bi-directional regression can be implemented by either training two separate models using forward and reverse sequences or training one joint model that can forecast in both directions.

Two Separate Models (biLSTM-S)

Appendix figure A-3a implements bi-directional regression by training two prediction-based models separately. One model trains on the sequences in the forward direction $x_{i,f}$, and the other trains on the sequences in the reverse direction $x_{i,r}$. For this study, the prediction-based models are LSTM-NDT with the same hyperparameters. While this implementation works with any prediction-based models, training time will be twice the time it takes to train one model.

One Joint Model (biLSTM-J)

Appendix figure A-3b implements bi-directional regression using one joint model to reduce training time. Instead, the implementation uses a biLSTM layer to eliminate the need to reverse the inputs. The biLSTM layer consists of two LSTMs trained on inputs in the forward and reverse directions. The last hidden state from the biLSTM layer is used to forecast in the forward direction f, while the first hidden state is used to forecast in the reverse direction r. Both hidden states are connected to their respective dropout and dense layer, similar to LSTM-NDT. The biLSTM layer uses

40 units, for a total of 80 units across the two LSTM sub-components, to keep the dimensionality of the hidden states consistent with that of LSTM-NDT.

Calculating bi-directional anomaly scores

Bi-directional prediction-based anomaly scores are similar to prediction-based anomaly scores, with a few minor modifications. Let T denote the number of observations in the time series, t denote the time series, and n denote the input length. Also, let f denote the sequence of predictions in the forward direction, r denote the sequence of predictions in the reverse direction, and α_p denote the function to calculate prediction-based anomaly scores.

First, calculate the anomaly scores in the forward direction $\alpha_p(t, f)$ for indices $i \in [n+1, T]$ and the anomaly scores in the reverse direction $\alpha_p(t, r)$ for indices $i \in [0, T-n]$. If masking is used, then the first m values of $\alpha_p(t, f)$ are replaced with zeros, and the first m values of $\alpha_p(t, r)$ are replaced with $min(\alpha_p(t, r))$. Then, the $\alpha_p(t, f)$ are padded with n zeros in the beginning while $\alpha_p(t, r)$ are padded with n at the end to align the anomaly scores. The bi-directional anomaly scores α_b consist of averages of both scores in overlapping intervals and the max between both scores in non-overlapping intervals.

$$\alpha_b(t, f, r) = \begin{cases} \alpha_p(t, f) & [0, n+m] \\ \frac{1}{2}\alpha_p(t, f) + \frac{1}{2}\alpha_p(t, r) & [n+m, T-(n+m)] \\ \alpha_p(t, r) & [T-(n+m), T] \end{cases} \tag{4.1}$$

Appendix figure A-3c shows a visualization of combining the forward and reverse anomaly scores.

Results & Discussion

Appendix table B.7 compares the results of bi-directional prediction-based masked models with those of baseline prediction-based masked models. The biLSTM-S-M and biLSTM-J-M models produced average F1 scores that are slightly higher than

those produced by LSTM-NDT-1-M. Both models saw a significant increase in F1 scores for `YAHOOA3` and `YAHOOA4`, as those datasets have point anomalies at the beginning that baseline prediction-based models missed. Moreover, the drawbacks of the masking function no longer apply to the `YAHOOA3` dataset since there are anomaly scores at the beginning. However, bi-directional models generally produce more false positive predictions, which lowers the F1 scores for some datasets. For example, LSTM-NDT already produces false positive predictions for every peak in the `ec2_cpu_utilization_c6585a` signal that has no anomalies (PL4). Filling in the anomaly scores at the beginning will only add to the list of false positive predictions.

4.2 Autoencoders with Regression (AER)

4.2.1 Motivation for Prediction and Reconstruction Scores

Prediction-based and reconstruction-based anomaly scores each have their successes and limitations. For example, prediction-based anomaly scores easily identify point anomalies but produce relatively more false positives. On the other hand, reconstruction-based anomaly scores easily identify contextual anomalies and produce relatively more false negatives.

RS2 - Reconstruction anomaly scores better capture contextual anomalies

Figure 4-1 revisits the issues prediction-based models have in detecting contextual anomalies (PL2). The reconstruction-based anomaly scores spiked while prediction-based anomaly scores remained close to zero at the contextual anomaly. This behavior occurs for prediction-based anomaly scores since the contextual anomaly pattern was easy to model. Reconstruction-based anomalies struggled to recreate the entire interval, since the model likely tries to reconstruct values from simple anomalous intervals and complex non-anomalous intervals. The sudden shift from an intricate cyclic pattern to a simple pattern results in high reconstruction-based anomaly scores.

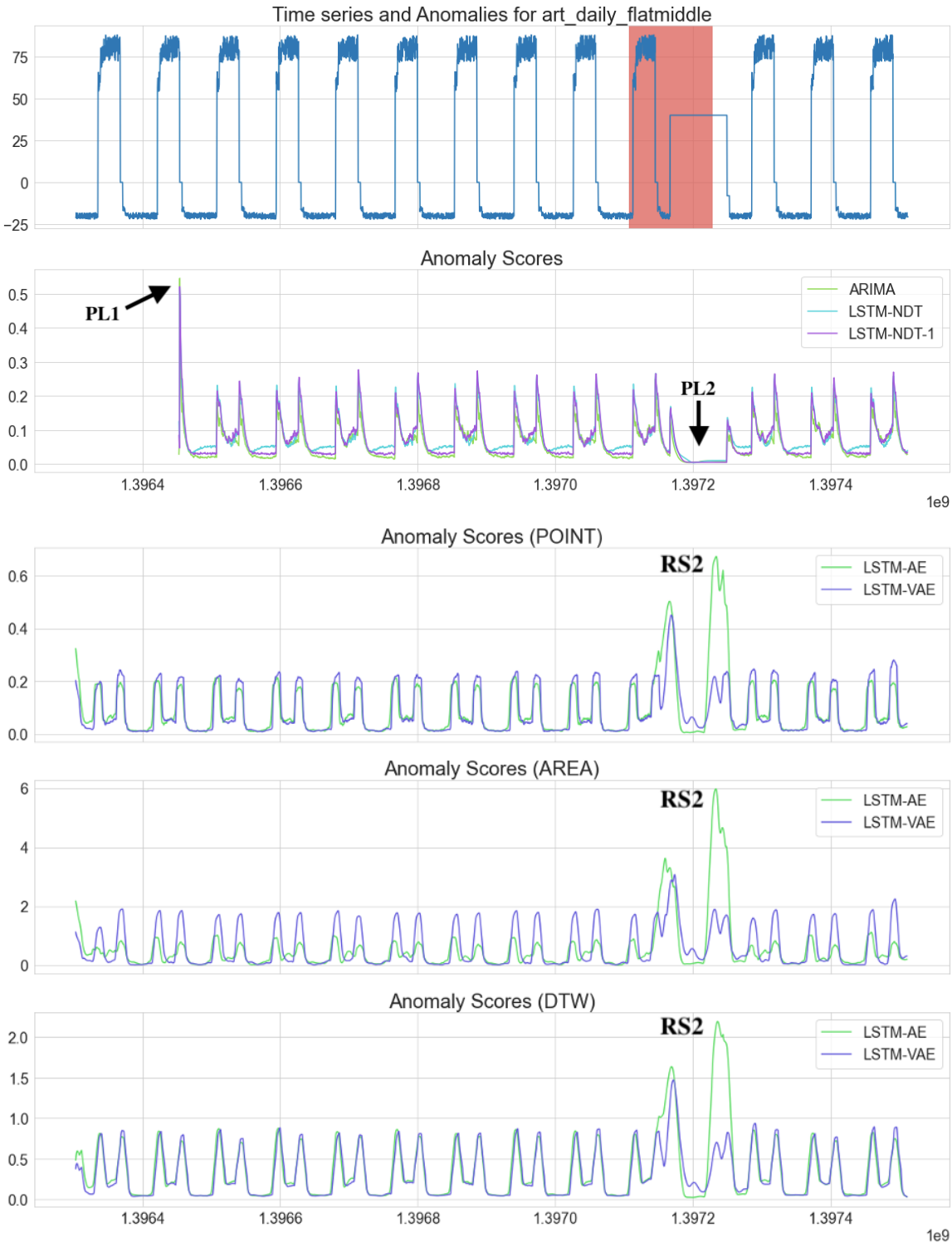

Figure 4-1: The `art_daily_flatmiddle` signal from `artificialWithAnomaly` dataset shows how prediction-based models fail to capture contextual anomalies (PL2) while reconstruction-based methods can do so (RS2).

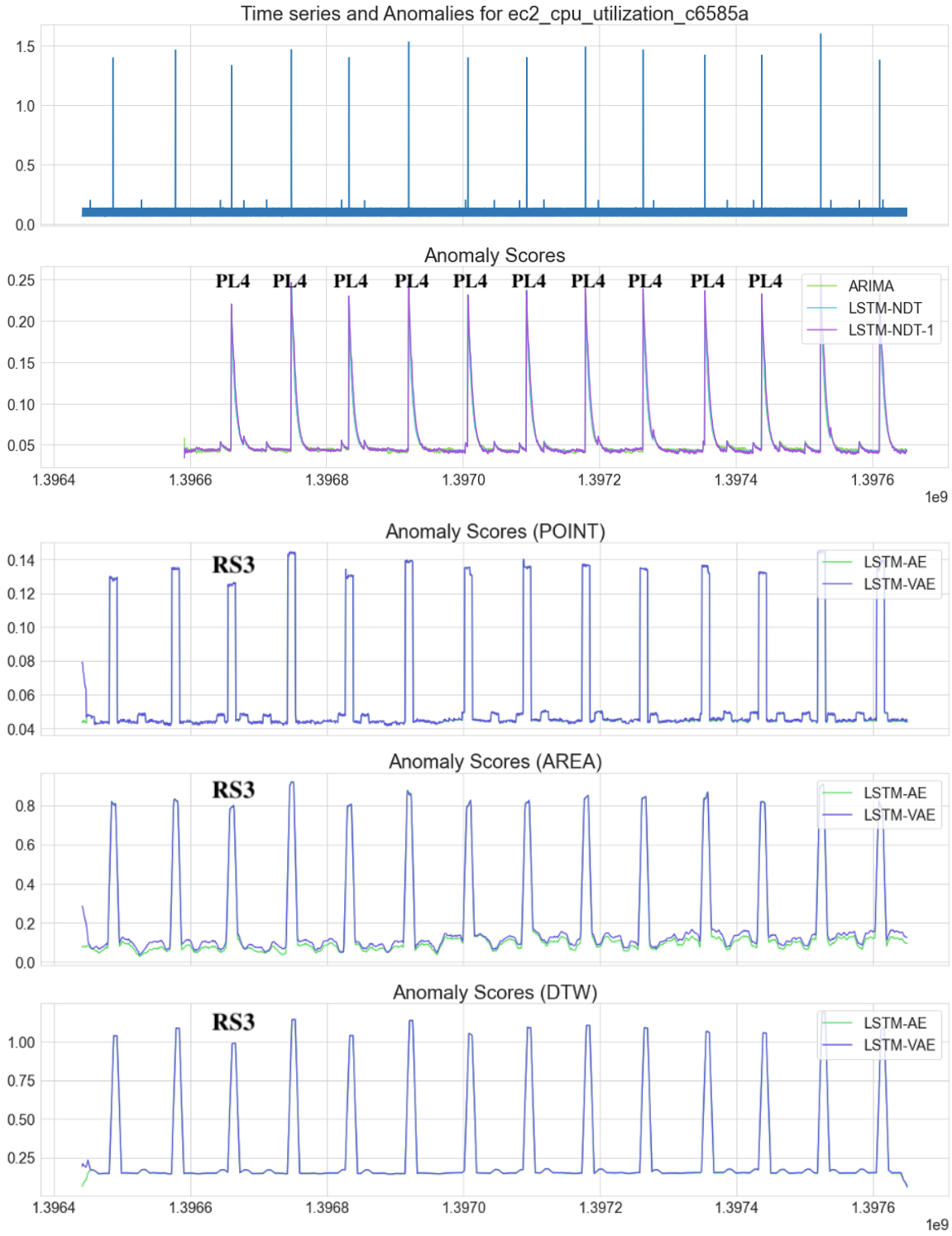

Figure 4-2: `ec2_cpu_utilization_c6585a` signal from `realAWSCloudwatch` dataset shows prediction-based models making many false positive predictions (PL4) while reconstruction-based models do not (RS3).

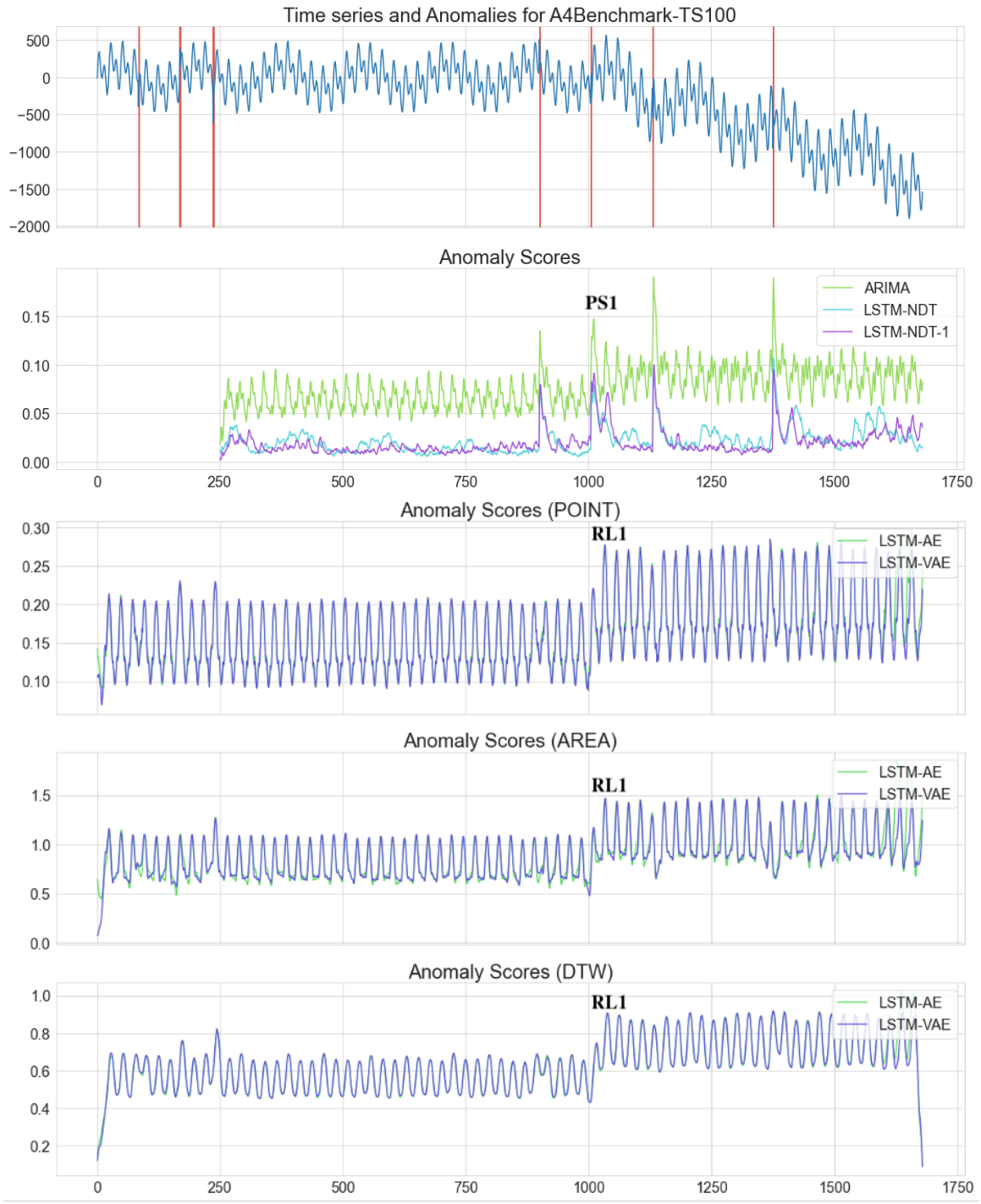

Figure 4-3: `A4Benchmark-TS100` signal from `YAHOOA4` dataset shows reconstruction-based models failing to capture point anomalies (RL1) while prediction-based models can easily capture them (PS1).

RS3 - Reconstruction anomaly scores reduce false positives

Figure 4-2 revisits the issue of prediction-based models producing many false positives caused by sudden spikes in prediction-based anomaly scores (PL4). While reconstruction anomaly scores showed similar patterns, the peaks were flatter. These flat tops help avoid false-positive predictions since the peak scores no longer exceeded four standard deviations from the mean of the locally adaptive threshold.

PS1 - Prediction anomaly scores better capture point anomalies

Figure 4-3 revisits the issues with reconstruction-based models producing many false negatives (RL1). While the reconstruction anomaly scores create some noticeable peaks, the peaks are not as apparent as those in prediction-based anomaly scores. This difference causes reconstruction-based models to miss point anomalies, resulting in lower average F1 scores for datasets like `YAHOOA3` and `YAHOOA4`. Since there is still a minor peak, multiplying both anomalies scores might amplify the scores at point anomalies.

4.2.2 Two Separate Models

The most straightforward implementation of a prediction-reconstruction model is to train one prediction-based model and one reconstruction-based model separately (AER-S). For simplicity and unbiased comparison, the bi-directional prediction-based masked model is biLSTM-NDT-1-M, and the reconstruction-based masked model is LSTM autoencoder with dynamic time warping (LSTM-AE-DTW-M).

4.2.3 One Joint Model

The joint model borrows ideas from the LSTM autoencoder and the bi-directional LSTM-NDT to produce prediction and reconstruction scores. Let the input be $x_i \in \mathbb{R}^{(n-2)\times d}$ such that $n-2$ is the number of observations and d is the dimensionality. The output is the one-step reverse prediction $r_i \in \mathbb{R}$, reconstructed sequence $y_{i+1} \in \mathbb{R}^{n-2}$, and the one-step ahead prediction $f_n \in \mathbb{R}$. Let t be the time series. The

reconstruction loss V_{rec} is the mean squared error between the time series $t_{[i+1,n-1]}$ and the reconstructed sequence y_{i+1}. The prediction loss V_{pred} is the average of the mean squared error between the time series t_i, t_n and the predicted r_i, f_n. The contribution of the prediction and reconstruction loss is determined by γ. The full objective function is:

$$\frac{\gamma}{2}V_{pred}(t_i, r_i) + (1-\gamma)V_{rec}(t_{[i+1,n-1]}, y_{i+1}) + \frac{\gamma}{2}V_{pred}(t_n, f_n) \tag{4.2}$$

By default, the hyperparameters are $n = 100$ observations per input and $\gamma = 0.5$ to give equal importance to the prediction and reconstruction losses. One biLSTM layer with $b = 30$ units is used for both the encoder and decoder. The latent space is the same dimension as the last hidden state of the bidirectional LSTM layer, which is $2b$.

Latent Space Regression (AER-L)

Figure A-4a shows an implementation that uses the features learned from the latent space to forecast one step ahead and reverse observations. The architecture is the same as in the biLSTM-J model, but the last hidden state is also used as input into the decoder of an LSTM-AE model. This design mimics that of the regression architectures.

Reconstructed Space Regression (AER-R)

Figure A-4b shows an implementation that directly reconstructs and forecasts in both directions without making changes to the latent space. The repeated vector layer in the decoder increases by two units compared with the AER-L model. This increase allows the model with input of length $n-2$ to produce outputs of length n. This design mimics that of the autoencoder architectures.

4.2.4 Prediction-Reconstruction Mixture Scores

The bi-directional prediction-based anomaly scores α_b and reconstruction-based anomaly errors α_r can both be used to create the combined anomaly scores α_c. This section

explores several ways of combining the two scores.

Prediction-based Only (PRED)

The combined anomaly scores α_c are calculated using only the bi-directional prediction-based anomaly scores.

$$\alpha_c(t, r, y, f) = \alpha_b(t, f, r) \tag{4.3}$$

Reconstruction-based Only (REC)

The combined anomaly scores α_c are calculated using only the reconstruction-based anomaly scores. The calculation of reconstruction-based anomaly scores defaults to using DTW since it outperforms point-wise differencing and area differencing.

$$\alpha_c(t, r, y, f) = \alpha_{r,d}(t, y) \tag{4.4}$$

Convex Combination (SUM)

The combined anomaly scores α_c are calculated using a convex combination with parameter weight β that controls the two errors' relative importance (by default $\beta = 0.5$). Both prediction-based and reconstruction-based anomaly scores are min-max scaled to between `[0, 1]` before the combination.

$$\alpha_c(t, r, y, f) = (1 - \beta)\alpha_{r,d}(t, y) + \beta\alpha_b(t, f, r) \tag{4.5}$$

Product (MULT)

The combined anomaly scores α_c are calculated using a point-wise product between the two scores to emphasize both scores' high values. β controls the relative importance of the two errors (by default $\beta = 1$). Both prediction-based and reconstruction-based anomaly scores are min-max scaled to between `[1, 2]` before the combination.

$$\alpha_c(t, r, y, f) = \beta\alpha_{r,d}(t, y) \odot \alpha_b(t, f, r) \tag{4.6}$$

Univariate Datasets		**ARIMA -M**	**LSTM -NDT-1 -M**	**LSTM -AE -DTW-M**	**TadGAN -TF2 -M**	**biLSTM -J -M***	**AER -R -M***
NASA	MSL	0.4565	0.4854	0.4571	0.5843	0.4898	0.6111
	SMAP	0.359	0.7391	0.7717	0.6301	0.6853	0.7846
YAHOO	A1	0.7519	0.7407	0.6448	0.5342	0.7581	0.7405
	A2	0.8093	0.9875	0.9633	0.8462	0.9448	0.9728
	A3	0.8069	0.7762	0.5938	0.395	0.9123	0.8821
	A4	0.6899	0.694	0.367	0.2913	0.8061	0.7329
NAB	Art	0.5	0.6	0.4444	0.6154	0.5455	0.6667
	AWS	0.5176	0.5312	0.6786	0.6774	0.5116	0.7119
	AdEx	0.7692	0.8148	0.6429	0.72	0.7407	0.7333
	Traffic	0.5	0.6111	0.6154	0.5806	0.6667	0.7179
	Tweets	0.5758	0.6087	0.5357	0.5882	0.5753	0.5676
Avg. F1 ($\mu \pm \sigma$)		0.6124 ± 0.1584	0.6899 ± 0.143	0.6104 ± 0.1654	0.5875 ± 0.1493	0.6942 ± 0.1554	**0.7383 ± 0.1138**
Avg. Sec ($\mu \pm \sigma$)		44 ± 110	10 ± 4	13 ± 5	145 ± 55	18 ± 9	22 ± 8
Total Precision		0.8453	0.8741	0.8571	0.6602	0.8739	0.9039
Total Recall		0.6445	0.6564	0.4342	0.3467	0.7969	0.7369
Total F1		0.7314	0.7498	0.5764	0.4546	0.8337	0.8119
Multivariate Datasets		**ARIMA -M**	**LSTM -NDT-1 -M**	**LSTM -AE -DTW-M**	**TadGAN -TF2 -M**	**biLSTM -J -M***	**AER -R -M***
NASA	MSL	-	0.5432	0.4857	0.5238	0.4731	0.6111
	SMAP	-	0.6235	0.6418	0.5882	0.6329	0.7833
INET	SMD	-	0.1551	0.144	0.131	0.1444	0.1489
Avg. F1 ($\mu \pm \sigma$)		-	0.4406 ± 0.2505	0.4238 ± 0.2546	0.4143 ± 0.2475	0.4168 ± 0.2491	**0.5144 ± 0.3281**
Avg. Sec ($\mu \pm \sigma$)		-	39 ± 42	48 ± 51	829 ± 1032	71 ± 77	76 ± 78
Total Precision		-	0.5496	0.5787	0.5066	0.5207	0.6684
Total Recall		-	0.1757	0.1506	0.1534	0.1664	0.1651
Total F1		-	0.2663	0.239	0.2355	0.2523	0.2648
UCR Dataset		**ARIMA -M**	**LSTM -NDT-1 -M**	**LSTM -AE -DTW-M**	**TadGAN -TF2 -M**	**biLSTM -J -M***	**AER -R -M***
UCR	UCR	0.1481	0.472	0.3333	0.1642	0.3378	0.4868
Total Precision		0.4681	0.4569	0.2427	0.1087	0.2697	0.4134
Total Recall		0.088	0.488	0.532	0.336	0.452	0.592
Avg. Sec ($\mu \pm \sigma$)		335 ± 539	86 ± 106	135 ±193	1898 ± 2876	387 ± 564	160 ± 214

Table 4.1: Results of new models (highlighted with *) compared to prediction-based and reconstruction-based baseline models. All anomaly scores are masked for a fair comparison. The lowest scores are highlighted in dark red, while the highest scores are highlighted in dark green.

4.2.5 Results & Discussion

Multiplication & prediction-based only combined scores are informative

Appendix B.8, B.9, B.10 documents the results for the AER-S, AER-L, and AER-R models. The results include F1 scores for each dataset using various methods to combine bi-directional prediction-based and reconstruction-based anomaly scores. MULT and PRED combination methods produce the highest average F1 scores. All models prefer the PRED combination method for the `YAHOOA3` and `YAHOOA4` datasets since they consist of only point anomalies. Likewise, all methods prefer the MULT combination method for the `MSL` and `SMAP` datasets. REC combination method consistently produced the lowest average F1 score across all models, which is consistent with other reconstruction-based baseline models. Another note is the trade-off between the precision and the recall for the MULT and and PRED combination methods: The MULT combination method has higher precision while the PRED combination method has higher recall. Finally, the average F1 scores for regression in the reconstruction space (AER-R) are higher than in the latent space (AER-L). This disparity suggests that the hidden state used to reconstruct the input conflicts with the hidden state used to forecast one step ahead.

AER-R outperforms baseline models

Table 4.1 shows the results for all baseline masked models and two proposed masked models (biLSTM-J-M, AER-R-M). AER-R-M uses a similar optimization method as TadGAN to select the best F1 scores across different anomaly combination methods. The results suggest that combining prediction-based with reconstruction-based methods often leads to better F1 scores than combining critic-based with reconstruction-based anomaly scores. This occurs because the combination of critic-based and reconstruction-based scores produced by TadGAN offers the same information as other reconstruction-based scores produced by methods like LSTM-AE-DTW-M. The optimized average F1 score of the AER-R-M model is 0.7383 for the univariate dataset, which is a significant increase over the average F1 scores of 0.5875 from TadGAN-

M and 0.6124 from ARIMA. In addition, AER-R has a lower training time than TadGAN-M without convergence issues as is the case with ARIMA. Finally, unlike other prediction-based models, AER-R also has the flexibility to combine prediction-based and reconstruction-based scores to improve F1 scores based on the dataset.

Chapter 5

Attention Mechanisms

The attention mechanism was originally developed for machine translation tasks in encoder-decoder models. For example, Bahdanau attention allowed the decoder to learn the most relevant parts of the input sequence using a weighted combination of the encoded input vectors. Later, multi-head attention was introduced to parallelize the calculations and is commonly used in transformer encoder layers. This chapter explores the effect of replacing the LSTM layers with various attention mechanisms in prediction-based models like LSTM-NDT and reconstruction-based methods like TadGAN on anomaly scores.

5.1 Bahdanau Attention (B)

Bahdanau attention [3] was introduced to address issues with decoders having limited access to information when long sequences are encoded into a fixed-length vector. The solution was to create a weighted sum of the input encoded states to allow the decoder to directly access relevant information instead of relying on the last hidden state of RNN-based architectures. This implementation of the attention mechanism consists of three main components: alignment scores, weights, and context vector. First, the network learns some weights and biases to convert the input hidden states $h \in \mathbb{R}^{n \times d}$ into alignment scores of size $\mathbb{R}^n$. Then, the softmax function is applied to these alignment scores to create the α weights. Finally, the weights are used to create the

context vector c, the weighted sum of hidden states.

$$c = \sum_{i=1}^{n} \alpha_i * h_i \tag{5.1}$$

TensorFlow's `tf.keras.layers.AdditiveAttention` is an implementation of Bahdanau attention, but it is only suitable for linear-based and CNN-based networks. A custom implementation is needed for RNN-based networks like LSTM-NDT and TadGAN.

5.2 Transformer Encoders (E)

Transformer encoders [18] represent the input sequences using different sequential positions in place of RNNs and CNNs. Advantages to this approach are that it can learn dependencies between distant observations and calculate the output in parallel instead of sequential.

First, the input passes through an embedding layer. Then, positional encodings are added to allow the layer to learn the relative positions within the input. The original implementation of positional encoding uses a sine function for even indices and a cosine function for odd indices.

$$PE_{2i} = sin(pos/10000^{2i/d}) \tag{5.2}$$

$$PE_{2i+1} = cos(pos/10000^{2i/d}) \tag{5.3}$$

The closeness between two hidden states is determined by their similarity in the context and position in the input.

The position-aware embeddings are passed into a multi-head attention block as query Q, key K, and value V to learn a representation using self-attention. The multi-head comes from splitting the dimensions of the embeddings into multiple segments. This splitting allows the model to attend to information from different representation subspaces and positions. Moreover, the total computational cost remains the same

as using single-headed attention. Each Q, K, V has its linear layer, and the output is passed into scaled dot-product attention.

$$Attention(Q, K, V) = softmax_k\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5.4}$$

The dot-product attention is scaled by a factor of the square root of the depth to prevent large magnitudes that result in small gradients. Finally, the attention output from each head is concatenated and passed into one final dense layer.

The multi-head attention layer output is added with the original embeddings before undergoing layer normalization. The residual connection and layer normalization both help to avoid vanishing gradients issues. Then, a feed-forward network with reLu activation is applied to keep the output of the encoder block in the same dimensions as the inputs. Finally, the feed-forward network output is added with the multi-head attention layer output before undergoing layer normalization. The encoder models use an Adam optimizer with a custom learning rate scheduler to improve learning. The specifics of the transformer block are located in the appendix figure A-5a.

5.3 Time Series Transformer Encoders (T)

Several modifications are made to fine-tune the transformer encoder for time series data. First, an adjustable positional encoding Time2Vec [11] for time series is used in place of a fixed sinusoidal positional encoding function. Second, batch normalization is used instead of layer normalization since the input lengths n are fixed in training. The specifics of the transformer block are located in the appendix figure A-5b.

5.4 Results & Discussion

Various attention mechanisms were applied to LSTM-NDT and TadGAN models. In the case of Bahdanau Attention, the attention layer is applied directly to the hidden state output sequences of the LSTM layer. In the case of transformer encoders and time series transformer encoders, those layers were used in place of LSTM layers.

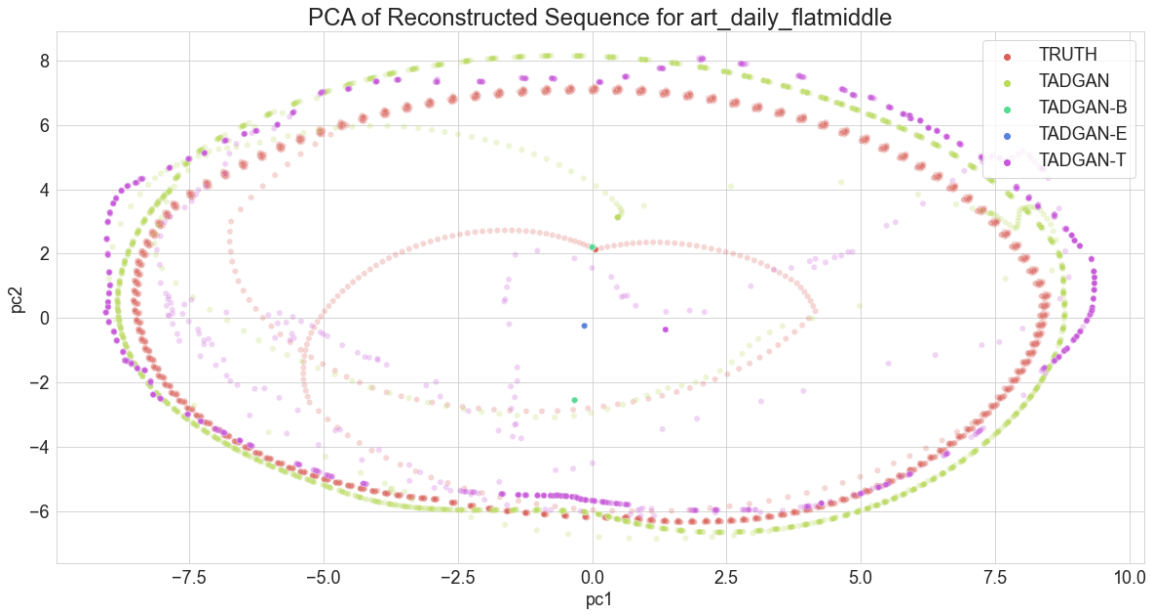

Figure 5-1: `art_daily_flatmiddle` signal from `artificialWithAnomaly` dataset shows mode collapse of reconstructed outputs from TadGAN-B and TadGAN-E attention variations.

5.4.1 Attention-based LSTM-NDT

No significant improvements with attention mechanisms

Results in appendix table B.11 shows that the attention-based models either perform similarly or worse than the base model without attention. While the transformer encoder version LSTM-NDT-1-E performs similarly to the base LSTM-NDT, the training time nearly doubled even with the support of GPUs. Moreover, the models with Bahdanau attention LSTM-NDT-B and time series transformer encoder LSTM-NDT-T showed a significant drop in average F1 scores. There could be more noise when considering the entirety of the hidden states, leading to inadequate training of parameters for the attention mechanisms.

5.4.2 Attention-based TadGAN

Mode collapse with certain attention mechanisms

Results in appendix table B.12 also support the idea that models with the attention mechanism did not offer many benefits compared to the LSTM variation. While the average F1 scores dropped for every attention variation, the training time was significantly reduced using the transformer encoder in TadGAN-E. Figure 5-1, which shows the PCA of reconstructed outputs, gives insight into the performance decrease in attention variations. While TadGAN's reconstructed outputs resemble the truth outputs, TadGAN-B and TadGAN-E's reconstructed outputs converged into a single point in the plot. This behavior suggests mode collapse issues since those models could only produce one output type. While TadGAN-T was able to produce outputs similar to TadGAN, other unexplained factors caused the drop in performance.

Duality of critics and generators

Despite the mode collapse issues, results in appendix table B.12 show that F1 scores of TadGAN-E only dropped around ~ 0.07 compared to TadGAN. To investigate this further, figure 5-2 plots the breakdown of the anomaly scores created from reconstruction and critic scores for TadGAN and TadGAN-E. While both models produce similar combined scores that successfully identify the contextual anomaly, reconstruction and critic scores contributed to the combined score differently. For example, in TadGAN (green) without mode collapse issues, the reconstruction scores peaked at the contextual anomaly while the critic scores showed non-informative cyclic patterns. This finding is consistent with Geiger et al., which finds instability when critic scores are used alone as an anomaly score. On the other hand, in TadGAN-E (purple) with mode collapse issues, the critic scores spiked at the contextual anomaly while the reconstruction scores showed non-informative patterns. This behavior occurs because the min-max optimization framework of TadGAN prioritizes the optimization of either the generators or the critics. Figure 5-3 shows the training loss for the critic x (cx), critic z (cz), and encoder-generator (eg) for both TadGAN and TadGAN-E.

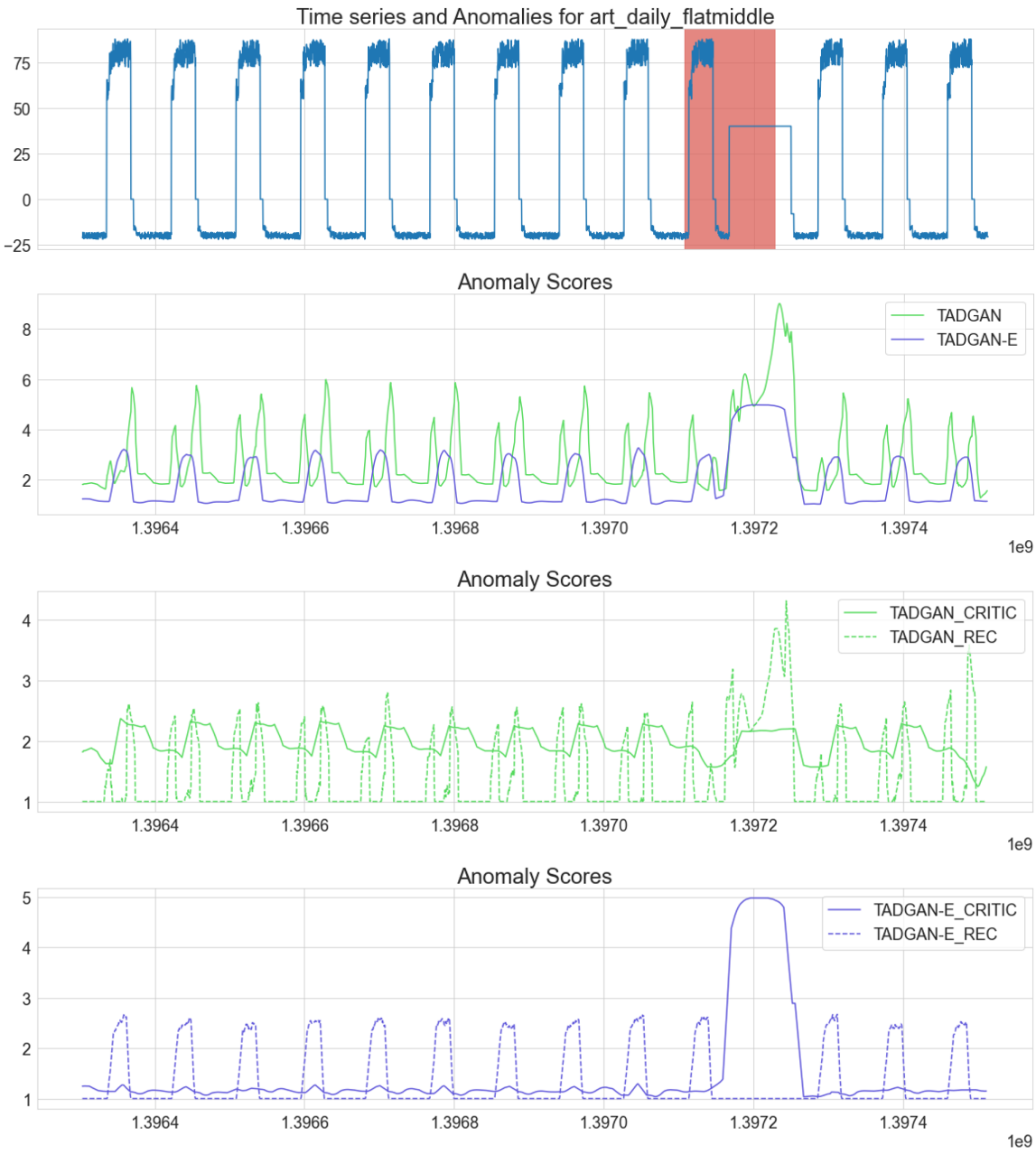

Figure 5-2: `art_daily_flatmiddle` signal from `artificialWithAnomaly` dataset shows the breakdown of reconstruction and critic anomaly scores for TadGAN (green) and TadGAN-E (purple).

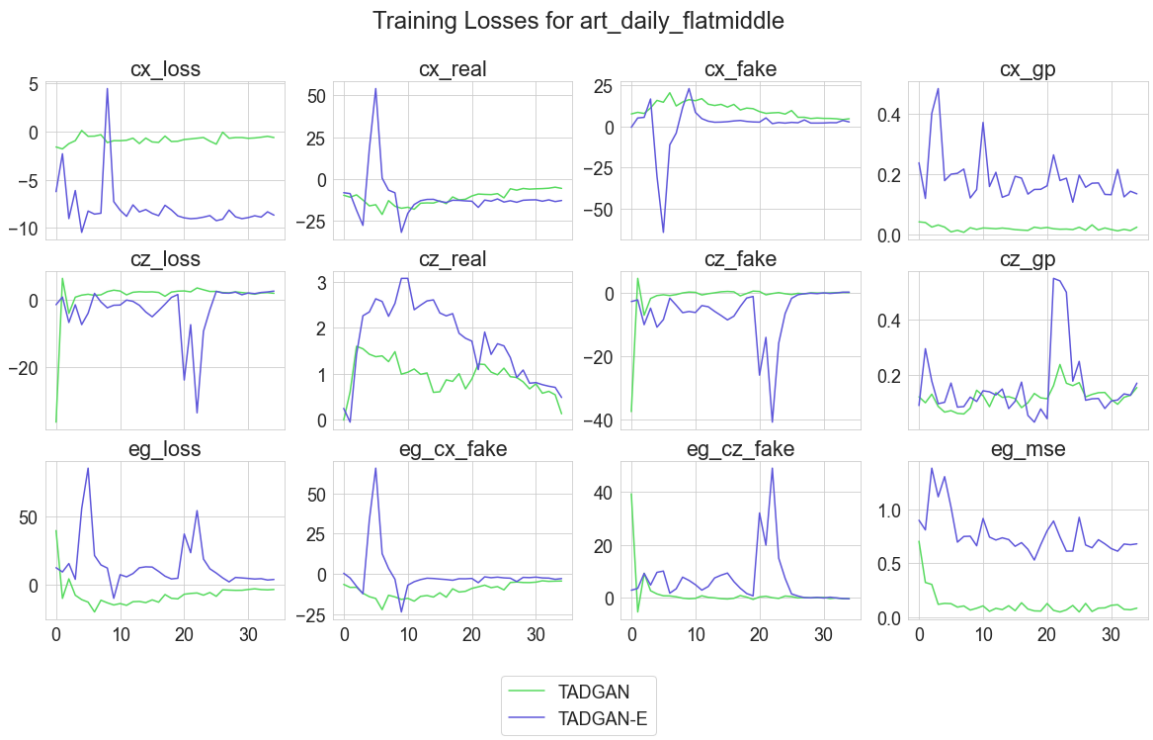

Figure 5-3: `art_daily_flatmiddle` signal from `artificialWithAnomaly` dataset shows the training loss for TadGAN (green) and TadGAN-E (purple).

TadGAN-E favors optimizing critic x as the model ended with a lower critic x loss `cx_loss` than TadGAN. On the other hand, TadGAN favors optimizing the generator as the model ended with a lower reconstructed mean squared error loss `eg_mse` than TadGAN-E. The lower average F1 scores for mode collapse models suggest that prioritizing reconstruction scores is better than prioritizing critic scores for identifying anomalies.

Chapter 6

Future Work

6.1 Summary

This study introduces the idea of masking anomaly scores, bi-directional regression for prediction-based methods, and autoencoder with regression (AER) to overcome identified limitations and leverage successes from existing methods, as summarized in table 6.1. In addition, this study also explored the effect of replacing LSTM with attention-based architectures on the average contextual F1 scores of existing methods. While the attention-based alternatives offered no improvement in the average F1 scores, the results unintentionally provided insights into the duality of critics and generators in the TadGAN architecture.

While combining prediction-based and reconstruction-based anomaly scores in the AER model offers additional insights for anomaly detection, there are still many improvements and unanswered questions. This chapter divides the improvements and answered questions into several sections: AER architectures, attention mechanisms, dataset re-evaluation, and multivariate datasets.

6.2 AER Architectures

The architecture of autoencoder with regression models can be improved since it is currently based on naive biLSTM-AE and LSTM-NDT models. The idea can

Method	ID	Limitations	Corrected in AER by
Prediction	PL1	High anomaly scores at the start result in false positives	Masking Anomaly Scores
	PL2	Low anomaly scores for contextual anomalies with simple patterns	MULT Combination Method
	PL3	Missing predictions at the start of the time series	Bi-directional Regression
	PL4	Short-sightedness produces many false positives	MULT Combination Method
Reconstruction	RL1	Reconstruction-based scores reduce peaks for point anomalies	PRED Combination Method
	RL2	Bad train-test splits result in mode collapse	Unresolved
Method	**ID**	**Successes**	**Leveraged in AER via**
Prediction	PS1	Prediction-based anomaly scores better capture point anomalies	PRED Combination Method
Reconstruction	RS1	DTW scores are better at capturing anomalies	DTW Reconstruction Errors
	RS2	Reconstruction-based anomaly scores better capture contextual anomalies	MULT Combination Method
	RS3	Reconstruction-based anomaly scores reduce false positives	MULT Combination Method

Table 6.1: Summary of successes and limitations of existing prediction-based and reconstruction-based methods.

be extended to any regression-based or reconstruction-based model with minimum changes to the objective function and the architecture. Also, more experimentation is needed to determine the optimal γ controlling the contribution of prediction and reconstruction loss.

Another research question centers around determining the optimal combination of anomaly scores without using the ground truth labels. The current methodology requires prior knowledge about the anomaly type (e.g., point or contextual) contained in the signal to determine the optimal combination of anomaly scores. A solution could be to employ a consensus schema that only accepts the anomaly sequence if three or more combination methods agree on the sequence being anomalous. However, there might be more efficient methods to combine the anomaly scores. For example,

relaxing to a semi-supervised setting, the labeled data could be used to fine-tune the combination of anomaly scores for the dataset.

6.3 Attention Mechanisms

Despite not providing many benefits in this study, the attention mechanisms still have room for improvement. For instance, the transformer encoder layer is a valid alternative to LSTM layers as the length and dimensionality of the input increase. More research in this direction may also provide further insights into how to minimize the duality of critics and generators in TadGAN models. The combination of critic-based and reconstruction-based anomaly scores might be more informative if critics and generators are equally well-trained.

6.4 Dataset Re-evaluation

The issue with the bad train-test split in at least one signal of the `MSL` dataset remains unresolved (RL2). For instance, the `C-2` signal from the univariate `MSL` dataset has a train split consisting of only -1 values. In contrast, the test split consists of patterns with anomalies unrelated to the train split. Therefore, the models will not learn meaningful information in order to find anomalies in such signals. The univariate datasets `MSL` and `SMAP` in the `Orion` benchmark library should be re-evaluated since some information was lost in the conversion from multivariate to univariate datasets.

6.5 Multivariate Datasets

The performance of many-to-one models depends greatly on the target channel. Hence, the performance of these models suffers in the case of multivariate datasets where the target channel is not apparent, as seen in the `SMD` dataset. A solution could be to train a many-to-one model for each channel; however, this approach can be costly depending on the complexity of the model and the number of channels in

the input. Another solution could be to forecast the values of every channel in the next step for prediction-based models or to reconstruct the entire input with every channel for reconstruction-based models. However, this approach requires modifying both the objective function and the method to calculate anomaly scores, especially in the case of DTW.

Appendix A

Figures

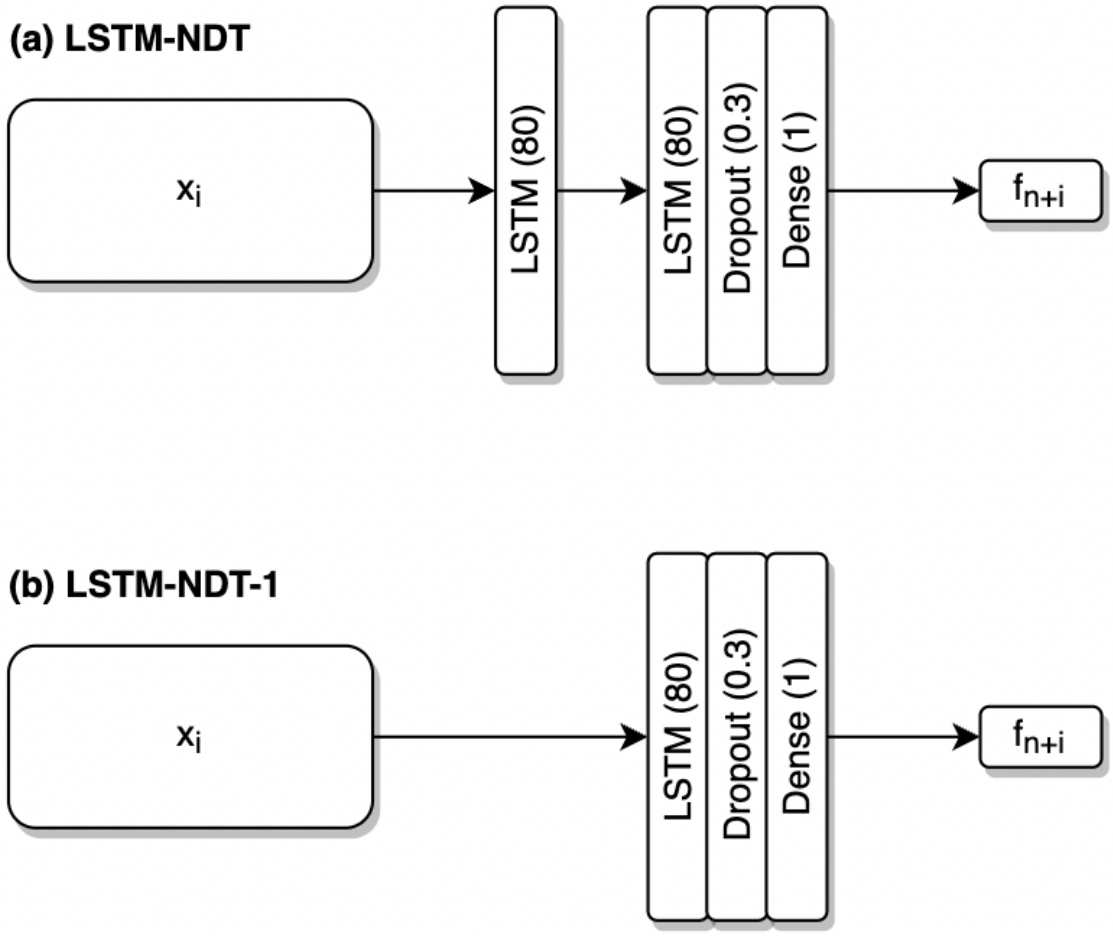

Figure A-1: The architecture of the prediction-based models. (a) LSTM-NDT is the original two-layer LSTM model. (b) LSTM-NDT-1 is the one-layer version with fewer parameters.

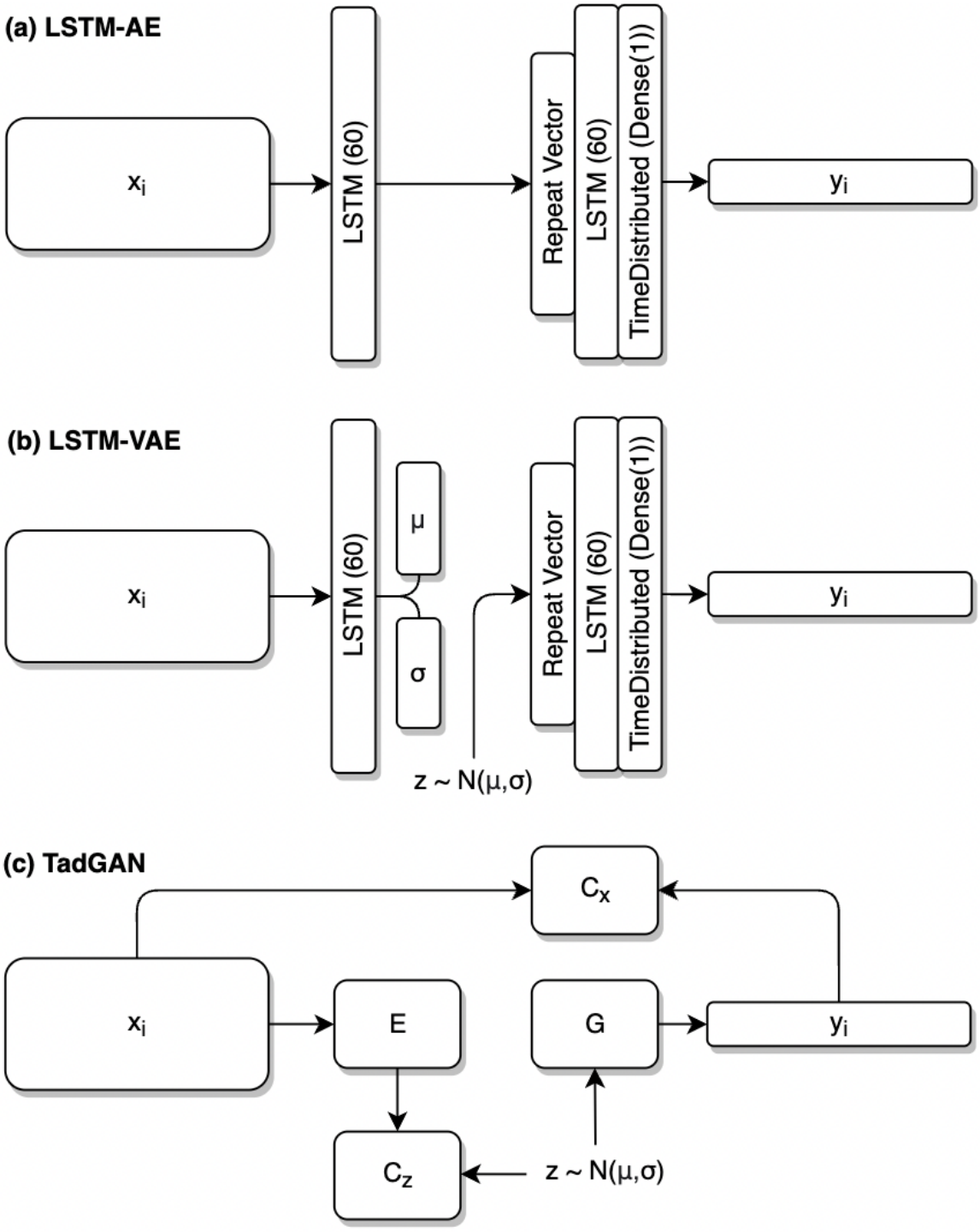

Figure A-2: The architecture of the reconstruction-based models. (a) LSTM-AE is the autoencoder model with LSTM layers. (b) LSTM-VAE is the variational autoencoder model with LSTM layers. (c) TadGAN consists of an encoder E, generator G, critic C_x, and critic C_z.

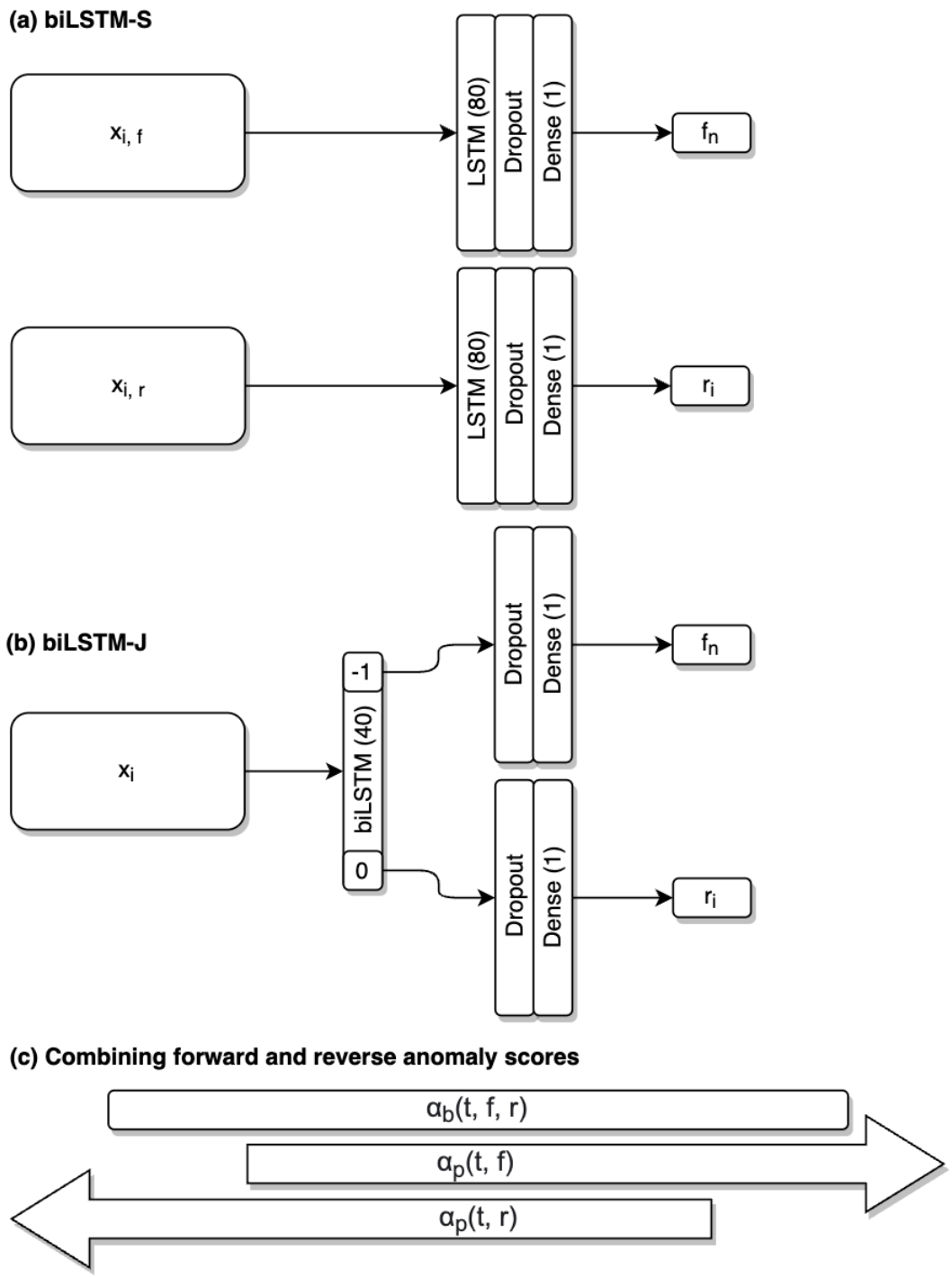

Figure A-3: The architecture of bi-directional regression models and methodology to combine forward and reverse anomaly scores. (a) biLSTM-S consists of two separate LSTM-NDT models trained on the forward and reversed input sequences. (b) biLSTM-J is one joint model trained to forecast in both directions simultaneously. (c) Visualization of the method to combine forward and reverse anomaly scores.

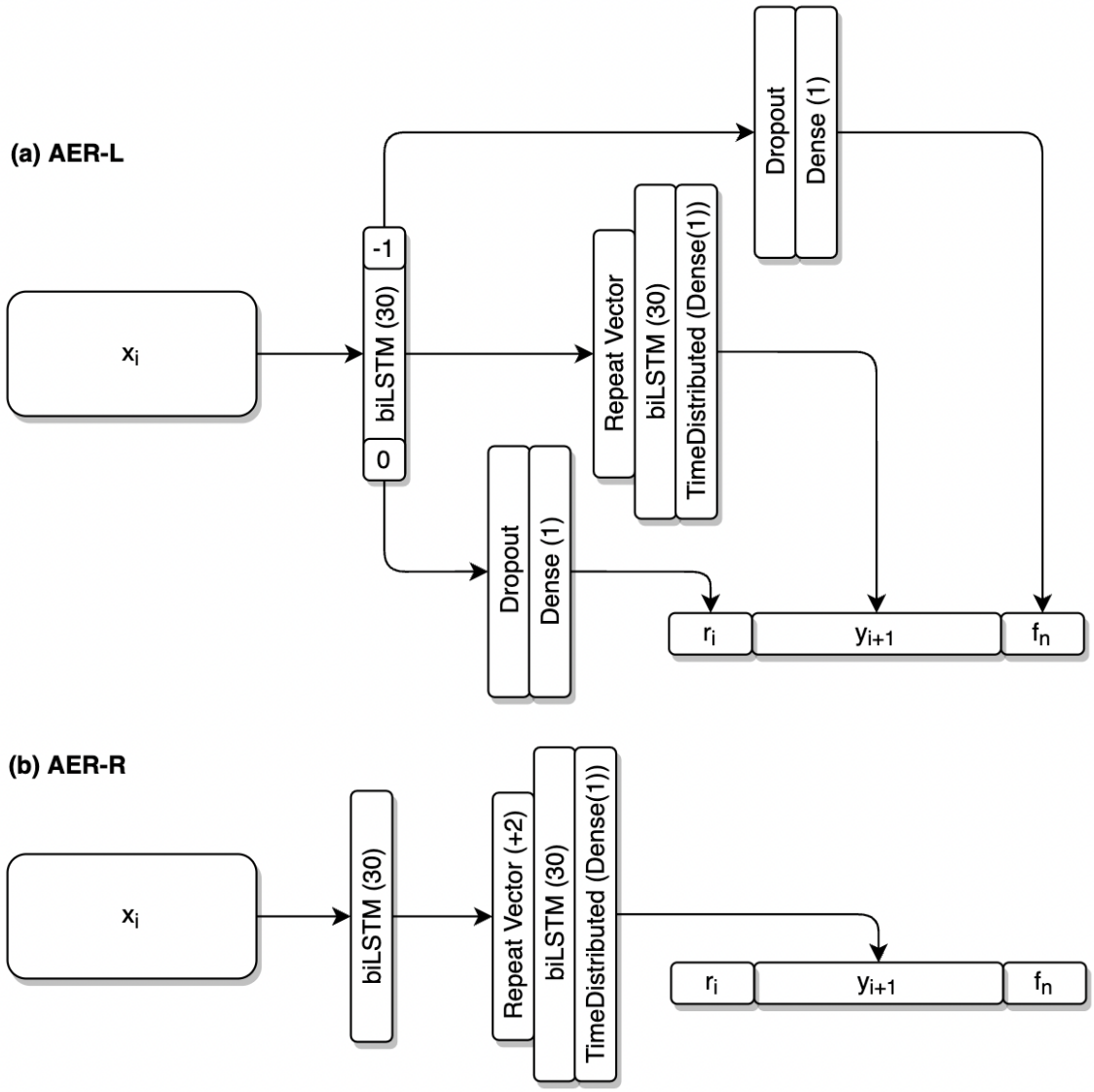

Figure A-4: Joint models inspired by LSTM autoencoders and shared bi-directional LSTM regressors. (a) AER-L uses the latent space to forecast values and uses the same latent space as inputs to the decoder. (b) AER-R increases the number of repeat vectors to reconstruct the input sequences and to forecast values in both directions.

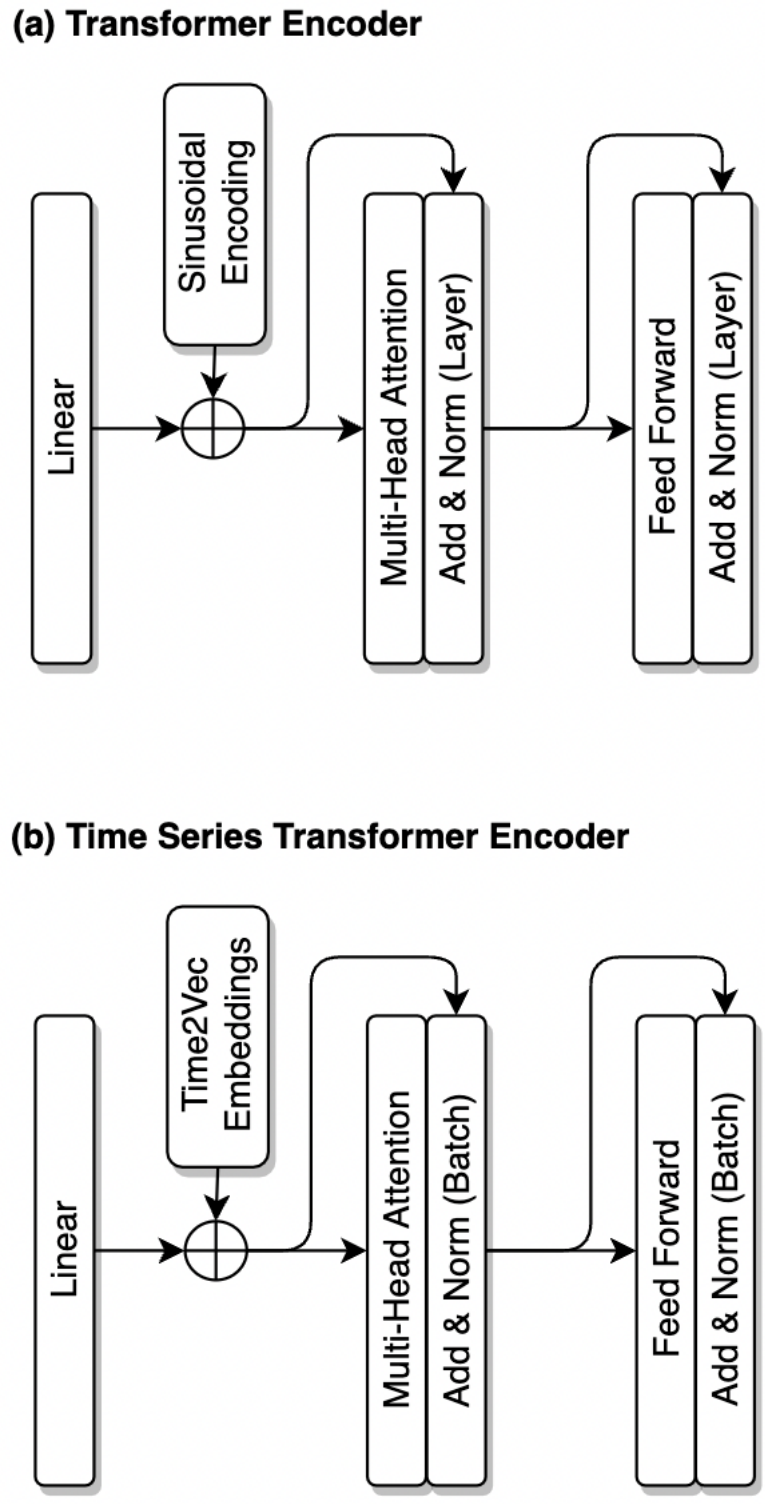

Figure A-5: The architecture of attention encoders. (a) Transformer encoder used for natural language processing tasks. (b) Time series transformer encoder adapted for time series.

Appendix B

Tables

t	values of a time series
T	total number of observations in time series
n	length of input x
d	dimensionality of the input x
x	input for all models
y	output for reconstruction-based models
f	forward predicted values for prediction-based models
r	reverse predicted values for prediction-based models
m	number of observations from the start to mask
α_p	prediction-based anomaly scores
α_b	bi-directional prediction-based anomaly scores
$\alpha_{r,p}$	reconstruction-based anomaly scores created from point-wise differencing
$\alpha_{r,a}$	reconstruction-based anomaly scores created from area differencing
$\alpha_{r,d}$	reconstruction-based anomaly scores created from dynamic time warping
α_c	prediction-reconstruction mixture anomaly scores

Table B.1: List of notations.

Univariate Datasets		**ARIMA**	**LSTM -NDT**	**LSTM -NDT-1**
NASA	MSL	0.4421	**0.5155**	0.4808
	SMAP	0.3333	**0.7075**	0.6892
YAHOO	A1	**0.7332**	0.7206	0.7325
	A2	0.8074	**0.9801**	0.9704
	A3	**0.8176**	0.7442	0.7794
	A4	**0.6998**	0.6379	0.6933
NAB	Art	0.3529	**0.4000**	**0.4000**
	AWS	0.5176	0.5128	**0.5312**
	AdEx	0.7407	0.7407	**0.7857**
	Traffic	0.5000	**0.6667**	0.5946
	Tweets	0.5672	0.5797	**0.6000**
Avg. F1 ($\mu \pm \sigma$)		0.5920 $\pm$ 0.1767	0.6551 $\pm$ 0.1545	**0.6597 $\pm$ 0.1606**
Avg. Sec ($\mu \pm \sigma$)		44 $\pm$ 110	17 $\pm$ 8	10 $\pm$ 4
Total Precision		0.8332	0.8495	**0.8555**
Total Recall		0.6552	0.6177	**0.6603**
Total F1		0.7336	0.7153	**0.7453**
Multivariate Datasets		**ARIMA**	**LSTM -NDT**	**LSTM -NDT-1**
NASA	MSL	-	0.5111	**0.5366**
	SMAP	-	**0.6623**	0.6023
INET	SMD	-	0.1397	**0.1547**
Avg. F1 ($\mu \pm \sigma$)		-	**0.4377 $\pm$ 0.2689**	0.4312 $\pm$ 0.2417
Avg. Sec ($\mu \pm \sigma$)		-	70 $\pm$ 80	39 $\pm$ 42
Total Precision		-	0.5163	**0.5299**
Total Recall		-	0.1602	**0.1757**
Total F1		-	**0.2689**	0.2417
UCR Dataset		**ARIMA**	**LSTM -NDT**	**LSTM -NDT-1**
UCR	UCR	0.1236	0.3908	**0.4164**
Total Precision		0.2075	0.3315	**0.3631**
Total Recall		0.088	0.476	**0.488**
Avg. Sec ($\mu \pm \sigma$)		335 $\pm$ 539	144 $\pm$ 173	86 $\pm$ 106

Table B.2: Results for all prediction-based baseline models.

Univariate Datasets		**LSTM -AE -PD**	**LSTM -AE -AD**	**LSTM -AE -DTW**	**LSTM -VAE -PD**	**LSTM -VAE -AD**	**LSTM -VAE -DTW**
NASA	MSL	0.4533	**0.48**	0.4571	0.4304	0.3951	**0.4324**
	SMAP	0.695	0.6667	**0.7259**	0.5874	0.6027	**0.6056**
YAHOO	A1	0.6279	0.5723	**0.6391**	0.5856	0.5865	**0.6171**
	A2	0.8673	0.8821	**0.961**	0.8297	0.8246	**0.9124**
	A3	0.4381	0.3383	**0.5938**	0.4666	0.3706	**0.5977**
	A4	0.2412	0.1513	**0.3678**	0.2303	0.1624	**0.3263**
NAB	Art	0.5455	**0.6154**	0.4444	**0.5455**	0.5333	0.4444
	AWS	**0.7241**	0.6667	0.6786	**0.7119**	0.6364	0.7018
	AdEx	0.5833	**0.7692**	0.6429	0.64	0.6923	**0.7586**
	Traffic	0.5333	0.5	**0.5926**	0.5517	0.4828	**0.5926**
	Tweets	0.5333	0.5	**0.5357**	0.5172	0.5	**0.5357**
Avg. F1 ($\mu \pm \sigma$)		0.5675 ± 0.1654	0.5584 ± 0.2008	**0.6035 ± 0.1607**	0.5542 ± 0.1548	0.5261 ± 0.1774	**0.5931 ± 0.1626**
Avg. Sec ($\mu \pm \sigma$)		13 ± 5	13 ± 5	13 ± 5	14 ± 7	14 ± 7	14 ± 7
Total Precision		0.7602	0.7242	**0.8466**	0.7026	0.6801	**0.8011**
Total Recall		0.3414	0.2805	**0.4347**	0.3521	0.2942	**0.427**
Total F1		0.4712	0.4044	**0.5744**	0.4691	0.4107	**0.5571**
Multivariate Datasets		**LSTM -AE -PD**	**LSTM -AE -AD**	**LSTM -AE -DTW**	**LSTM -VAE -PD**	**LSTM -VAE -AD**	**LSTM -VAE -DTW**
NASA	MSL	**0.5067**	**0.5067**	0.4857	0.4675	0.4267	**0.4789**
	SMAP	0.6286	**0.6383**	0.6187	**0.6383**	0.6241	0.6056
INET	SMD	0.139	0.1303	**0.144**	0.1487	0.1434	**0.1489**
Avg. F1 ($\mu \pm \sigma$)		0.4248 ± 0.2549	**0.4251 ± 0.2636**	0.4161 ± 0.2449	**0.4182 ± 0.2485**	0.3981 ± 0.2416	0.4111 ± 0.2358
Avg. Sec ($\mu \pm \sigma$)		48 ± 51	48 ± 51	48 ± 51	54 ± 55	54 ± 55	54 ± 55
Total Precision		**0.5583**	0.5355	0.5644	0.5561	0.5377	**0.5577**
Total Recall		**0.1519**	0.1493	0.1506	**0.1572**	0.1506	0.1532
Total F1		**0.2388**	0.2335	0.2377	**0.2451**	0.2353	0.2404
UCR Dataset		**LSTM -AE -PD**	**LSTM -AE -AD**	**LSTM -AE -DTW**	**LSTM -VAE -PD**	**LSTM -VAE -AD**	**LSTM -VAE -DTW**
UCR	UCR	0.3135	0.2879	**0.3292**	0.3396	0.2992	**0.354**
Total Precision		0.2318	0.2095	**0.2384**	0.255	0.2227	**0.2643**
Total Recall		0.484	0.46	**0.532**	0.508	0.456	**0.536**
Avg. Sec ($\mu \pm \sigma$)		135 ± 193	135 ± 193	135 ± 193	138 ± 194	138 ± 194	138 ± 194

Table B.3: Results for LSTM-AE and LSTM-VAE models with point-wise differencing (PD), area differencing (AD), and dynamic time warping (DTW) reconstruction-based anomaly scores.

Univariate Datasets		**LSTM -AE -DTW**	**LSTM -VAE -DTW**	**TadGAN -TF1**	**TadGAN -TF2**
NASA	MSL	0.4571	0.4324	**0.5714**	0.5843
	SMAP	**0.7259**	0.6056	0.5976	0.6133
YAHOO	A1	**0.6391**	0.6171	0.5849	0.5325
	A2	**0.961**	0.9124	0.8612	0.8421
	A3	0.5938	**0.5977**	0.4109	0.3907
	A4	0.3678	0.3263	0.3284	0.2969
NAB	Art	0.4444	0.4444	**0.6667**	0.5714
	AWS	0.6786	**0.7018**	0.6557	0.6774
	AdEx	0.6429	0.7586	**0.8333**	0.72
	Traffic	0.5926	0.5926	**0.6061**	0.5806
	Tweets	0.5357	0.5357	**0.5882**	**0.5882**
Avg. F1 ($\mu \pm \sigma$)		0.6035 ± 0.1607	0.5931 ± 0.1626	0.6095 ± 0.1549	**0.5816 ± 0.1474**
Avg. Sec ($\mu \pm \sigma$)		13 ± 5	14 ± 7	73 ± 27	145 ± 55
Total Precision		**0.8466**	0.8011	0.6755	0.6469
Total Recall		**0.4347**	0.427	0.3687	0.3487
Total F1		**0.5744**	0.5571	0.477	0.4531
Multivariate Datasets		**LSTM -AE -DTW**	**LSTM -VAE -DTW**	**TadGAN -TF1**	**TadGAN -TF2**
NASA	MSL	0.4857	0.4789	**0.557**	0.5238
	SMAP	0.6187	0.6056	**0.641**	0.5625
INET	SMD	0.144	**0.1489**	0.1078	0.131
Avg. F1 ($\mu \pm \sigma$)		0.4161 ± 0.2449	0.4111 ± 0.2358	**0.4353 ± 0.2867**	0.4058 ± 0.2387
Avg. Sec ($\mu \pm \sigma$)		48 ± 51	54 ± 55	104 ± 28	829 ± 1032
Total Precision		**0.5644**	0.5577	0.5091	0.4936
Total Recall		0.1506	**0.1532**	0.148	0.1532
Total F1		0.2377	**0.2404**	0.2293	0.2339
UCR Dataset		**LSTM -AE -DTW**	**LSTM -VAE -DTW**	**TadGAN -TF1**	**TadGAN -TF2**
UCR	UCR	0.3292	**0.354**	0.1501	0.1625
Total Precision		0.2384	**0.2643**	0.0989	0.1071
Total Recall		0.532	**0.536**	0.312	0.336
Avg. Sec ($\mu \pm \sigma$)		135 ± 193	138 ± 194	855 ± 1300	1898 ± 2876

Table B.4: Results for all reconstruction-based baseline models.

Univariate Datasets		**ARIMA**	**ARIMA -M**	**LSTM -NDT**	**LSTM -NDT -M**	**LSTM -NDT-1**	**LSTM -NDT-1 -M**
NASA	MSL	0.4421	**0.4565**	0.5155	**0.5208**	0.4808	**0.4854**
	SMAP	0.3333	**0.359**	0.7075	**0.7536**	0.6892	**0.7391**
YAHOO	A1	0.7332	**0.7519**	0.7206	**0.7287**	0.7325	**0.7407**
	A2	0.8074	**0.8093**	0.9801	**0.9875**	0.9704	**0.9875**
	A3	**0.8176**	0.8069	**0.7442**	0.7335	**0.7794**	0.7762
	A4	**0.6998**	0.6899	**0.6379**	0.6378	0.6933	**0.694**
NAB	Art	0.3529	**0.5**	0.4	**0.6**	0.4	**0.6**
	AWS	0.5176	0.5176	0.5128	0.5128	0.5312	0.5312
	AdEx	0.7407	**0.7692**	0.7407	**0.7692**	0.7857	**0.8148**
	Traffic	0.5	0.5	0.6667	**0.6857**	0.5946	**0.6111**
	Tweets	0.5672	**0.5758**	0.5797	**0.5882**	0.6	**0.6087**
Avg. F1 ($\mu \pm \sigma$)		0.592 ± 0.1767	**0.6124 ± 0.1584**	0.6551 ± 0.1545	**0.6834 ± 0.1353**	0.6597 ± 0.1606	**0.6899 ± 0.143**
Avg. Sec ($\mu \pm \sigma$)		44 ± 110	44 ± 110	17 ± 8	17 ± 8	10 ± 4	10 ± 4
Total Precision		0.8332	**0.8453**	0.8495	**0.8649**	0.8555	**0.8741**
Total Recall		**0.6552**	0.6445	**0.6177**	0.6105	**0.6603**	0.6564
Total F1		**0.7336**	0.7314	0.7153	**0.7157**	0.7453	**0.7498**
Multivariate Datasets		**ARIMA**	**ARIMA -M**	**LSTM -NDT**	**LSTM -NDT -M**	**LSTM -NDT-1**	**LSTM -NDT-1 -M**
NASA	MSL	-	-	0.5111	**0.5287**	0.5366	**0.5432**
	SMAP	-	-	0.6623	**0.6892**	0.6023	**0.6235**
INET	SMD	-	-	0.1397	**0.14**	0.1547	**0.1551**
Avg. F1 ($\mu \pm \sigma$)		-	-	0.4377 ± 0.2689	**0.4526 ± 0.2824**	0.4312 ± 0.2417	**0.4406 ± 0.2505**
Avg. Sec ($\mu \pm \sigma$)		-	-	70 ± 80	70 ± 80	39 ± 42	39 ± 42
Total Precision		-	-	0.5163	**0.5404**	0.5299	**0.5496**
Total Recall		-	-	0.1678	0.1678	0.1757	0.1757
Total F1		-	-	0.2532	**0.256**	0.2639	**0.2663**
UCR Dataset		**ARIMA**	**ARIMA -M**	**LSTM -NDT**	**LSTM -NDT -M**	**LSTM -NDT-1**	**LSTM -NDT-1 -M**
UCR	UCR	0.1236	**0.1481**	0.3908	**0.4461**	0.4164	**0.472**
Total Precision		0.2075	**0.4681**	0.3315	**0.4229**	0.3631	**0.4569**
Total Recall		0.088	0.088	**0.476**	0.472	0.488	0.488
Avg. Sec ($\mu \pm \sigma$)		335 ± 539	335 ± 539	144 ± 173	144 ± 173	86 ± 106	86 ± 106

Table B.5: Results after masking prediction-based anomaly scores (denoted by M).

Univariate Datasets		**LSTM -AE -DTW**	**LSTM -AE -DTW-M**	**LSTM -VAE -DTW**	**LSTM -VAE -DTW-M**	**TadGAN -TF2**	**TadGAN -TF2 -M**
NASA	MSL	0.4571	0.4571	0.4324	**0.4384**	0.5843	0.5843
	SMAP	0.7259	**0.7717**	0.6056	**0.7049**	0.6133	**0.6301**
YAHOO	A1	0.6391	**0.6448**	0.6171	**0.6292**	0.5325	**0.5342**
	A2	0.961	**0.9633**	0.9124	**0.9406**	0.8421	**0.8462**
	A3	0.5938	0.5938	**0.5977**	0.5946	0.3907	**0.395**
	A4	**0.3678**	0.367	0.3263	**0.3279**	**0.2969**	0.2913
NAB	Art	0.4444	0.4444	0.4444	0.4444	0.5714	**0.6154**
	AWS	0.6786	0.6786	0.7018	**0.7143**	0.6774	0.6774
	AdEx	0.6429	0.6429	0.7586	0.7586	0.72	0.72
	Traffic	0.5926	**0.6154**	0.5926	**0.6154**	0.5806	0.5806
	Tweets	0.5357	0.5357	0.5357	0.5357	0.5882	0.5882
Avg. F1 ($\mu \pm \sigma$)		0.6035 ± 0.1607	**0.6104 ± 0.1654**	0.5931 ± 0.1626	**0.6095 ± 0.1714**	0.5816 ± 0.1474	**0.5875 ± 0.1493**
Avg. Sec ($\mu \pm \sigma$)		13 ± 5	13 ± 5	14 ± 7	14 ± 7	145 ± 55	145 ± 55
Total Precision		0.8466	**0.8571**	0.8011	**0.8293**	0.6469	**0.6602**
Total Recall		**0.4347**	0.4342	0.427	**0.4261**	**0.3487**	0.3467
Total F1		0.5744	**0.5764**	0.5571	**0.563**	0.4531	**0.4546**
Multivariate Datasets		**LSTM -AE -DTW**	**LSTM -AE -DTW-M**	**LSTM -VAE -DTW**	**LSTM -VAE -DTW-M**	**TadGAN -TF2**	**TadGAN -TF2 -M**
NASA	MSL	0.4857	0.4857	0.4789	**0.4928**	0.5238	0.5238
	SMAP	0.6187	**0.6418**	0.6056	**0.6825**	0.5625	**0.5882**
INET	SMD	0.144	0.144	0.1489	**0.1491**	0.131	0.131
Avg. F1 ($\mu \pm \sigma$)		0.4161 ± 0.2449	**0.4238 ± 0.2546**	0.4111 ± 0.2358	**0.4415 ± 0.2704**	0.4058 ± 0.2387	**0.4143 ± 0.2475**
Avg. Sec ($\mu \pm \sigma$)		48 ± 51	48 ± 51	54 ± 55	54 ± 55	829 ± 1032	829 ± 1032
Total Precision		0.5644	**0.5787**	0.5577	**0.6138**	0.4936	**0.5066**
Total Recall		0.1506	0.1506	0.1532	0.1532	0.1532	**0.1534**
Total F1		0.2377	**0.239**	0.2404	**0.2452**	0.2339	**0.2355**
UCR Dataset		**LSTM -AE -DTW**	**LSTM -AE -DTW-M**	**LSTM -VAE -DTW**	**LSTM -VAE -DTW-M**	**TadGAN -TF2**	**TadGAN -TF2 -M**
UCR	UCR	0.3292	**0.3333**	0.354	**0.3607**	0.1625	**0.1642**
Total Precision		0.2384	**0.2427**	0.2643	**0.2718**	0.1071	**0.1087**
Total Recall		0.532	0.532	0.536	0.536	0.336	0.336
Avg. Sec ($\mu \pm \sigma$)		135 ±193	135 ±193	138 ± 194	138 ± 194	1898 ± 2876	1898 ± 2876

Table B.6: Results after masking reconstruction-based anomaly scores (denoted by M).

Univariate Datasets		**ARIMA -M**	**LSTM -NDT-1 -M**	**biLSTM -S-M**	**biLSTM -J-M**
NASA	MSL	0.4565	0.4854	**0.5049**	0.4898
	SMAP	0.359	**0.7391**	0.662	0.6853
YAHOO	A1	0.7519	0.7407	0.755	**0.7581**
	A2	0.8093	**0.9875**	0.9494	0.9448
	A3	0.8069	0.7762	0.8947	**0.9123**
	A4	0.6899	0.694	0.7917	**0.8061**
NAB	Art	0.5	**0.6**	0.5455	0.5455
	AWS	0.5176	**0.5312**	0.4884	0.5116
	AdEx	0.7692	**0.8148**	0.7857	0.7407
	Traffic	0.5	0.6111	**0.6842**	0.6667
	Tweets	0.5758	**0.6087**	0.5867	0.5753
Avg. F1 (μ ± σ)		0.6124 ± 0.1584	0.6899 ± 0.143	**0.6953 ± 0.1549**	0.6942 ± 0.1554
Avg. Sec (μ ± σ)		44 ± 110	10 ± 4	19 ± 9	18 ± 9
Total Precision		0.8453	**0.8741**	0.8688	0.8739
Total Recall		0.6445	0.6564	0.7782	**0.7969**
Total F1		0.7314	0.7498	0.821	**0.8337**
Multivariate Datasets		**ARIMA -M**	**LSTM -NDT-1 -M**	**biLSTM -S-M**	**biLSTM -J-M**
NASA	MSL	-	**0.5432**	0.4632	0.4731
	SMAP	-	0.6235	0.6258	**0.6329**
INET	SMD	-	**0.1551**	0.1415	0.1444
Avg. F1 (μ ± σ)		-	**0.4406 ± 0.2505**	0.4102 ± 0.2465	0.4168 ± 0.2491
Avg. Sec (μ ± σ)		-	39 ± 42	93 ± 92	71 ± 77
Total Precision		-	**0.5496**	0.504	0.5207
Total Recall		-	**0.1757**	0.1664	0.1664
Total F1		-	**0.2663**	0.2502	0.2523
UCR Dataset		**ARIMA -M**	**LSTM -NDT-1 -M**	**biLSTM -S-M**	**biLSTM -J-M**
UCR	UCR	0.1481	**0.472**	0.3293	0.3378
Total Precision		**0.4681**	0.4569	0.266	0.2697
Total Recall		0.088	**0.488**	0.432	0.452
Avg. Sec (μ ± σ)		335 ± 539	86 ± 106	380 ± 592	387 ± 564

Table B.7: Results of baseline prediction-based models compared to separate (S) and joint (J) bi-directional models. Masking (M) is applied to all models.

Univariate Datasets		**ARIMA -M**	**AER-S -M**	**AER-S -PRED -M**	**AER-S -SUM -M**	**AER-S -REC -M**	**AER-S -MULT -M**
NASA	MSL	0.4565	**0.5714**	0.4898	0.4872	0.4571	**0.5714**
	SMAP	0.359	**0.7717**	0.6853	0.6712	**0.7717**	0.7705
YAHOO	A1	0.7519	**0.8159**	0.7581	0.7534	0.6448	**0.8159**
	A2	0.8093	**0.9657**	0.9448	0.9471	0.9633	**0.9657**
	A3	0.8069	**0.9123**	**0.9123**	0.745	0.5938	0.7822
	A4	0.6899	**0.8061**	**0.8061**	0.5733	0.367	0.6217
NAB	Art	0.5	**0.5455**	**0.5455**	0.5	0.4444	0.5
	AWS	0.5176	**0.7273**	0.5116	**0.7273**	0.6786	0.6154
	AdEx	0.7692	0.7407	**0.7407**	0.7333	0.6429	0.6429
	Traffic	0.5	**0.6667**	**0.6667**	0.5806	0.6154	0.6061
	Tweets	0.5758	**0.6269**	0.5753	0.5846	0.5357	**0.6269**
Avg. F1 (μ ± σ)		0.6124 ± 0.1584	**0.7409 ± 0.1328**	**0.6942 ± 0.1554**	0.6639 ± 0.1354	0.6104 ± 0.1654	0.6835 ± 0.1342
Avg. Sec (μ ± σ)		44 ± 110	-	-	-	-	-
Total Precision		0.8453	**0.9249**	0.8739	0.8715	0.8571	**0.9052**
Total Recall		0.6445	**0.7918**	**0.7969**	0.5862	0.4342	0.6258
Total F1		0.7314	**0.8532**	**0.8337**	0.7009	0.5764	0.74
Multivariate Datasets		**ARIMA -M**	**AER-S -M**	**AER-S -PRED -M**	**AER-S -SUM -M**	**AER-S -REC -M**	**AER-S -MULT -M**
NASA	MSL	-	0.5714	0.4731	0.4634	0.4857	**0.5714**
	SMAP	-	0.7705	0.6329	0.6144	0.6418	**0.7705**
INET	SMD	-	0.1444	**0.1444**	0.1442	0.144	0.1208
Avg. F1 (μ ± σ)		-	0.4954 ± 0.3199	0.4168 ± 0.2491	0.4073 ± 0.2401	0.4238 ± 0.2546	**0.4876 ± 0.3329**
Avg. Sec (μ ± σ)		-	-	-	-	-	-
Total Precision		-	0.6474	0.5207	0.5286	0.5787	**0.7273**
Total Recall		-	0.1625	**0.1664**	0.1585	0.1506	0.148
Total F1		-	0.2598	**0.2523**	0.2439	0.239	0.2459
UCR Dataset		**ARIMA -M**	**AER-S -M**	**AER-S -PRED -M**	**AER-S -SUM -M**	**AER-S -REC -M**	**AER-S -MULT -M**
UCR	UCR	0.1481	**0.4168**	0.3378	0.3711	0.3333	**0.4168**
Total Precision		**0.4681**	0.341	0.2697	0.2816	0.2427	**0.341**
Total Recall		0.088	**0.536**	0.452	**0.544**	0.532	0.536
Avg. Sec (μ ± σ)		335 ± 539	-	-	-	-	-

Table B.8: Results of baseline ARIMA-M masked model compared to separate autoencoder regression masked models (AER-S-M).

Univariate Datasets		**ARIMA -M**	**AER-L -M**	**AER-L -PRED -M**	**AER-L -SUM -M**	**AER-L -REC -M**	**AER-L -MULT -M**
NASA	MSL	0.4565	**0.5455**	0.4842	0.4737	0.4706	**0.5455**
	SMAP	0.359	**0.75**	0.6714	0.7273	0.6929	**0.75**
YAHOO	A1	0.7519	0.75	0.7187	0.6921	0.6647	**0.75**
	A2	0.8093	**0.961**	0.9314	0.9381	0.961	**0.961**
	A3	0.8069	**0.9074**	**0.9074**	0.7438	0.6161	0.7665
	A4	0.6899	**0.7682**	**0.7682**	0.5681	0.4234	0.5891
NAB	Art	0.5	0.5	0.4286	0.3636	0.4444	**0.5**
	AWS	0.5176	**0.6897**	0.4889	**0.6897**	0.6667	0.6349
	AdEx	0.7692	0.7333	**0.7333**	0.7333	0.7143	0.7097
	Traffic	0.5	**0.5806**	0.5556	**0.5806**	0.5714	0.5455
	Tweets	0.5758	**0.5806**	0.5405	0.5588	0.566	**0.5806**
Avg. F1 (μ ± σ)		0.6124 ± 0.1584	**0.706 ± 0.1463**	0.6571 ± 0.1716	0.6426 ± 0.1551	0.6174 ± 0.1522	**0.6666 ± 0.1356**
Avg. Sec (μ ± σ)		44 ± 110	22 ± 8	22 ± 8	22 ± 8	22 ± 8	22 ± 8
Total Precision		0.8453	**0.9093**	0.8563	0.8655	0.8654	**0.8948**
Total Recall		0.6445	**0.6743**	**0.771**	0.5781	0.4598	0.5977
Total F1		0.7314	**0.7744**	**0.8114**	0.6932	0.6005	0.7167
Multivariate Datasets		**ARIMA -M**	**AER-L -M**	**AER-L -PRED -M**	**AER-L -SUM -M**	**AER-L -REC -M**	**AER-L -MULT -M**
NASA	MSL	-	0.5833	0.5116	0.5128	0.4507	**0.5833**
	SMAP	-	0.75	0.5862	0.5904	0.6815	**0.75**
INET	SMD	-	0.1495	0.1419	0.1423	**0.1495**	0.1232
Avg. F1 (μ ± σ)		-	0.4943 ± 0.31	0.4132 ± 0.2379	0.4152 ± 0.2395	0.4272 ± 0.2668	**0.4855 ± 0.3246**
Avg. Sec (μ ± σ)		-	76 ± 78	76 ± 78	76 ± 78	76 ± 78	76 ± 78
Total Precision		-	0.651	0.4811	0.5	0.596	**0.7197**
Total Recall		-	0.1651	**0.1678**	0.1625	0.1559	0.1493
Total F1		-	0.2634	**0.2488**	0.2453	0.2471	0.2473
UCR Dataset		**ARIMA -M**	**AER-L -M**	**AER-L -PRED -M**	**AER-L -SUM -M**	**AER-L -REC -M**	**AER-L -MULT -M**
UCR	UCR	0.1481	**0.4632**	0.4058	0.4183	0.3779	**0.4632**
Total Precision		**0.4681**	0.3805	0.3169	0.3199	0.2884	**0.3805**
Total Recall		0.088	**0.592**	0.564	**0.604**	0.548	0.592
Avg. Sec (μ ± σ)		335 ± 539	159 ± 218	159 ± 218	159 ± 218	159 ± 218	159 ± 218

Table B.9: Results of baseline ARIMA-M masked model compared to joint autoencoder regression masked models in the latent space (AER-L-M).

Univariate Datasets		**ARIMA -M**	**AER-R -M**	**AER-R -PRED -M**	**AER-R -SUM -M**	**AER-R -REC -M**	**AER-R -MULT -M**
NASA	MSL	0.4565	**0.6111**	0.5	0.5122	0.4865	**0.6111**
	SMAP	0.359	**0.7846**	0.6623	0.7075	0.7049	**0.7846**
YAHOO	A1	0.7519	**0.7405**	0.6761	0.7	0.7086	**0.7405**
	A2	0.8093	**0.9728**	0.9292	0.9381	**0.9728**	0.9586
	A3	0.8069	**0.8821**	**0.8821**	0.7218	0.6187	0.7505
	A4	0.6899	**0.7329**	**0.7329**	0.5363	0.42	0.5796
NAB	Art	0.5	**0.6667**	**0.6667**	0.5	0.4444	0.5
	AWS	0.5176	**0.7119**	0.4742	**0.7119**	0.6786	0.6562
	AdEx	0.7692	0.7333	0.6667	**0.7333**	0.6429	0.6207
	Traffic	0.5	**0.7179**	**0.7179**	0.5882	0.5714	0.5625
	Tweets	0.5758	0.5676	**0.5676**	0.5152	0.5	0.5246
Avg. F1 (μ ± σ)		0.6124 ± 0.1584	**0.7383 ± 0.1138**	**0.6796 ± 0.1397**	0.6513 ± 0.1346	0.6135 ± 0.1576	0.6626 ± 0.1353
Avg. Sec (μ ± σ)		44 ± 110	22 ± 8	22 ± 8	22 ± 8	22 ± 8	22 ± 8
Total Precision		0.8453	**0.9039**	0.8337	0.8533	0.8778	**0.8963**
Total Recall		0.6445	**0.7369**	**0.745**	0.5598	0.4619	0.5849
Total F1		0.7314	**0.8119**	**0.7869**	0.6761	0.6053	0.7079
Multivariate Datasets		**ARIMA -M**	**AER-R -M**	**AER-R -PRED -M**	**AER-R -SUM -M**	**AER-R -REC -M**	**AER-R -MULT -M**
NASA	MSL	-	0.6111	0.5432	0.5263	0.5067	**0.6111**
	SMAP	-	0.7833	0.5783	0.6289	0.6618	**0.7833**
INET	SMD	-	0.1489	0.1411	0.1379	**0.1489**	0.1283
Avg. F1 (μ ± σ)		-	0.5144 ± 0.3281	0.4209 ± 0.2429	0.431 ± 0.259	0.4391 ± 0.263	**0.5076 ± 0.3396**
Avg. Sec (μ ± σ)		-	76 ± 78	76 ± 78	76 ± 78	76 ± 78	76 ± 78
Total Precision		-	0.6684	0.5104	0.5259	0.5825	**0.7566**
Total Recall		-	0.1651	**0.1625**	0.1612	0.1585	0.1519
Total F1		-	0.2648	0.2465	0.2467	0.2492	**0.253**
UCR Dataset		**ARIMA -M**	**AER-R -M**	**AER-R -PRED -M**	**AER-R -SUM -M**	**AER-R -REC -M**	**AER-R -MULT -M**
UCR	UCR	0.1481	**0.4868**	0.4214	0.4276	0.3812	**0.4868**
Total Precision		**0.4681**	0.4134	0.3296	0.3297	0.2911	**0.4134**
Total Recall		0.088	**0.592**	0.584	**0.608**	0.552	0.592
Avg. Sec (μ ± σ)		335 ± 539	160 ± 214	160 ± 214	160 ± 214	160 ± 214	160 ± 214

Table B.10: Results of baseline ARIMA-M masked model compared autoencoder regression masked models in the reconstruction space (AER-R-M).

Univariate Datasets		**ARIMA**	**LSTM -NDT-1**	**LSTM -NDT-1 -B**	**LSTM -NDT-1 -E**	**LSTM -NDT-1 -T**
NASA	MSL	0.4421	0.4808	**0.5676**	0.5238	**0.5676**
	SMAP	0.3333	0.6892	**0.6906**	0.6565	0.6457
YAHOO	A1	0.7332	0.7325	0.6777	**0.7793**	0.7378
	A2	0.8074	**0.9704**	0.958	0.9349	0.9373
	A3	**0.8176**	0.7794	0.574	0.8153	0.7092
	A4	**0.6998**	0.6933	0.3368	0.6672	0.545
NAB	Art	0.3529	0.4	0.375	0.3529	**0.4211**
	AWS	0.5176	**0.5312**	0.4938	0.5	0.5067
	AdEx	0.7407	**0.7857**	0.6207	0.6207	0.6923
	Traffic	0.5	0.5946	**0.6154**	0.6047	0.5294
	Tweets	0.5672	0.6	0.5915	**0.6462**	0.4062
Avg. F1 ($\mu \pm \sigma$)		0.592 ± 0.1767	**0.6597 ± 0.1606**	0.591 ± 0.1659	0.6456 ± 0.1597	0.6089 ± 0.1555
Avg. Sec ($\mu \pm \sigma$)		44 ± 110	10 ± 4	12 ± 6	21 ± 10	19 ± 11
Total Precision		0.8332	0.8555	0.8078	**0.8675**	0.865
Total Recall		0.6552	0.6603	0.4295	**0.6633**	0.5428
Total F1		0.7336	0.7453	0.5609	**0.7517**	0.667
Multivariate Datasets		**ARIMA**	**LSTM -NDT-1**	**LSTM -NDT-1 -B**	**LSTM -NDT-1 -E**	**LSTM -NDT-1 -T**
NASA	MSL	-	0.5366	**0.5429**	0.5333	0.5
	SMAP	-	0.6023	**0.6364**	0.5942	0.5342
INET	SMD	-	**0.1547**	0.1228	0.123	0.1459
Avg. F1 ($\mu \pm \sigma$)		-	0.4312 ± 0.2417	**0.434 ± 0.2736**	0.4168 ± 0.2563	0.3934 ± 0.215
Avg. Sec ($\mu \pm \sigma$)		-	39 ± 42	61 ± 77	87 ± 104	78 ± 87
Total Precision		-	0.5299	**0.5515**	0.5245	0.5
Total Recall		-	**0.1757**	0.1413	0.1413	0.1546
Total F1		-	**0.2639**	0.225	0.2227	0.2361
UCR Dataset		**ARIMA**	**LSTM -NDT-1**	**LSTM -NDT-1 -B**	**LSTM -NDT-1 -E**	**LSTM -NDT-1 -T**
UCR	UCR	0.1236	0.4164	0.2723	0.2816	0.2184
Total Precision		0.2075	0.3631	0.219	0.2566	0.1927
Total Recall		0.088	0.488	0.36	0.312	0.252
Avg. Sec ($\mu \pm \sigma$)		335 ± 539	86 ± 106	161 ± 250	195 ± 250	160 ± 201

Table B.11: Results for LSTM-NDT with different attention mechanisms.

Univariate Datasets		**TadGAN -TF2**	**TadGAN -TF2-B**	**TadGAN -TF2-E**	**TadGAN -TF2-T**
NASA	MSL	0.5843	0.5747	0.5263	**0.6098**
	SMAP	0.6133	0.5465	0.6056	**0.6154**
YAHOO	A1	0.5325	0.3544	**0.5342**	0.4549
	A2	**0.8421**	0.5893	0.6536	0.54
	A3	0.3907	**0.3916**	0.2564	0.2645
	A4	0.2969	**0.3146**	0.3065	0.2775
NAB	Art	**0.5714**	0.3077	0.4615	0.4444
	AWS	**0.6774**	0.4615	0.5902	0.5085
	AdEx	0.72	0.1053	**0.7619**	0.5263
	Traffic	**0.5806**	0.3404	0.3871	0.4706
	Tweets	0.5882	0.4912	0.5846	**0.6197**
Avg. F1 ($\mu \pm \sigma$)		**0.5816 ± 0.1474**	0.407 ± 0.1441	0.5153 ± 0.1513	0.4847 ± 0.1227
Avg. Sec ($\mu \pm \sigma$)		145 ± 55	148 ± 56	112 ± 60	161 ± 68
Total Precision		**0.6469**	0.5875	0.6468	0.5875
Total Recall		**0.3487**	0.2959	0.2635	0.2486
Total F1		**0.4531**	0.3935	0.3745	0.3494
Multivariate Datasets		**TadGAN -TF2**	**TadGAN -TF2-B**	**TadGAN -TF2-E**	**TadGAN -TF2-T**
NASA	MSL	0.5238	**0.6265**	0.5	0.5833
	SMAP	0.5625	0.5542	**0.6395**	0.6197
INET	SMD	0.131	**0.1506**	0.1248	0.1168
Avg. F1 ($\mu \pm \sigma$)		0.4058 ± 0.2387	**0.4438 ± 0.2565**	0.4214 ± 0.2662	0.4399 ± 0.2804
Avg. Sec ($\mu \pm \sigma$)		829 ± 1032	727 ± 875	479 ± 573	728 ± 872
Total Precision		0.4936	0.5181	0.5022	**0.5596**
Total Recall		0.1532	**0.1704**	0.1532	0.1427
Total F1		0.2339	**0.2565**	0.2348	0.2274
UCR Dataset		**TadGAN -TF2**	**TadGAN -TF2-B**	**TadGAN -TF2-E**	**TadGAN -TF2-T**
UCR	UCR	**0.1625**	0.0888	0.0914	0.0988
Total Precision		**0.1071**	0.0594	0.0628	0.067
Total Recall		**0.336**	0.176	0.168	0.188
Avg. Sec ($\mu \pm \sigma$)		1898 ± 2876	1870 ± 2835	1258 ± 1905	1932 ± 2908

Table B.12: Results for TadGAN with different attention mechanisms.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *ArXiv*, 1409, 09 2014.

[4] Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, AAAIWS'94, page 359–370. AAAI Press, 1994.

[5] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey, 2019.

[6] Kukjin Choi, Jihun Yi, Changhwa Park, and Sungroh Yoon. Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. *IEEE Access*, 9:120043–120065, 2021.

[7] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Tadgan: Time series anomaly detection using generative adversarial networks, 2020.

[8] Ruei-Jie Hsieh, Jerry Chou, and Chih-Hsiang Ho. Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing. In *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*, pages 90–97, Nov 2019.

[9] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, Jul 2018.

[10] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two pure transformers can make one strong gan, and that can scale up, 2021.

[11] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. Time2vec: Learning a vector representation of time, 2019.

[12] J. Kusuma, L. Doherty, and K. Ramchandran. Distributed compression for sensor networks. In *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)*, volume 1, pages 82–85 vol.1, 2001.

[13] Dan Li, Dacheng Chen, Lei Shi, Baihong Jin, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks, 2019.

[14] Daehyung Park, Yuuna Hoshi, and Charles C. Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder, 2017.

[15] Eduardo H. M. Pena, Marcos V. O. de Assis, and Mario Lemes Proença. Anomaly detection using forecasting methods arima and hwds. In *2013 32nd International Conference of the Chilean Computer Science Society (SCCC)*, pages 63–66, 2013.

[16] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. Sensitivity of pca for traffic anomaly detection. *SIGMETRICS Perform. Eval. Rev.*, 35(1):109–120, jun 2007.

[17] Vyacheslav P. Tuzlukov and Cheng. *Signal Processing Noise*. CRC Press, Inc., USA, 2002.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[19] Renjie Wu and Eamonn Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2021.

[20] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.