Uncertainty-Aware Time Series Anomaly Detection

Paul Wiessner, Grigor Bezirganyan, Sana Sellami, Richard Chbeir, Hans-Joachim Bungartz

future internet

MDPI

Article

Uncertainty-Aware Time Series Anomaly Detection

Paul Wiessner [1], Grigor Bezirganyan [2], Sana Sellami [2], Richard Chbeir [3,*] and Hans-Joachim Bungartz [1]

[1] Department of Informatics, Technische Universität München, 85748 Garching, Germany; e-wiessner@in.tum.de (P.W.); bungartz@cit.tum.de (H.-J.B.)
[2] CNRS National Centre for Scientific Research, Aix Marseille University, 13397 Marseille, France; grigor.bezirganyan@univ-amu.fr (G.B.); sana.sellami@univ-amu.fr (S.S.)
[3] Department of Computer Science, University Pau & Pays Adour, E2S-UPPA, 64012 Anglet, France
* Correspondence: richard.chbeir@univ-pau.fr

Abstract: Traditional anomaly detection methods in time series data often struggle with inherent uncertainties like noise and missing values. Indeed, current approaches mostly focus on quantifying epistemic uncertainty and ignore data-dependent uncertainty. However, consideration of noise in data is important as it may have the potential to lead to more robust detection of anomalies and a better capability of distinguishing between real anomalies and anomalous patterns provoked by noise. In this paper, we propose LSTMAE-UQ (Long Short-Term Memory Autoencoder with Aleatoric and Epistemic Uncertainty Quantification), a novel approach that incorporates both aleatoric (data noise) and epistemic (model uncertainty) uncertainties for more robust anomaly detection. The model combines the strengths of LSTM networks for capturing complex time series relationships and autoencoders for unsupervised anomaly detection and quantifies uncertainties based on the Bayesian posterior approximation method Monte Carlo (MC) Dropout, enabling a deeper understanding of noise recognition. Our experimental results across different real-world datasets show that consideration of uncertainty effectively increases the robustness to noise and point outliers, making predictions more reliable for longer periodic sequential data.

Keywords: anomaly detection; time series; uncertainty quantification; deep neural networks; bayesian network

Citation: Wiessner, P.; Bezirganyan, G.; Sellami, S.; Chbeir, R.; Bungartz, H.-J. Uncertainty-Aware Time Series Anomaly Detection. *Future Internet* **2024**, *16*, 403. https://doi.org/10.3390/fi16110403

Academic Editors: Zhihao Liu, Franco Davoli, Davide Borsatti and Carlo Blundo

Received: 8 July 2024
Revised: 5 October 2024
Accepted: 28 October 2024
Published: 31 October 2024

1. Introduction

Anomaly detection in time series [1] is a challenging task spreading over various domains such as the Internet of Things (IoT), healthcare, logistics, etc. In connected environments, for example, anomaly detection faces additional significant challenges due to the inherent uncertainty surrounding data dynamics and system interactions. Indeed, time series data collected from various IoT devices often exhibit complex patterns influenced by interconnected factors, making it difficult to distinguish normal behavior from anomalies with certainty. The presence of uncertainty further complicates the anomaly detection in several ways [2], as it obscures the distinction between normal and anomalous behavior and thus affects the reliability of model predictions. Indeed, uncertainty [3] encompasses ambiguity or lack of information about the data, distinguishing it from anomalies, which represent deviations from expected patterns. We can decompose uncertainty into two different components: aleatoric and epistemic. Aleatoric uncertainty describes the inherent randomness in data, while epistemic uncertainty reflects the model's confidence in its predictions. For example, consider temperature and humidity sensors in a smart building. Aleatoric uncertainty may arise from natural fluctuations in temperature due to changes in occupancy or weather conditions, leading to potential misclassification of these fluctuations as anomalies. On the other hand, epistemic uncertainty could result from a lack of comprehensive training data, making the model less confident in its ability to accurately identify real anomalies, such as a sudden equipment malfunction. Considering both uncertainties is important for enhancing model robustness. On the one hand, addressing

aleatoric uncertainty helps the model handle noise, reducing the risk of false positives by distinguishing random fluctuations from true anomalies. On the other hand, acknowledging epistemic uncertainty informs decision-makers about the model's confidence levels, which is particularly vital in critical domains such as healthcare and IoT, where precise assessments of anomaly detection results are essential. By quantifying these uncertainties, we can guide practitioners in determining whether additional data or model adjustments are necessary to enhance the reliability of detections.

Addressing uncertainty in anomaly detection requires advanced approaches [3], such as Bayesian methods [4] or ensemble learning [5], which offer promising avenues for capturing and quantifying uncertainty in time series data. These techniques provide probabilistic assessments of anomalies, enabling decision-makers to gauge the reliability of detection results and prioritize response efforts accordingly. Traditional anomaly detection techniques [6] struggle to adapt to these uncertainties, leading to either missed anomalies or a high rate of false positives.

In this work, we explore a way to consider both aleatoric and epistemic uncertainties for anomaly detection in time series data. Based on existing works [7,8], our approach constructs an Autoencoder based on LSTM (Long Short-Term Memory) [9] for accurate time-series modeling [10]. Autoencoders are a well-established architecture for unsupervised anomaly detection. LSTMs foster an understanding of the complex relations in time series. The combination aims to learn normal behavior in data to be able to distinguish it from anomalous data. Further, we implement a solution for separate aleatoric and epistemic uncertainty quantification. Using a multi-layer neural network structure, we approach epistemic uncertainty quantification using the Bayesian posterior approximation method Monte Carlo (MC) Dropout [11]. For aleatoric uncertainty, a split head of the autoencoder architecture separately predicts per-point aleatoric uncertainty trained by a modified loss function. By combining these approaches into one model, we want to evaluate uncertainty quantification capabilities and its efficiency in univariate time series data. We argue that involving uncertainty supports the anomaly detection process to achieve more precise predictions. Consequently, we aim to have a closer look at uncertainty and how it behaves in the presence of anomalies. It is particularly interesting to investigate the behavior of aleatoric uncertainty compared to epistemic uncertainty in this context.

The rest of this paper is structured as follows. Section 2 describes several related studies. Section 3 describes our approach to anomaly detection under uncertainty. Section 4 presents experimental results. Finally, we draw a conclusion in Section 5.

2. Related Work

This section describes different works on both anomaly detection methods as well as anomaly detection under uncertainty.

2.1. Anomaly Detection Approaches

With the rise of deep neural networks, various methods have been developed to approach anomaly detection [12,13]. Unlike traditional machine learning approaches, deep neural networks excel in handling complexity and flexibility enabling them to scale to high dimensional data. Recurrent Neural Networks (RNN) are a type of neural network specifically designed for sequential data. Through a hidden state, they keep information over past states. Nevertheless, RNNs have a small memory and forget long-range dependencies. Moreover, they struggle with vanishing/exploding gradients. Long Short-Term Memory (LSTM) [9] was developed to overcome these problems making it valuable for anomaly detection in sequential data. LSTMs also found application in anomaly detection. The study in [14] uses an LSTM network to predict healthy electrocardiogram signals. A further variation of LSTMs has been developed by [15] where LSTM layers are stacked. It is supposed to be trained on normal data and uses multiple predictions in the future to find anomalies based on prediction error.

Autoencoders [16] try to compress the information into a lower dimensional (latent) space, to extract the most relevant features. In [17], the authors use this approach to detect anomalies from satellite telemetry and artificial data, while in [18], the authors proposed *Donut*, an unsupervised anomaly detection algorithm based on Variational Autoencoders (VAE) [19] for anomaly detection. VAEs try to learn a distribution over the latent space through variational inference. Since they model the latent space as a distribution instead of point estimates, one can generate new data points through sampling. Finally, there are approaches that combine several presented techniques. The work presented in [20] shows a combination of an LSTM with an encoder and decoder for multi-sensor time series data. OmniAnomaly [21] uses a combination of GRU and VAE with Planar Normalizing Flow. Combinations of autoencoder and time series forecasting methods based on latent space were also introduced and successfully applied. For example, [22] uses an autoencoder to reconstruct data and additionally uses a feed-forward network to predict one value ahead in terms of timesteps. Similar to that, [23] uses an autoencoder to predict one step ahead in time, and one step in the past relative to the given input window.

2.2. Anomaly Detection Approaches with Uncertainty Quantification

Uncertainty is often modeled through probabilistic approaches. However, probabilistic approaches alone fail to properly distinguish between *aleatoric* and *epistemic* uncertainties [24]. Several strategies evolved to handle uncertainty in anomaly detection. Some follow the idea of alleviating uncertainty in the input data before anomaly detection [25,26]. While this approach increases the robustness of the prediction model it fails to provide a measure for uncertainty. Other approaches have proposed the application of conformal methods, such as [27], which introduces a framework called cross-conformal anomaly detection. This framework aims to strike a balance between statistical accuracy and computational efficiency for uncertainty-quantified anomaly detection. Bayesian deep learning provides tools for uncertainty quantification of deep learning models [8]. It is a subset of probabilistic machine learning approaches, where prior distribution is placed on the model parameters and one needs to estimate the posterior distributions based on the observed data. Since exact Bayesian methods for deep learning are still intractable, various methods have been proposed to approximate the posterior distribution, namely Monte Carlo Dropout [11], Bayes by Backpropagation [28] or Ensembles [5]. The Bayesian approach [29] was also applied to autoencoders. Reference [30] shows in an empirical study how the integration of uncertainty in the score function is likely to improve anomaly detection for general datasets. In [31], the authors take this one step further by changing the autoencoder to a VAE considering uncertainty as an anomaly score. The authors of [32] show the applicability of those methods on time series data comparing multi-layer perceptron networks to LSTM networks. In [33], the authors use an LSTM and turn it into a Bayesian network to predict anomalies in satellite telemetry data by comparing various kinds of posterior approximations. The study in [7] employs an LSMT autoencoder with point-wise prediction including model uncertainty calculation.

Most approaches focus on quantifying epistemic uncertainty, neglecting the inherent data uncertainty. In [8], the authors proposed to use an additional network output to quantify aleatoric uncertainty in computer vision. A method for calculating the predictive uncertainty, which can be separated into model and data uncertainties, is provided using negative log likelihood (NLL) [5,34]. It relies on decoupling the NLL based on MC sampling to achieve an anomaly probability as well as a split measure for model and data uncertainty. Similarly, decomposition can be achieved by decoupling prediction uncertainty [22] derived from the variance of the prediction distribution.

2.3. Take Away

Uncertainty quantification is crucial for making informed decisions based on time series data. However, most existing anomaly detection approaches primarily focus on epistemic uncertainty, which reflects model limitations, neglecting the inherent randomness

or noise in the data themselves (aleatoric uncertainty). This work proposes a novel approach that uses separate techniques to quantify both aleatoric and epistemic uncertainty in time series data. Unlike existing works that solely aim for highly accurate anomaly detection, our approach tackles noise in data by quantifying aleatoric uncertainty. This allows for more robust anomaly detection and addresses model uncertainty to enhance the overall reliability of predictions.

3. Proposed Approach

In this section, we describe the approach we developed, called LSTMAE-UQ (code is available at https://github.com/p199671/LSTM-AE-UQ.git (accessed on 1 September 2024)) (Long Short-Term Memory Autoencoder with Aleatoric and Epistemic Uncertainty Quantification), to address the problem of detecting uncertain anomalies in univariate time series data. LSTMAE-UQ integrates LSTM cells in the autoencoder architecture and quantifies uncertainties in both data and model. We chose this combination for two main reasons: Firstly, by using LSTM, we can capture the temporal structure of time series data. Secondly, an autoencoder is an unsupervised learning approach that does not require labels for learning. Additionally, it allows for univariate and multivariate time series anomaly detection and sets the basis for epistemic uncertainty quantification by its multi-layer neural network structure. The combination of LSTM and the autoencoder can learn to capture relevant features of time series data and aims to reconstruct them properly. Moreover, unlike previous works which focus mainly on epistemic uncertainty, our approach considers both uncertainties separately. To this aim, we adapt the model inspired by [8] to output an estimate of data uncertainty as well as epistemic uncertainty using MC Dropout [11].

The architecture of LSTMAE-UQ is illustrated (to ease readability, we do not show the averaging over the MC samples for reconstructions and uncertainty for the reconstruction error) in Figure 1. The task of the described architecture is to learn representative features of the input data given the window size and reconstruct it properly. The input data consists of one observation for each timestep. We reconstruct these observations which should be as close to the mean-normalized input as possible (see example in Table 1). Through a separate head, apart from the reconstruction output, the model also returns the variance, which corresponds to the respective data uncertainty.

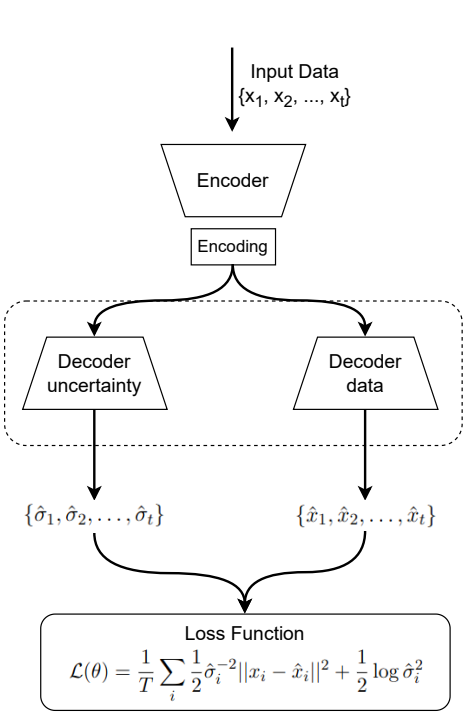
Table 1. Example of desired reconstruction.

Original Input Window	Normalized Input Window	Desired Reconstruction
$(1 \quad 4 \quad 4)$	$(-2 \quad 1 \quad 1)$	$(-2 \quad 1 \quad 1)$

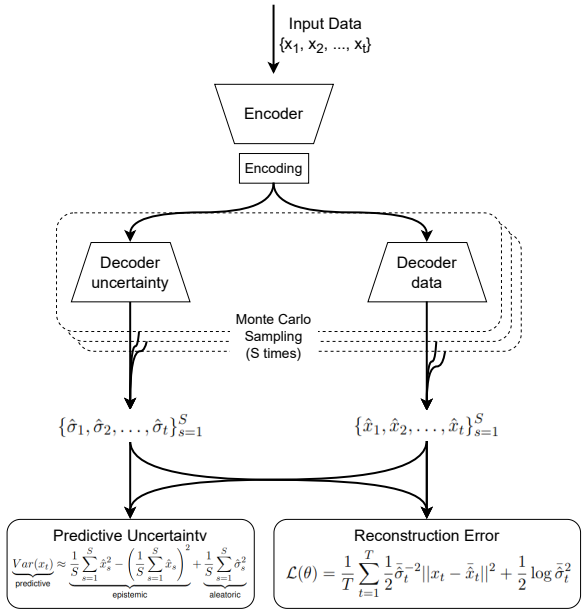
An autoencoder consists of two parts: an encoder and a decoder. In our work, each of the encoders and decoders is constructed of a two-layer LSTM network (Table 2). The first layer of the encoder takes the time series as input. The second layer, smaller in size than the first, encodes the output of the input layer to a lower dimensional representation. The dimensions of the encoding is 0.8 times the input window size. The first layer of the decoder uses the encoding as input. The output is passed to the second decoder layer which aims to reconstruct the input dimensions. Additionally, we split the head of the network and added a small fully connected network. This second output layer will be trained to calculate an estimate of aleatoric uncertainty.

Table 2. Network Layers.

Encoder	LSTM:in → Drop → LSTM:encoding
Decoder	LSTM:encoding → Drop → LSTM:reconstruction → Drop → FC:out LSTM:encoding → Drop → LSTM:uncertainty → Drop → FC:out



(**a**) Model architecture during training.



(**b**) Model architecture during prediction. For better readability, the averaging over the MC samples for reconstructions and uncertainty for the reconstruction error is not shown.

Figure 1. Model Architecture.

3.1. Loss Function

The model output consists of two values through two decoders: (1) the reconstruction of the input data, and (2) the estimate of aleatoric uncertainty. We use these two values in a modified version of the mean squared error (MSE) loss function, introduced by [8]. Reconstruction is used as usual in MSE loss. Special is the usage of the aleatoric uncertainty estimate which attributes to the MSE a loss resulting in direct influence on algorithm training. For an input vector X, we define the loss in Equation (1).

$$\mathcal{L}(\theta) = \frac{1}{T}\sum_i \frac{1}{2}\hat{\sigma}_i^{-2}||x_i - \hat{x}_i||^2 + \frac{1}{2}\log\hat{\sigma}_i^2, \tag{1}$$

where T is the number of Monte Carlo samples, $\hat{x}_i$ is the reconstruction of the input value x_i, and σ_i^2 is the variance representing aleatoric uncertainty (the second output of the model).

The minimization objective consists of two parts: (1) the first regression part is obtained through the stochastic sampling of model parameters (MC Dropout) which represents the difference of reconstructed output and input (weighted by aleatoric uncertainty), and (2) the second part is used as an uncertainty regularization term.

Labels to learn the uncertainty are not necessary. Through the main task of learning the regression, the variance $\hat{\sigma}_i^2$ is implicitly learned from the loss function.

In practice, the network is trained to predict the log variance $v_i := \log\hat{\sigma}_i^2$. The minimization objective changes accordingly as exemplified in Equation (2):

$$\mathcal{L}(\theta) = \frac{1}{T}\sum_i \frac{1}{2}\exp(-v_i)||x_i - \hat{x}_i||^2 + \frac{1}{2}v_i. \tag{2}$$

By predicting the log-variance instead of variance, the training process is more numerically stable, as we avoid potential division by zero.

Additional prediction of uncertainty helps to improve the robustness of the network. Please note that $\frac{1}{2}\hat{\sigma}^{-2}$ (the red curve in Figure 2) has a tempering effect on the residual

discouraging to predict very high or very low uncertainties. On the contrary, for high learned uncertainty, the residual gains lower importance. This serves as an automatic mechanism to involve data uncertainty in the model training and makes it less prone to noise. The model is also discouraged from either predicting only high or low uncertainty for all data points. Figure 2 helps explain this interplay. Predicting high uncertainty will cause the term $\frac{1}{2}\log\hat{\sigma}^2$ to be large which, in turn, penalizes the model. On the other hand, predicting low uncertainty leads to a higher value of the residual term $\hat{\sigma}^{-2}$. And again, it would penalize the model by exaggerating the contribution of very low uncertainties.

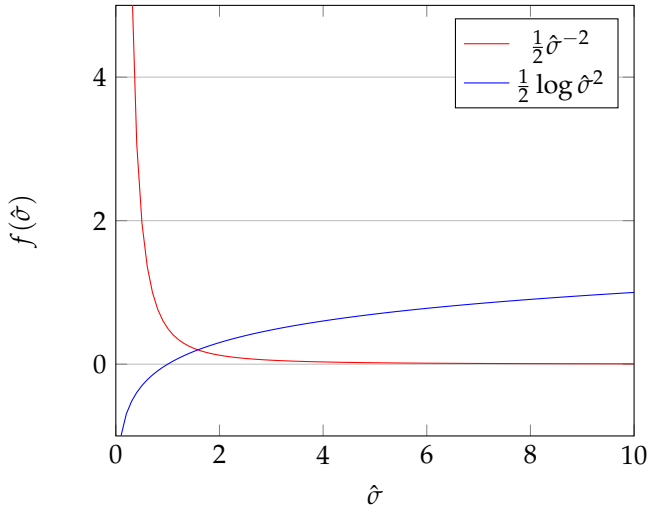


Figure 2. Visualiation of uncertainty terms in the loss function.

3.2. Reconstruction Error

The goal of an autoencoder is to reconstruct the input data sequence as well as possible. After training on normal data, it is able to reconstruct normal data patterns properly. In contrast, it will fail to properly reconstruct abnormal data patterns. In this work, we use the error between input data and reconstructed data as anomaly score. For a given observation, we define this in Definition 1.

Definition 1 (Anomaly Scores)**.** *We define any observation during prediction time by a 2-tuple:*

$$s : (o, r)$$

where

- *o is the observation,*
- *r is the reconstruction error.*

Each observation of the time series data is assigned an anomaly score, resulting in a point-wise scoring of the data points. We summarize this in Definition 2.

Definition 2 (Time Series Scoring)**.** *The result of a time series anomaly detection algorithm is a time series scoring*

$$S = \{s_1, s_2, ..., s_n\}$$

with $s_i \in \mathbb{R}$ being assigned to its corresponding input $d_i \in D$ and D corresponds to data.

A widely used metric for the reconstruction error is the MSE [10]. For this algorithm, we again use the loss function described in Equation (2) As described, it is based on MSE which is popular for autoencoders. MSE is a mathematically simple metric that provides clear and easy interpretability. In comparison to L_1 loss, it takes the square over the difference between input and reconstruction. This makes it more sensitive to larger

errors and such supporting the task of anomaly detection. In addition, the extension by integration of aleatoric uncertainty allows the direct influence of data uncertainty on the anomaly probability.

3.3. Uncertainty Calculation

In this work, we quantify aleatoric and epistemic uncertainty separately. Combining them results in predictive uncertainty. Definition 3 defines this formally.

Definition 3 (Predictive uncertainty)**.** *The uncertainty resulting from inherent data uncertainty and model uncertainty is defined as predictive uncertainty:*

$$\mathcal{U}^{predictive} = \mathcal{U}^{aleatoric} + \mathcal{U}^{epistemic}$$

with

- $\mathcal{U}^{aleatoric}$ *as uncertainty in data and*
- $\mathcal{U}^{epistemic}$ *as model uncertainty*

Aleatoric uncertainty is learned by the network and serves as additional network output. To additionally quantify epistemic uncertainty we use MC Dropout. Sampling multiple times from the posterior distribution results in a set $\{\hat{x}_t, \hat{\sigma}_t^2\}_{t=1}^T$ of T sampled outputs.

From that, we can derive epistemic uncertainty by calculating the variance over predictions:

$$\hat{x} = \frac{1}{T}\sum_{t=1}^{T} \hat{x}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T} \hat{x}_t\right)^2. \tag{3}$$

Calculating the mean of uncertainties over all MC samples gives the final aleatoric uncertainty:

$$\hat{\sigma}^2 = \frac{1}{T}\sum_{t=1}^{T} \hat{\sigma}_t^2. \tag{4}$$

Combined, we can summarize, with Equations (3) and (4) the predictive uncertainty for a given input vector x as follows:

$$\underbrace{Var(x)}_{\text{predictive}} \approx \underbrace{\hat{x}}_{\text{epistemic}} + \underbrace{\hat{\sigma}^2}_{\text{aleatoric}}. \tag{5}$$

Algorithm 1 summarizes the steps from input to anomaly and uncertainty metric calculation in pseudocode.

3.4. Thresholding Mechanism

The final task for the algorithm to achieve is to translate the anomaly score, and if so also additional scores, into binary anomaly labels. Therefore, we can define an anomaly as in Definition 4.

Definition 4 (Anomaly)**.** *An anomaly is defined as a prediction that fulfills the following conditions:*

$$s > s*$$

where

- *s is the anomaly score,*
- *s* is the anomaly score threshold.*

Algorithm 1 Algorithm to produce anomaly score.

Input: input sequence $x = \{x_1, x_2, \ldots, x_T\}$, MC dropout sample size S, encoder $\mathcal{E}(\cdot)$, decoder data $\mathcal{D}_d(\cdot)$, decoder uncertainty $\mathcal{D}_u(\cdot)$
Output: A, U_p
$\hat{x} \leftarrow \{\}$
$\hat{\sigma} \leftarrow \{\}$

MC Dropout sampling
for $s \leftarrow 1$ to S **do**
$z \leftarrow \mathcal{E}(x)$ ▷ encode input sequence
$\hat{x} \leftarrow \hat{x} \cup \mathcal{D}_d(z)$ ▷ reconstruct encoding z
$\hat{\sigma} \leftarrow \hat{\sigma} \cup \mathcal{D}_u(z)$ ▷ predict aleatoric uncertainty
end for

Bayesian Approximation over MC Dropout Samples
$\bar{x} \leftarrow \frac{1}{S}\sum_{s=1}^{S} \hat{x}_s^2$
$\bar{\sigma} \leftarrow \frac{1}{S}\sum_{s=1}^{S} \hat{\sigma}_s^2$

Anomaly and Uncertainty Metrics
$A \leftarrow \frac{1}{T}\sum_{s=1}^{S} \frac{1}{2}\hat{\sigma}_s^{-2}||x_s - \hat{x}_s||^2 + \frac{1}{2}\log\hat{\sigma}_i^2$ ▷ anomaly score
$U_a \leftarrow \frac{1}{S}\sum_{s=1}^{S} \hat{\sigma}_s^2$ ▷ aleatoric uncertainty
$U_e \leftarrow \frac{1}{S}\sum_{s=1}^{S} \hat{x}_s^2 - (\frac{1}{S}\sum_{s=1}^{S} \hat{x}_s)^2$ ▷ epistemic uncertainty
$U_p \leftarrow U_e + U_a$ ▷ predictive uncertainty

return A, U_p

Thresholding is an orthogonal task to anomaly score calculation and an algorithm-independent problem.

To choose an appropriate mechanism, we first set up two assumptions about anomalies in the data:

- **Assumption 1: Rare Events.** We begin by assuming anomalies are rare events. This means the proportion of anomalous labeled data points is vanishingly small compared to the total number of data points.
- **Assumption 2: Random Occurrence.** Despite their rarity, we cannot predict when or where an anomaly will appear within a time series. Anomalies are assumed to occur at random times.

Since the threshold for anomaly detection is set based on the training data, Assumption 2 becomes crucial. Because anomalies are assumed to occur randomly (Assumption 1), the training data themselves might contain some anomalies, even though they are rare. This can affect the model's ability to detect future anomalies effectively.

Consequently, we decided to adopt a thresholding mechanism that sets the threshold at the $0.999th$ quartile based on the anomaly score of the training data. This means in practice that 99.9% of the training data are seen as normal with 0.1% considered anomalous. This is a hyperparameter that can be tuned, based on prior knowledge of the dataset. This of course holds the risk of setting a threshold slightly lower than necessary and labeling some data points as false positives. On the other side, when longer anomalies occur in the training set, they can be labeled as false negatives. However, this approach tries to keep both sides low while trying not to overcomplicate this task.

4. Experiments

This section describes the experiments that we have conducted in order to evaluate our approach. We recall the following research questions:

1. How do uncertainty quantification measures react when different levels of uncertainty are introduced into the data?
2. How do the uncertainties react to different types of anomalies?
3. What is the impact of considering uncertainty on the performance of anomaly detection models?

4.1. Uncertainty Detection Capabilities

We built two uncertainty estimation techniques into the model—one to detect aleatoric uncertainty—and the other one to detect epistemic uncertainty. In this experiment, we want to test how they react to different levels of noise in the data.

4.1.1. Data

We prepare five different synthetic pairs of datasets, using the synthetic data generation tool gutenTAG [35], each consisting of a train and a test set. All of them follow the same basic data pattern of a sine function. Therefore, we introduce noise to the data by adding a controlled deviation to the existent base data [36]. This may correspond to more imprecise measurements often present in real-world data.

Five different levels of noise are introduced, one for each pair of training and testing data. The different levels of noise are as follows: 0%, 1%, 10%, 30%, 50%. The two datasets of each pair are generated separately and are not identical to avoid testing on a perfectly learned training dataset (Table 3). We use a standard amplitude of 1 and 100 equally distributed points. A visualization is depicted in Figure 3. The data are available at https://github.com/p199671/LSTM-AE-UQ/tree/master/data (accessed on 1 September 2024).

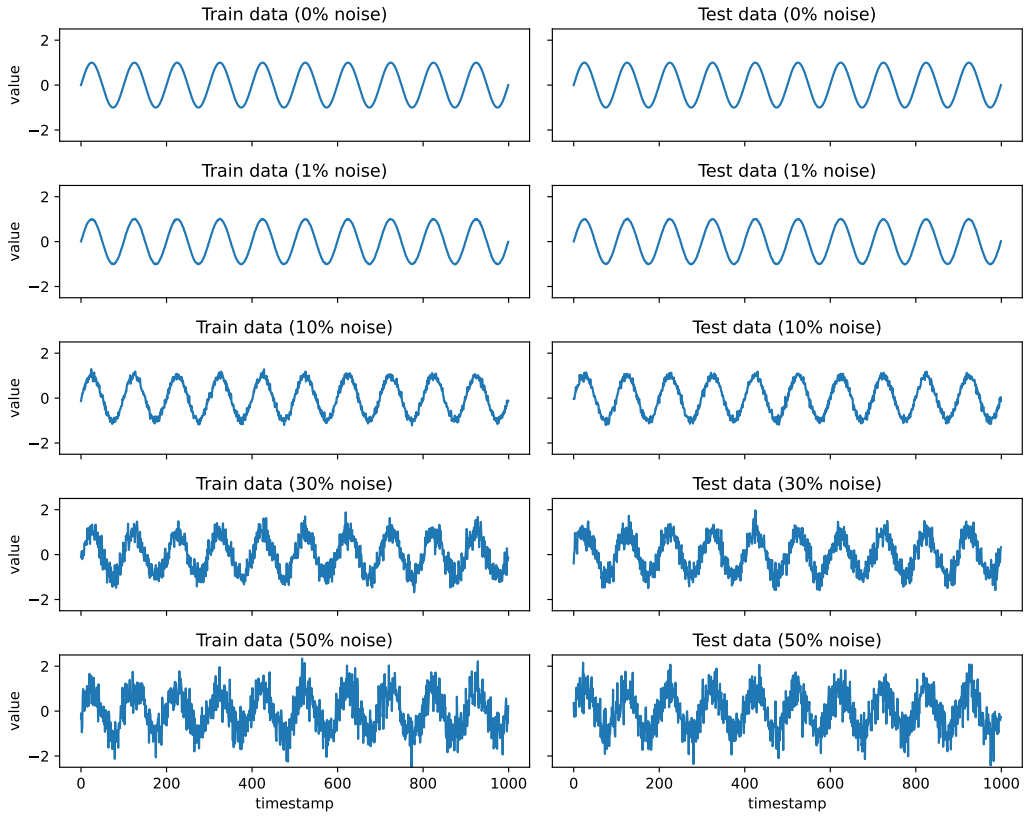


Figure 3. Sine data for training and testing with different levels of noise.

Table 3. Base sine specifications with noise levels.

Attribute	Value
Function	$\sin(x)$
Amplitude	1.0
Period	100 points
Length	1000 points
Noise	0%, 1%, 10%, 30%, 50%

The declaration in percent relates to the whole value range of the base function, meaning 1% refers to an absolute value of 0.01 for a given value range of 1. These noise levels can be seen as real-world emulation where noise increasingly obscures true observations.

4.1.2. Model Training

We train the model (Table 4) for 250 epochs with a test on validation data every 10th epoch. This is a performance improvement regarding the computation it takes for MC sampling. Additionally, we implement an early stopping mechanism to avoid overfitting. This terminates the training procedure after the third increased validation loss in a row. In the experiments, we refer to the small model as an architecture of 50-40-50 and to the big model as 100-80-100 which corresponds to a compression rate of 0.2. Further, the model is trained on a separate, clean training set, if not specified differently. We separate the training set into training and validation sets with a ratio of 80 : 20. The learning rate is set to a standard value of 0.001. For Bayesian posterior approximation, we set dropout between layers to 0.3 throughout all test cases and MC sampling size to 100. We use a batch size of 64 samples per batch to enhance performance. Additionally, the Adam optimizer is employed, and we utilize a modified loss function, detailed in Section 3.1. The parameters are based on the work of [7], whose similar model architecture makes these parameters a suitable choice, also enhancing comparability.

Table 4. Parameters for model training.

Attribute	Value
epochs	250
validation	every 10th epoch
early stopping	after 3rd validation
train/val ratio	80 : 20
learning rate	0.001
dropout rate	0.3
batch size	64
dropout sample size	100
optimizer	Adam
loss	modified mse

4.1.3. Results

Figures 4 and 5 show the results of the experiment for aleatoric and epistemic uncertainty accordingly. In this experiment, we tested two different model sizes and two window sizes. Each model is trained on data with a certain level of noise according to the legend, and evaluated on varying noise levels, indicated on the X-axis. The Y-axis shows the value of aleatoric/epistemic uncertainty taken as the median over the whole test results.

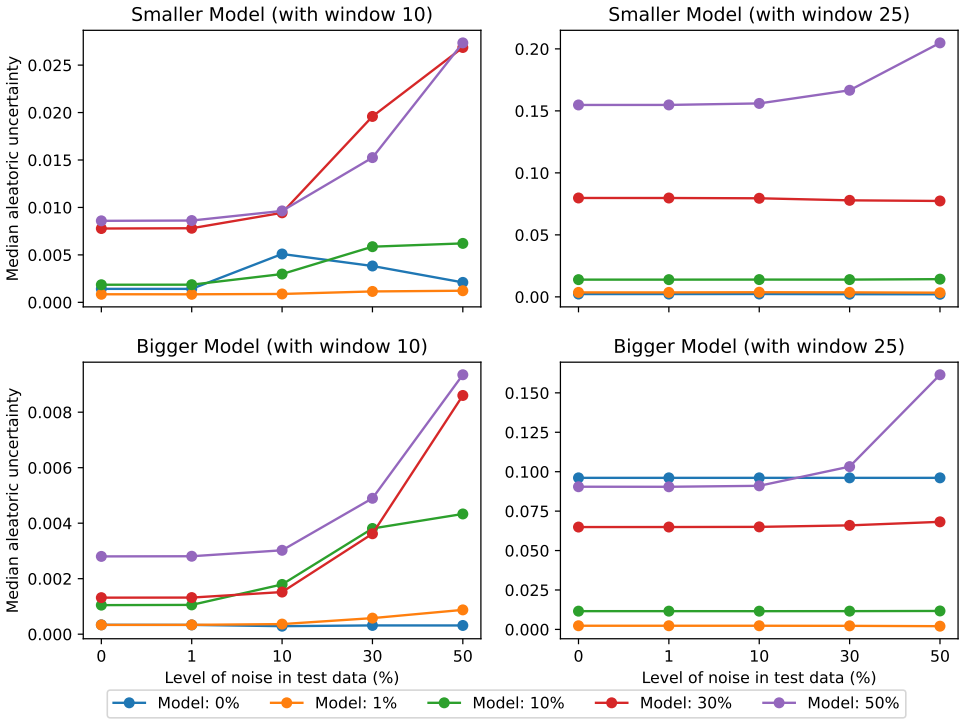


Figure 4. Median aleatoric uncertainty for models trained on a certain level of noise and evaluated on several other levels of noise in data.

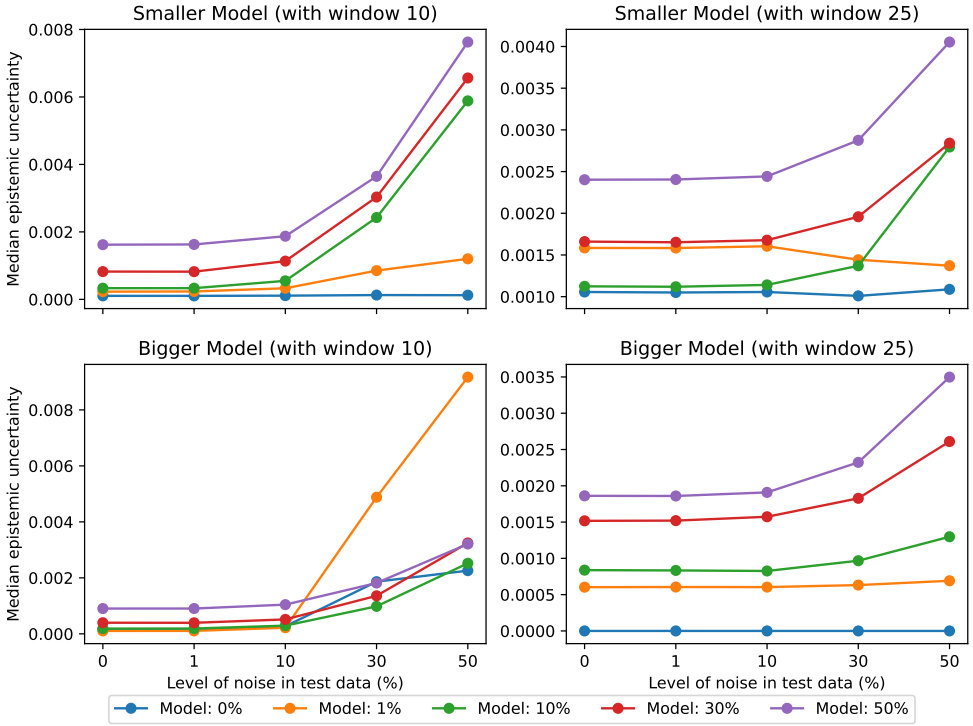


Figure 5. Median epistemic uncertainty for models trained on a certain level of noise and evaluated on several other levels of noise in data.

Firstly, our results show a general trend that a higher level of noise in the data leads to a higher noise estimate. This relationship is indicative of the model's capacity to effectively learn and respond to varying noise levels. However, it is noteworthy that a model trained

on data with low to no noise demonstrates limited sensitivity to different levels of noise. In such cases, the model's noise prediction appears less reliable. This observation underlines the importance of training the model on data close to the one on which it is going to be tested. A model that learned diverse levels of noise in training data will be able to better differentiate these levels during the test and enhance its adaptability.

The influence of the input window on noise detection is also sufficiently visible. A model's ability to differentiate between noise levels in the test data is notably impacted by the noise levels it was exposed to during training. Models trained on higher noise levels tend to produce higher noise estimates, but they may lose the ability to distinguish between levels of noise in the test data, possibly due to overfitting. Further, the model size does not seem to have a significant effect on noise detection capabilities. This suggests that model performance is not necessarily tied to the complexity or size of the architecture. Moreover, the absence of a distinct absolute threshold for noise categorization implies that the model relies more on relative changes in noise levels. This relative perspective allows the model to adapt to the specific noise distribution it has been exposed to during training, underlining the importance of training data diversity.

The effect of noise in data on epistemic uncertainty shows parallel behavior to that observed for aleatoric uncertainty, providing several interesting insights (Figure 5).

In general, it becomes clear that epistemic uncertainty increases when the model is subject to higher levels of noise in the test data. This finding seems to support the thesis that the presence of noisy data has an influence on the model's epistemic uncertainty. Models trained on datasets with lower noise levels once again struggle with data with higher noise. This phenomenon underscores the importance of incorporating diverse noise levels in the training dataset, as models trained only on clean data may struggle to develop a good understanding of uncertainty. Furthermore, a notable observation is that the absolute values of epistemic uncertainty appear to be more consistent, especially when considering the same size of the input window. This suggests that, compared to aleatoric uncertainty, epistemic uncertainty may rely less on the specific model architecture and size and more on the noise within the data. Interestingly, models trained on higher noise levels exhibit an improved understanding of noise, expressed in the uncertainty of the model. This indicates that models can adapt to and learn from varying noise levels during the training process, enhancing their ability to interpret and quantify epistemic uncertainty in different noise conditions.

4.1.4. Discussion

This experiment gave a couple of insights into the model's proficiency in recognizing noise in the data and its effect on aleatoric and epistemic uncertainty.

We saw that noise in the training data plays an important role. It enables the model, to better recognize and assess noise it sees in test data. This is clearly visible in both aleatoric and epistemic uncertainty. Further, it supports the common thesis that noise in data has an influence on epistemic uncertainty. Higher noise leads to an increase in the model's ability to precisely reconstruct the input.

A takeaway from this experiment is the importance of training data. As with the intended goal of prediction, the training data have to be balanced to guarantee proper learning. This accounts also for noise. A model, that could learn noise from the training data, will not be able to recognize noise in test data. On the contrary, a model trained on high variance in data is more likely to recognize and categorize noise properly.

4.2. Uncertainty Quantification Evaluation

To investigate uncertainty behavior during anomalies, we generate different synthetic datasets based on the sine function using the Time Series Generator (GutenTAG) [35], which is an extensible tool to generate time series data with and without anomalies.

4.2.1. Data

We create five different pairs of time series, one for each anomaly type we aim (Figure 6). The datasets contain data points according to the mathematical function $sin(x)$. Each includes 100 points per period with an overall of 10 periods. The training set consists of clean data with little background noise. To each test dataset, we inject one anomaly which refers to a larger/smaller amplitude, an extremum, and a pattern anomaly as further specified in Table 5.

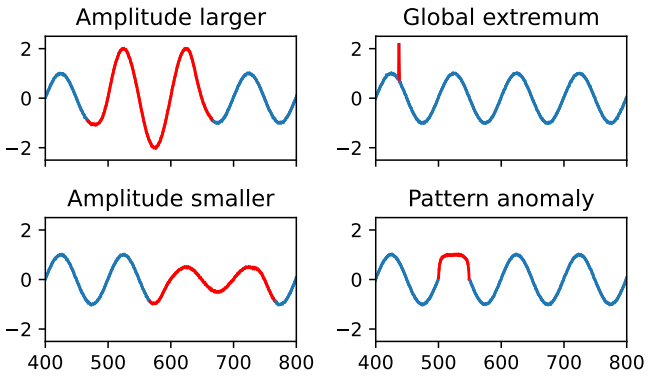


Figure 6. Anomalies in test data.

Table 5. Specifications of anomalies in the test dataset.

Anomaly	Length	Description
Amplitude larger	200	Larger amplitude with factor 2.
Amplitude smaller	200	Smaller amplitude with factor 0.5.
Global extremum	1	Extremum outside the scope of normal values.
Pattern	50	Deviation from pattern with larger radius at one turning point of the sine curve.

4.2.2. Results

Starting with the anomaly of a larger amplitude than normal, observations presented in Figure 7 show a general trend where larger amplitudes correspond to heightened uncertainties. The established patterns mostly continue, with increased intensity and/or elevated baseline levels during anomalies. Notably, the ability of the model to recognize anomalies as uncertainty appears to be dependent on the input size. Some window sizes show stronger reactions, especially smaller windows, while others seem to be unaffected by the introduced anomalies. For instance, when considering an input size of 100 and its associated aleatoric uncertainty, the impact of anomalies seems completely absent.

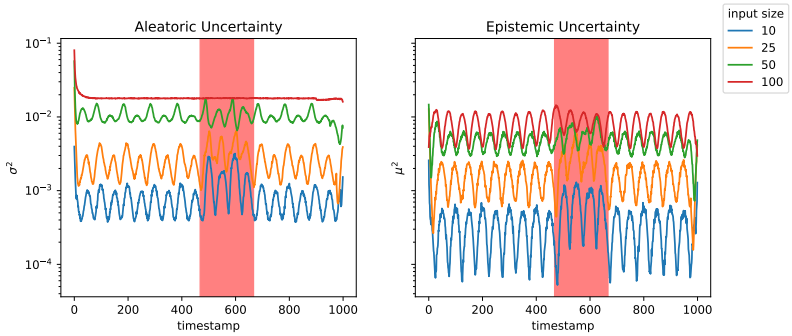


Figure 7. Results for experiments with anomalies of larger amplitude. The highlighted area in red is the range where the anomaly takes place (see Figure 6).

Having investigated the reaction of a larger amplitude, we now consider the opposite situation with a smaller amplitude. Results are illustrated in Figure 8. In contrast to the impact of larger amplitudes, a smaller amplitude is associated with reduced uncertainties. However, the general pattern in the data continues with less intense amplitudes. Also, certain input window sizes do not seem to influence the impact of smaller amplitude anomalies. Notably, input sizes such as 100 for aleatoric uncertainty, seem to miss the ability to capture the influence of reduced amplitudes.

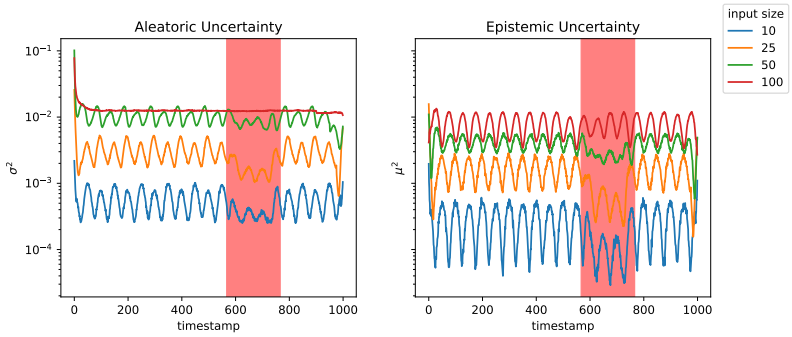


Figure 8. Results for experiments with anomaly of smaller amplitude. The highlighted area in red is the range where the anomaly takes place (see Figure 6).

From amplitudes, we continue with the extremum anomaly with results in Figure 9. The anomaly is visible in the smallest input window size for both aleatoric and epistemic uncertainties. A distinct peak emerges in the uncertainty profiles, providing an indication of the anomaly's influence. In the second smallest input window size, deviations in the uncertainty patterns become more subtle. These deviations show up as a short noisy range or a dent in the uncertainty profiles. The distinct peaks observed in the smallest window size no longer appear. Interestingly, the extremum anomaly appears to have a minimal impact on uncertainties associated with larger input window sizes. For the majority of cases, uncertainties remain unimpacted.

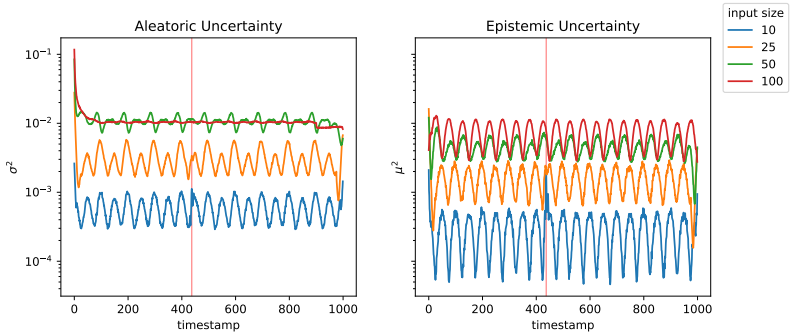


Figure 9. Results for experiments with anomaly of global extremum. The highlighted area in red is the range where the anomaly takes place (see Figure 6).

The results of introducing a pattern anomaly marked by a more rounded turning point in the sine curve, are introduced in Figure 10. The pattern anomaly shows a more intensive impact on uncertainties associated with smaller input window sizes (i.e., 10, 25). The deviations in uncertainty profiles are more prominent in configurations with reduced temporal context, indicating heightened sensitivity to anomalies in these scenarios. In contrast, larger input window sizes exhibit either subtle changes or no discernible impact on uncertainties. The observed deviations are notably stronger for epistemic uncertainty. This implies a more nuanced response of the model to pattern anomalies, with epistemic uncertainty capturing and reflecting the alterations in the learned patterns more intensively. A distinctive observation emerges concerning the temporal sensitivity around the turning point. The model shows a more intense reaction in uncertainties at locations close to the

turning point, while during the turning point itself, the uncertainty profiles tend to revert to roughly the normal pattern. This aligns with expected behavior, given the model's capacity to learn and adapt from the sine curve's inherent turning points.

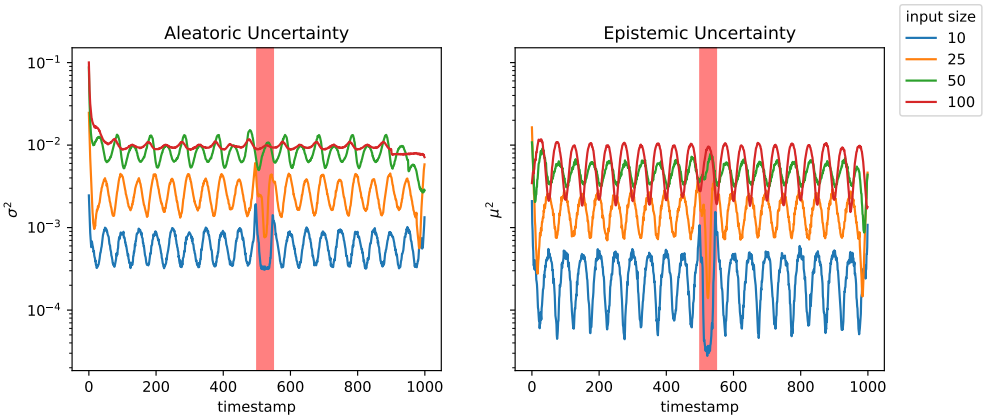


Figure 10. Results for experiments with pattern anomaly. The highlighted area in red is the range where the anomaly takes place (see Figure 6).

4.2.3. Discussion

In this experiment, we saw that the model is responsive to various anomalies in its aleatoric and epistemic uncertainties. All uncertainty types exhibit patterns in reaction to anomalies, providing insights into the model's capacity to capture and quantify uncertainties under multiple conditions.

A clear trend emerges, indicating that the duration of an anomaly temporally influences its effect on uncertainties. Longer-lasting anomalies, as deviations in amplitude, tend to show a more visible impact on smaller as well as larger input window sizes, highlighting the temporal sensitivity of the model. In contrast, short anomalies, such as point anomalies, are predominantly captured by shorter input windows.

Epistemic uncertainty often demonstrates a more confident behavior during anomalous regions. Despite the differences, both aleatoric and epistemic uncertainties frequently exhibit similar patterns, providing confirmation that epistemic uncertainty is influenced by aleatoric uncertainty.

A recurring observation is the occurrence of strong increases or decreases, or abnormal patterns at the beginnings and endings of curves. This behavior may be attributed to the influence of input windows, where limited information at the start and termination contributes to uncertainty until sufficient context is available. To avoid this and achieve more reliable curves, filters can be adapted. An observation that occurred not to any pattern is that some input window sizes are unaffected by anomalies. While others react heavily during anomalies, they continue normal patterns. An explanation could deliver the overfitting problem. As validation is conducted only every 10th epoch and early stopping after a third increase in loss, this leaves the possibility that overfitting is not properly detected.

4.3. Model Performance

We evaluate the model performance on a variety of real-world univariate time series datasets based on the TSB-UAD (https://github.com/TheDatumOrg/TSB-UAD) (accessed on 1 September 2024). This benchmark holds a repository of 12686 time series with labeled anomalies containing real-world, synthetic and artificial data. From those, we pick all 18 datasets from real-world applications, containing 1980 time series. They were recorded from diverse domains, such as transportation, healthcare, web services or environmental monitoring. The datasets are variable in size and differ in anomaly types, densities and length. We specifically picked those as they are partly recorded by sensors in real environments and thus prone to incorrect measurements which corresponds to our use case.

Table 6 provides a deeper insight into the details of each dataset. For each count of time series per dataset, the average length over time series is listed. Furthermore, it contains information about the average number of contiguous anomalies, the average number of abnormal points and the difficulty of the dataset according to relative contrast.

Table 6. Summary characteristics of the 18 benchmark datasets in TSB-UAD. The R_c value measures the separability of a point to the other points.

Dataset	Count	Average Length	Average # Anomalies	Average # of Abnormal Points	R_c
Daphnet [37]	45	21,760.0	7.6	2841.0	2.38
Dodgers [38]	1	50,400.0	133.0	5612.0	2.02
ECG [39]	52	230,351.9	195.6	15,634.0	8.33
Genesis [40]	6	16,220.0	3.0	50.0	2.28
GHL [41]	126	200,001.0	1.2	388.8	27.24
IOPS	58	102,119.2	46.5	2312.3	3.33
KDD21 [42]	250	77,415.06	1	196.5	10.67
MGAB [43]	10	100,000.0	10.0	200.0	27.64
MITDB [44]	32	650,000.0	210.1	72,334.3	7.19
NAB [45]	58	6301.7	2.0	575.5	2.67
NASA-MSL [46]	27	2730.7	1.33	286.3	1.97
NASA-SMAP [46]	54	8066.0	1.26	1032.4	4.18
Occupancy [47]	10	5725.8	18.3	1414.5	2.5
OPPORTUNITY [48]	465	31,616.9	2.0	1267.3	2.94
SensorScope [49]	23	27,038.4	11.2	6110.4	2.38
SMD [21]	281	25,562.3	10.4	900.2	3.39
SVDB [50]	115	230,400.0	208.0	27,144.5	7.14
YAHOO [51]	367	1561.2	5.9	10.7	3.25

4.3.1. Benchmark Algorithms

We compare our approach LSTMAE-UQ to a row of other algorithms: Isolation Forest [52], Local Outlier Factor [53], HBOS [54], NormA [55], One-Class Support Vector Machine [56], PCA [57], Convolutional Neural Network [58], LSTM-AD [15], AE, AE-UQ and LSTMAE. All baseline algorithms are implemented and accessible via the TSB-UAD (https://github.com/TheDatumOrg/TSB-UAD (accessed on 1 September 2024)).

4.3.2. Results

Figure 11 gives an overview of the results of the benchmark per algorithm for the four selected metrics AUC , precision, recall and F-score. For this representation, the performance results of all of the 1980 time series are taken into account. The proposed algorithm is in each plot on the right, highlighted in blue.

The plots show that LOF or Isolation Forest generally outperform the proposed LSTMAE-UQ method in terms of AUC, precision, recall, and F1-Score across multiple datasets. However, there are certain use cases for the proposed method in specific scenarios. Moreover, traditional anomaly detection algorithms, such as LOF (Local Outlier Factor) and Isolation Forest, lack interpretability when explaining the reason behind anomalies becomes crucial. Consequently, traditional methods may have difficulties adapting to the inherent uncertainties in time series data, which can lead to missed anomalies or a higher rate of false positives in certain scenarios. In contrast, LSTMAE-UQ is designed to address

both aleatoric uncertainty, which captures noise in the data, and epistemic uncertainty, which reflects model uncertainty.

Comparing the LSTMAE to its baselines LSTM and AE we note a couple of things. First compared to LSTM, we notice a superior median AUC. Medians of precision, recall and F-score are a bit lower, but the upper bound of the box is always higher. Secondly, compared to AE, performance is mostly similar over all metrics.

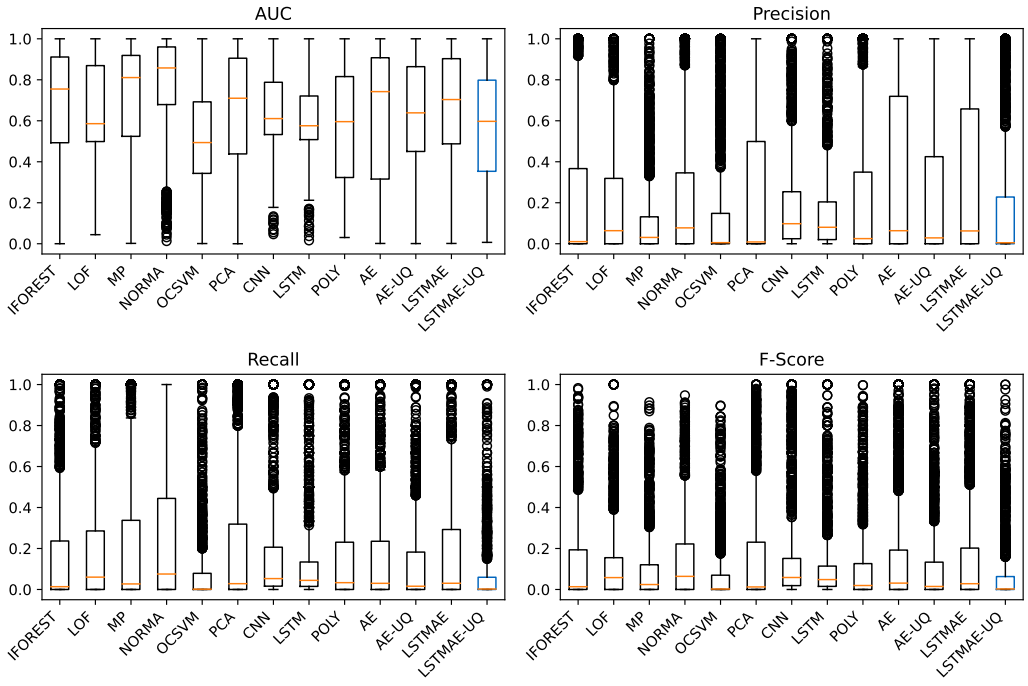
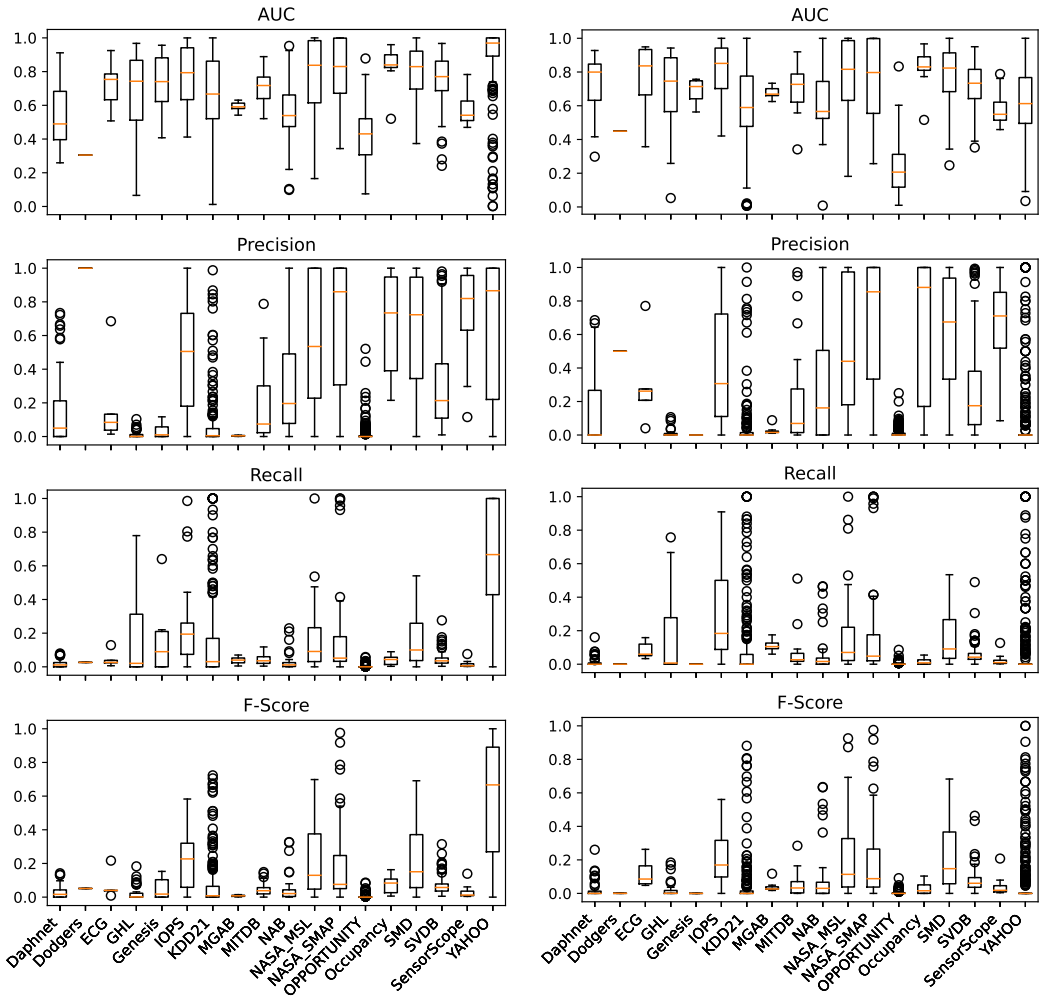


Figure 11. Performance evaluation per algorithm over all datasets. The proposed algorithm is highlighted with a blue box.

Evaluating the performance of our LSTMAE-UQ, we can see that the model shows rather well results for the AUC value. With a median of 0.60 it is comparable to OCSVM, POLY, or the baseline LSTM with a higher variability of results. Compared to its direct baseline, the LSTMAE, we notice a decrease in median AUC. Comparing AE and the version with uncertainty quantification, we notice a similar decrease. Taking a look at the other metrics, we receive relatively good precision compared to other values but rather low recall and F-score with medians close to 0. While the upper box bound for precision is generally comparably high, it is rather low for recall and precision. Also, compared to its baselines, the versions of AE and LSTMAE with uncertainty quantification show lower performance. Figure 12 compares the performance of the LSTMAE-UQ to the baseline LSTMAE. In a direct comparison of performance metrics between the two algorithms, we are able to mostly observe similar performance on most datasets. On the ECG dataset, the proposed approach is clearly able to outperform its baseline in all metrics. Despite achieving mostly similar results, it shows significant differences for the YAHOO dataset. While the LSTMAE baseline shows the best results compared to all remaining datasets, LSTMAE-UQ shows particularly low performance.

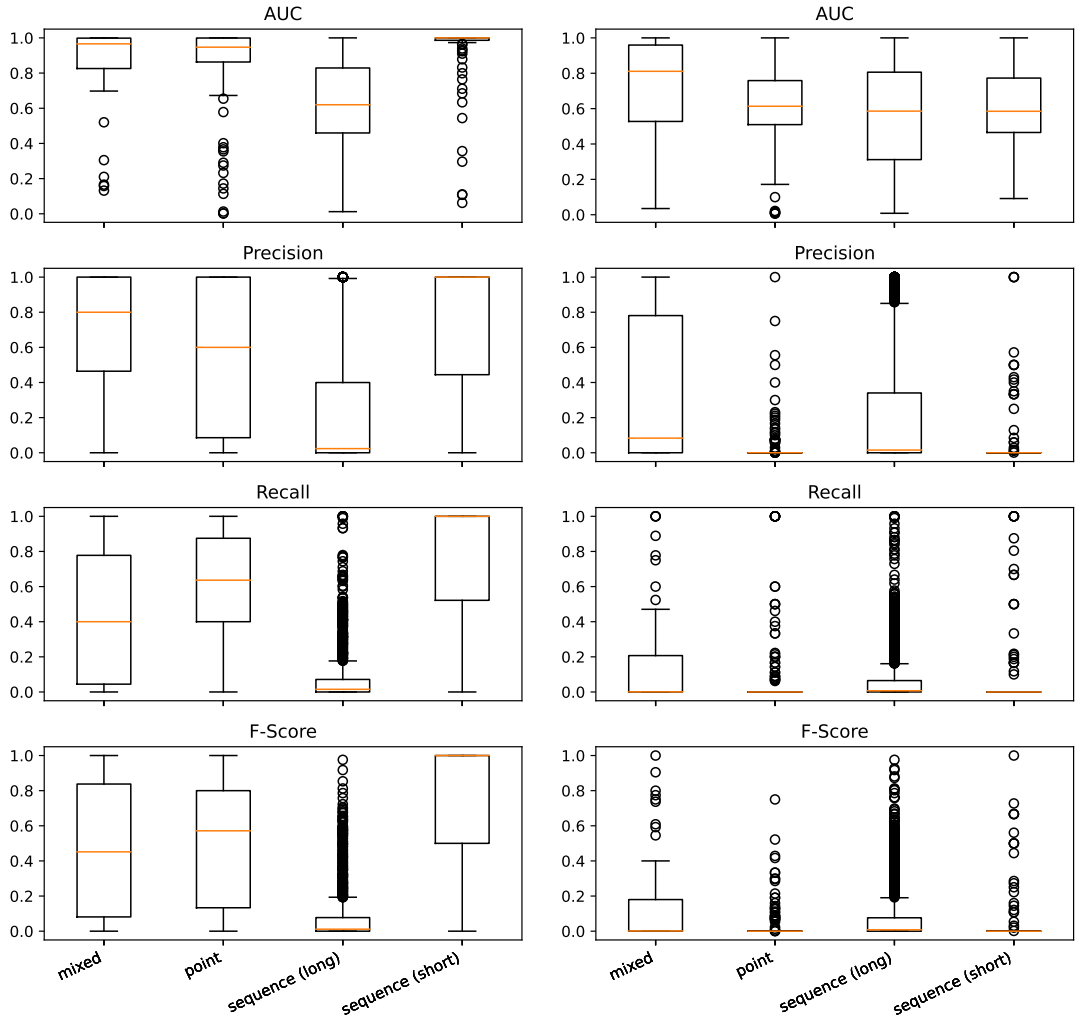
Further analysis of the anomaly types provides deeper insights. Table A1 (Appendix A) breaks down the datasets into four types of anomaly: point anomaly, short/long sequence anomaly and mixed anomalies. Point anomaly refers to an anomaly with length 1. Short sequence anomaly is defined as anomalies with an average duration of up to and including 10 data points with long anomalies starting from an average duration of more than 10. Mixed anomaly refers to a time series that contains point as well as short/long sequence anomalies. From this breakdown, we clearly see what the YAHOO dataset distinguishes. While the predominantly present anomaly type for most datasets is

long sequences, the time series of YAHOO datasets mainly contain point and short sequence anomalies. The only other dataset in this collection that contains point and short sequence anomalies is KDD21. Retrospectively, comparing the performance between LSTMAE and LSTMAE-UQ in Figure 12, we observe a similar decrease, yet not as severe as for YAHOO since the proportion of point and short sequence anomalies to long sequence anomalies is lower. To provide evidence about lower anomaly detection capabilities for point and short sequence anomalies, we evaluate the results on the anomaly types. Figure 13 shows a breakdown for the models LSTMAE and LSTMAE-UQ. We can observe stable behavior between models for long sequence anomalies among all metrics. However, comparing point and short sequence anomalies, LSTMAE-UQ shows a significant decrease in performance. The same accounts for time series with mixed anomaly types. The reason for that could be the consideration of uncertainty in the model. Consideration of uncertainty makes the model count point anomalies as well as short sequences deviating from normal patterns as noise and learns to ignore them. This hypothesis finds support in the findings of the experiment about the behavior of uncertainty during anomalies, where the impact of anomalies, especially for longer input windows, remains low for point anomalies.



(**a**) LSTMAE (**b**) LSTMAE-UQ

Figure 12. Performance evaluation of LSTM Autoencoder without and with aleatoric and epistemic uncertainty quantification for the real-world datasets of the TSB-UAD.



(**a**) LSTMAE (**b**) LSTMAE-UQ

Figure 13. Performance evaluation of the LSTM Autoencoder with uncertainty quantification split up according to anomaly type. Sequence (short) is defined as a sequence up to 10 data points, Sequence (long) contains all sequences longer than 10 points.

4.3.3. Discussion

In this experiment, we evaluated the performance of the LSTMAE-UQ against its baselines and to other algorithms on 18 real-world datasets. Assessing the performance of the LSTMAE in comparison to both baseline models, LSTM and AE, it becomes clear that performance is comparable. Despite the computational expense connected with LSTMs, especially when compared to conventional neurons, their added value in univariate time series anomaly detection is rather low. However, the versatility of LSTMs in handling both univariate and multivariate time series renders them attractive across a broad spectrum of applications.

Continuing with the evaluation of the LSTMAE-UQ model, notable observations include a relatively robust AUC value. However, precision stands out as relatively stronger in comparison to recall and F-score, although all metrics remain relatively low. This suggests that the model possesses a comparably good understanding of classes, and if it identifies an anomaly, the likelihood of it being a true anomaly is relatively high. Nevertheless, the algorithm shows shortcomings in accurately identifying the anomaly.

A comparison between LSTMAE-UQ and the baseline LSTMAE reveals a decrease in performance across all metrics. Diving deeper into the details, this decline can be led back to a particular anomaly type. While no significant difference is observed for longer sequence anomalies, the model shows shortcomings in detecting short sequence and point anomalies in comparison to its baseline. The reason for that could be consideration of uncertainty in the model. Consideration of uncertainty makes the model count point anomalies as well as short sequences deviating from normal patterns as noise and learns to ignore them. This hypothesis finds support in the findings of the experiment about the behavior of uncertainty during anomalies, where the impact of anomalies, especially for longer input windows, remains low for point anomalies. Adjusting sensitivity and window size could enhance the model's robustness across all anomaly types. In the overall picture, the presented algorithm is in the median to lower performance realm. Previously discovered properties explained this result. However, it might find application in areas where noise in data should be explicitly considered or where short and point anomalies are intentionally disregarded.

5. Conclusions

This work introduces LSTMAE-UQ, a novel approach in this work, we developed the LSTMAE-UQ approach to time series anomaly detection under uncertainty. Different from previous approaches, this work takes into consideration both types of uncertainty, aleatoric and epistemic based on the Bayesian posterior approximation method Monte Carlo (MC) Dropout. Experiments conducted on synthetic and real datasets emphasize the need for nuanced approaches in anomaly detection, taking into account the complex interplay between noise, anomalies, and model uncertainty.

In future work, we plan to compare LSTMAE-UQ with other uncertainty quantification methods. We are also particularly interested in investigating LSTMAE-UQ's performance in real-world data streams and exploring methods to quantify uncertainty specifically tailored for the IoT environment, considering factors like sensor noise and data transmission errors. Another key perspective for future work is to improve the interpretability of LSTMAE-UQ's uncertainty outputs. Currently, the model assigns uncertainty values to data points, but we lack a clear understanding of why the model assigns higher uncertainty to specific instances. By investigating methods to make these uncertainties more interpretable, we could gain valuable insights into the model's reasoning process. This interpretability would be crucial for building trust in LSTMAE-UQ's predictions. With a clearer understanding of the uncertainties, users could not only rely on the anomaly detection results but also understand the rationale behind the model's decisions. This is particularly important in safety-critical applications where justifying model outputs is essential.

Author Contributions: Conceptualization, P.W.; methodology, P.W., G.B., S.S. and R.C.; software, P.W.; validation, P.W.; formal analysis, P.W., S.S. and R.C.; investigation, P.W.; resources, P.W.; data curation, P.W.; writing—original draft preparation, P.W.; writing—review and editing, P.W., G.B., S.S., R.C. and H.-J.B.; visualization, P.W.; supervision, S.S. and R.C.; project administration, S.S.; funding acquisition, S.S. and R.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Code is available at https://github.com/p199671/LSTM-AE-UQ.git (accessed on 1 September 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Table A1. Breakdown of anomaly types according to the dataset. Sequence (short) is defined as a sequence up to and including 10 data points, Sequence (long) contains all sequences longer than 10 points.

Dataset	Point	Sequence (Short)	Sequence (Long)	Mixed
Daphnet	0	0	45	0
Dodgers	0	0	0	1
ECG	0	0	5	0
GHL	0	0	125	0
Genesis	0	0	6	0
IOPS	0	4	25	0
KDD21	17	7	223	0
MGAB	0	0	10	0
MITDB	0	0	32	0
NAB	0	0	52	0
NASA-MSL	0	0	27	0
NASA-SMAP	0	0	54	0
OPPORTUNITY	0	0	465	0
Occupancy	0	0	0	10
SMD	0	0	277	0
SVDB	0	0	115	0
SensorScope	0	0	23	0
YAHOO	232	82	14	36
Total	249	93	1498	47

References

1. Schmidl, S.; Wenig, P.; Papenbrock, T. Anomaly detection in time series: A comprehensive evaluation. *Proc. VLDB Endow.* **2022**, *15*, 1779–1797. [CrossRef]
2. Cofta, P.; Karatzas, K.; Orłowski, C. A conceptual model of measurement uncertainty in iot sensor networks. *Sensors* **2021**, *21*, 1827. [CrossRef] [PubMed]
3. Abdar, M.; Pourpanah, F.; Hussain, S.; Rezazadegan, D.; Liu, L.; Ghavamzadeh, M.; Fieguth, P.; Cao, X.; Khosravi, A.; Acharya, U.R.; et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Inf. Fusion* **2021**, *76*, 243–297. [CrossRef]
4. Neal, R.M. *Bayesian Learning for Neural Networks*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012; Volume 118.
5. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–12.
6. Chatterjee, A.; Ahmed, B.S. IoT anomaly detection methods and applications: A survey. *Internet Things* **2022**, *19*, 100568. [CrossRef]
7. Vidmark, A. Anomaly or not Anomaly, that is the Question of Uncertainty: Investigating the Relation Between Model Uncertainty and Anomalies Using a Recurrent Autoencoder Approach to Market Time Series. *DiVA* 2022. Available online: http://www.diva-portal.org/smash/get/diva2:1700446/FULLTEXT01.pdf (accessed on 1 September 2024)
8. Kendall, A.; Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11.
9. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
10. Michelucci, U. An Introduction to Autoencoders. *arXiv* **2022**, arXiv: 2201.03898.
11. Gal, Y.; Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2016; pp. 1050–1059.

12. Darban, Z.Z.; Webb, G.I.; Pan, S.; Aggarwal, C.C.; Salehi, M. Deep Learning for Time Series Anomaly Detection: A Survey. *CoRR* **2022**, *57*, 1–42.
13. Yan, P.; Abdulkadir, A.; Luley, P.; Rosenthal, M.; Schatte, G.A.; Grewe, B.F.; Stadelmann, T. A Comprehensive Survey of Deep Transfer Learning for Anomaly Detection in Industrial Time Series: Methods, Applications, and Directions. *IEEE Access* **2024**, *12*, 3768–3789. [CrossRef]
14. Chauhan, S.; Vig, L. Anomaly detection in ECG time signals via deep long short-term memory networks. In Proceedings of the 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Paris, France, 19–21 October 2015; pp. 1–7.
15. Malhotra, P.; Vig, L.; Shroff, G.; Agarwal, P. Long short term memory networks for anomaly detection in time series. In Proceedings of the Esann, Bruges, Belgium, 22–23 April 2015; Volume 2015, p. 89.
16. Chen, Z.; Yeo, C.K.; Lee, B.S.; Lau, C.T. Autoencoder-based network anomaly detection. In Proceedings of the 2018 Wireless Telecommunications Symposium (WTS), Phoenix, AZ, USA, 7–20 April 2018; pp. 1–5.
17. Sakurada, M.; Yairi, T. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, Gold Coast, QLD, Australia, 2 December 2014; pp. 4–11.
18. Xu, H.; Chen, W.; Zhao, N.; Li, Z.; Bu, J.; Li, Z.; Liu, Y.; Zhao, Y.; Pei, D.; Feng, Y.; et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018; pp. 187–196.
19. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. In Proceedings of the 2nd International Conference on Learning Representations, ICLR 2014; Conference Track Proceedings; Banff, AB, Canada, 14–16 April 2014.
20. Malhotra, P.; Ramakrishnan, A.; Anand, G.; Vig, L.; Agarwal, P.; Shroff, G. LSTM-based encoder-decoder for multi-sensor anomaly detection. *arXiv* **2016**, arXiv:1607.00148.
21. Su, Y.; Zhao, Y.; Niu, C.; Liu, R.; Sun, W.; Pei, D. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2828–2837.
22. Zhu, L.; Laptev, N. Deep and confident prediction for time series at uber. In Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDMW), New Orleans, LA, USA, 8–21 November 2017; pp. 103–110.
23. Wong, L.; Liu, D.; Berti-Equille, L.; Alnegheimish, S.; Veeramachaneni, K. AER: Auto-encoder with regression for time series anomaly detection. In Proceedings of the 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, 17–20 December 2022; pp. 1152–1161.
24. Hüllermeier, E.; Waegeman, W. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Mach. Learn.* **2021**, *110*, 457–506. [CrossRef]
25. Ma, J.; Sun, L.; Wang, H.; Zhang, Y.; Aickelin, U. Supervised anomaly detection in uncertain pseudoperiodic data streams. *ACM Trans. Internet Technol. (TOIT)* **2016**, *16*, 1–20. [CrossRef]
26. Ul Islam, R.; Hossain, M.S.; Andersson, K. A novel anomaly detection algorithm for sensor data under uncertainty. *Soft Comput.* **2018**, *22*, 1623–1639. [CrossRef]
27. Hennhöfer, O.; Preisach, C. Uncertainty Quantification in Anomaly Detection with Cross-Conformal *p*-Values. *arXiv* **2024**, arXiv:2402.16388.
28. Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; Wierstra, D. Weight uncertainty in neural network. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 6–11 July 2015; pp. 1613–1622.
29. Bernardo, J.M.; Smith, A.F. *Bayesian Theory*; John Wiley & Sons: Hoboken, NJ, USA, 2009; Volume 405. Available online: https://statisticalsupportandresearch.wordpress.com/wp-content/uploads/2019/03/josc3a9-m.-bernardo-adrian-f.-m.-smith-bayesian-theory-wiley-1994.pdf (accessed on 1 September 2024).
30. Legrand, A.; Trannois, H.; Cournier, A. Use of uncertainty with autoencoder neural networks for anomaly detection. In Proceedings of the 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), Sardinia, Italy, 3–5 July 2019; pp. 32–35.
31. Daxberger, E.; Hernández-Lobato, J.M. Bayesian variational autoencoders for unsupervised out-of-distribution detection. *arXiv* **2019**, arXiv:1912.05651.
32. Foldesi, L.; Valdenegro-Toro, M. Comparison of Uncertainty Quantification with Deep Learning in Time Series Regression. *arXiv* **2022**, arXiv:2211.06233.
33. Chen, J.; Pi, D.; Wu, Z.; Zhao, X.; Pan, Y.; Zhang, Q. Imbalanced satellite telemetry data anomaly detection model based on Bayesian LSTM. *Acta Astronaut.* **2021**, *180*, 232–242. [CrossRef]
34. Yong, B.X.; Brintrup, A. Bayesian autoencoders with uncertainty quantification: Towards trustworthy anomaly detection. *Expert Syst. Appl.* **2022**, *209*, 118196. [CrossRef]
35. Wenig, P.; Schmidl, S.; Papenbrock, T. TimeEval: A Benchmarking Toolkit for Time Series Anomaly Detection Algorithms. *Proc. VLDB Endow.* **2022**, *15*, 3678–3681. [CrossRef]
36. Shaikh, S.A.; Kitagawa, H. Top-k outlier detection from uncertain data. *Int. J. Autom. Comput.* **2014**, *11*, 128–142. [CrossRef]
37. Bachlin, M.; Plotnik, M.; Roggen, D.; Maidan, I.; Hausdorff, J.M.; Giladi, N.; Troster, G. Wearable assistant for Parkinson's disease patients with the freezing of gait symptom. *IEEE Trans. Inf. Technol. Biomed.* **2009**, *14*, 436–446. [CrossRef] [PubMed]

38. Ihler, A.; Hutchins, J.; Smyth, P. Adaptive event detection with time-varying poisson processes. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Long Beach, CA, USA, 6–10 August 2006; pp. 207–216.
39. Moody, G.B.; Mark, R.G. The impact of the MIT-BIH arrhythmia database. *IEEE Eng. Med. Biol. Mag.* **2001**, *20*, 45–50. [CrossRef] [PubMed]
40. von Birgelen, A.; Niggemann, O. Anomaly detection and localization for cyber-physical production systems with self-organizing maps. In *IMPROVE-Innovative Modelling Approaches for Production Systems to Raise Validatable Efficiency: Intelligent Methods for the Factory of the Future*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 55–71.
41. Filonov, P.; Lavrentyev, A.; Vorontsov, A. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv* **2016**, arXiv:1612.06676.
42. Keogh, E.; Taposh, D.R.; Naik, U.; Agrawal, A. Multi-dataset time-series anomaly detection competition. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Anchorage, AK, USA, 4–8 August 2021.
43. Thill, M.; Konen, W.; Bäck, T. MarkusThill/MGAB: The Mackey-Glass Anomaly Benchmark; Version v1. 0.1.; Zenodo: Geneva, Switzerland, 2020; Volume 10.
44. Moody, G.B.; Mark, R. *MIT-BIH Arrhythmia Database Directory*; MITBIH Database Distribution, Harvard—MIT Division of Health Sciences and Technology, Massachusetts Institute of Technology: Cambridge, MA, USA, 1992. Available online: http://www.physionet.org/physiobank/database/html/mitdbdir/mitdbdir.htm (accessed on 23 July 2008).
45. Ahmad, S.; Lavin, A.; Purdy, S.; Agha, Z. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing* **2017**, *262*, 134–147. [CrossRef]
46. Benecki, P.; Piechaczek, S.; Kostrzewa, D.; Nalepa, J. Detecting anomalies in spacecraft telemetry using evolutionary thresholding and LSTMs. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Lille, France, 10–14 July 2021; pp. 143–144.
47. Candanedo, L.M.; Feldheim, V. Accurate occupancy detection of an office room from light, temperature, humidity and CO_2 measurements using statistical learning models. *Energy Build.* **2016**, *112*, 28–39. [CrossRef]
48. Roggen, D.; Calatroni, A.; Rossi, M.; Holleczek, T.; Förster, K.; Tröster, G.; Lukowicz, P.; Bannach, D.; Pirkl, G.; Ferscha, A.; et al. Collecting complex activity datasets in highly rich networked sensor environments. In Proceedings of the 2010 Seventh International Conference on Networked Sensing Systems (INSS), Kassel, Germany, 15–18 June 2010; pp. 233–240.
49. Yao, Y.; Sharma, A.; Golubchik, L.; Govindan, R. Online anomaly detection for sensor systems: A simple and efficient approach. *Perform. Eval.* **2010**, *67*, 1059–1075. [CrossRef]
50. Greenwald, S.D.; Patil, R.S.; Mark, R.G. *Improved Detection and Classification of Arrhythmias in Noise-Corrupted Electrocardiograms Using Contextual Information*; IEEE: Piscataway, NJ, USA, 1990.
51. Laptev, N.; Amizadeh, S.; Billawala, Y. *S5-A Labeled Anomaly Detection Dataset, Version 1.0 (16M)*; Trinity College: Dublin, Ireland, 2015.
52. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422.
53. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 15–18 May 2000; pp. 93–104.
54. Goldstein, M.; Dengel, A. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012 Poster Demo Track* **2012**, *1*, 59–63.
55. Boniol, P.; Linardi, M.; Roncallo, F.; Palpanas, T.; Meftah, M.; Remy, E. Unsupervised and scalable subsequence anomaly detection in large data series. *VLDB J.* **2021**, *30*, 909–931. [CrossRef]
56. Schölkopf, B.; Williamson, R.C.; Smola, A.; Shawe-Taylor, J.; Platt, J. Support vector method for novelty detection. *Adv. Neural Inf. Process. Syst.* **1999**, *12*, 588.
57. Aggarwal, C.C.; Aggarwal, C.C. *An Introduction to Outlier Analysis*; Springer: Berlin/Heidelberg, Germany, 2017.
58. Munir, M.; Siddiqui, S.A.; Dengel, A.; Ahmed, S. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access* **2018**, *7*, 1991–2005. [CrossRef]