Anomaly Detection Approaches for Sequence Data

1 Introduction

Sequence data, also known as time-series data [1], is a set of time-ordered records [2]. Large volumes of real-world time-series data are increasingly collected from various sources, such as Internet of Things (IoT) sensors, network servers, and patient medical flow reports [2, 3, 4]. These records can get corrupted because of how the data is collected, transformed, and managed, and also because of malicious activities. Inaccurate data can lead to incorrect decisions. Thus, rigorous data quality testing approaches are required to ensure that the data is correct. The sequence records have typically strong correlation and dependency, which cannot be analysed independently for data quality testing [5]. As a result, the existing analysis and testing approaches for independent data records cannot be used for testing the sequence data.

Data quality tests for time series data validate the data in a sequence data store to detect violations of constraints over multiple records in time series. For example, semantic constraint validations check that the *patient_weight* growth rate change is positive and in the range [4, 22] lb for every infant. The validations also check for the relationship between the *patient_weight* and *blood_pressure*, and their growth rates over time for the adult patients.

There are different types of anomalies as constraint violations in the time series data. An *anomalous record* (outlier) is a record with unexpected values for its attributes in a time series. An *anomalous Sequences* is a sequence whose behaviour is significantly different from what is expected.

Systematic testing and evaluation techniques have been proposed by researchers and practitioners to detect each types of anomalies in sequence data. In this paper, we present a comprehensive survey by defining a classification framework for the testing and evaluation techniques applied to detect different types of anomalies. Our classification framework is based on *what* type of anomalies is detected in sequence data, and *how* it is detected through categorizing the different data quality testing approaches. We also discuss open problems and propose research directions.

2 Sequence Data

A time series T is a sequence of d-dimensional records [2] described using the vector
$T =< R_0, ..., R_{n-1} >$, where $R_i = (a_i^0, ..., a_i^{d-1})$ is a record at time i, for $0 \leq i \leq n-1$ and a_i^j is the j^{th} attribute of the i^{th} record. Existing data analysis approaches [2] assume that the time gaps between any pair of consecutive records differ by less than or equal to an epsilon value, i.e., the differences between the time stamps of any two consecutive records are nearly the same.

A time series can be *univariate* (d=1) or *multivariate* (d>1) [3]. A univariate time series has one time-dependent attribute. For example, a univariate time series can consist of daily temperatures recorded sequentially over 24-hour increments. A multivariate time series is used to simultaneously capture the dynamic nature of multiple attributes. For example, a multivariate time series from a climate data store [6] can consist of precipitation, wind speed, snow depth, and temperature data.

The research literature [7, 9] uses various features that describe the relationships among the time-series records and attributes. Trend and seasonality [10] are the most commonly used features. Trend is defined as the general tendency of a time series to increment, decrement, or stabilize over time [10]. For example, there may be an upward trend for the number of patients with cancer diagnosis. Seasonality is defined as the existence of repeating cycles in a time series [10]. For example, the sales of swimwear is higher during summers. A time series is *stationary* (non-seasonal) if all its statistical features, such as mean and variance are constant over time. Table 1 shows a set of features defined by Talagala et al. [7] to describe a time series.

A constraint is defined as a rule over the time-series features. For example, the mean (F_1) value of the daily electricity power delivered by a household must be in the range 0.1–0.5 KWH. We categorize the faults that can violate the constraints over time-series features as *anomalous records* and *anomalous sequences*.

Anomalous records. Given an input time series T, an anomalous record R_t is one whose observed value is significantly different from the expected value

Table 1: Time Series Features [7]

Feature	Description
F_1: Mean	Mean value of time series
F_2: Variance	Variance value of time series
F_3: Lumpiness	Variance of the variances across multiple blocks in time series
F_4: Lshift	Maximum difference in mean between consecutive blocks in time series
F_5: Vchange	Maximum difference in variance between consecutive blocks in time series
F_6: Linearity	Strength of linearity, which is the sum of squared residuals of time series from a linear autoregression
F_7: Curvature	Strength of curvature, which is the amount by which a time series curve deviates from being a straight line and calculated based on the coefficients of an orthogonal quadratic regression
F_8: Spikiness	Strength of spikiness, which is calculated based on the size and location of the peaks and troughs in time series
F_9: Season	Strength of seasonality, which is calculated based on a robust STL [8] decomposition
F_{10}: Peak	Strength of peaks, which is calculated based on the size and location of the peaks in time series
F_{11}: Trough	Strength of trough, which is calculated based on the size and location of the troughs in time series
F_{12}: BurstinessFF	Ratio between the variance and the mean (Fano Factor) of time series
F_{13}: Minimum	Minimum value of time series
F_{14}: Maximum	Maximum value of time series
F_{15}: Rmeaniqmean	Ratio between interquartile mean and the arithmetic mean of time series
F_{16}: Moment3	Third moment, which is a quantitative measure that identifies the skewness of time series
F_{17}: Highlowmu	Ratio between the means of data that is below and upper the global mean of time series
F_{18}: Trend	Strength of trend, which is calculated based on a robust STL decomposition

of T at t. An anomalous record may violate constraints over the features F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_{12}, F_{13}, F_{14}, F_{15}, F_{16}, and F_{17}. For example, if there is a constraint that imposes a range of values (F_{13}, F_{14}) for the infant patients' weights during their first three months, a record in the first three months with a weight value outside this range must be reported as faulty.

Anomalous sequences. Given a set of subsequences $T = \{T_0, ..., T_{m-1}\}$ in a time series T, a faulty sequence $T_j \in T$ is one whose behavior is significantly different from the majority of subsequences in T. An anomalous sequence may violate constraints over any of the features F_1 through F_{18}. For example, consider the constraint that imposes an upward trend (F_{18}) for the number of cars passing every second at an intersection from 6 to 7 am on weekdays. A decrease in this trend is anomalous.

3 Running Examples

We use the Yahoo server traffic datasets in the Yahoo Webscore program [11] and the NASA Shuttle dataset in the UCI ML repository [12]. The Yahoo server traffic datasets contain real and synthetic univariate time series, each of which contains 1,420 time ordered records with one time-dependent attribute called *traffic_value*. These datasets contain time series with random seasonality, trend and noise. Each data-point represents one hour's worth of traffic data. Table 2 shows the schema of the Yahoo server traffic table. Anomalies in the *Traffic_value* indicate potential security threats to the Yahoo user's data.

Table 2: Schema of *Yahoo* Server Traffic Table

Attribute Name	Data Type	Description
ID	Numeric	Unique
Time	DateTime	Nullable
Traffic_value	Float	Nullable

The NASA Shuttle multivariate dataset contains 58,000 time-ordered records with eight time-dependent numerical attributes, namely, A_1 to A_8. Table 3 shows the schema of the NASA Shuttle table. Incorrect values in these attributes have negative consequences for the aerospace industry that conducts research, designs, manufactures, operates, and maintains the spacecrafts.

Table 3: Schema of *NASA Shuttle* Table

Attribute Name	**Data Type**	**Description**
ID	Numeric	Unique
Time	DateTime	Nullable
A_1	Float	Nullable
A_2	Float	Nullable
A_3	Float	Nullable
A_4	Float	Nullable
A_5	Float	Nullable
A_6	Float	Nullable
A_7	Float	Nullable
A_8	Float	Nullable

4 Anomaly Detection in Sequence Data

Machine Learning-based techniques for outlier detection from non-sequence data, such as Support Vector Machine (SVM) [13], Local Outlier Factor (LOF) [14], Isolation Forest (IF) [15], and Elliptic Envelope (EE) [16] have been used in the literature to detect anomalous records from a time series [17]. These approaches discover the constraints in individual data records and cannot be used for testing time-series data as constraints may exist over multiple attributes and records in a time series. The records in a sequence have strong correlations and dependencies with each other, and constraint violations over multiple records cannot be discovered by analyzing records in isolation [5].

We classify the approaches that detect anomalies in time-series data into two groups based on anomaly types they can detect from input datasets; these are anomalous record detection and anomalous sequence detection. Figure 1 shows the classification framework we propose for anomaly detection techniques based on anomaly types they can detect in time-series data. The framework presents *what* is detected in terms of anomaly types and *how* they are detected. A rounded rectangle represents a class and an edge rectangle represents a technique.

4.1 Approaches to Detect Anomalous Records

We categorize these approaches based on how they analyze the time-series data as *time series modeling* and *time series decomposition* techniques.

4.1.1 Time Series Modeling Techniques

Given a time series $T = \{R_t\}$, these techniques model the time series as a linear/non-linear function f that associates current value of a time series to its past values. Next, the techniques use f to provide the predicted value of R_t at time t, denoted by R'_t, and calculate a prediction error $PE_t = |R_t - R'_t|$. The techniques report R_t as outlier if the prediction error falls outside a fixed threshold value. Every model f has a set of parameters, which are estimated using *stochastic* or *machine learning* techniques.

In the stochastic modeling techniques, a time series is considered as a set of random variables $T = \{R_t,\ t = 0, ..., n\}$, where R_t is from a certain probability model [10]. Examples of these techniques are *Autoregressive (AR)*, *Moving Average (MA)*, and *Autoregressive Integrated Moving Average (ARIMA)* and *Holt-Winters (HW)* models.

Autoregressive (AR) models [18]. In an Autoregressive model, the current value of a record in a time series is a linear combination of the past record values plus a random error. An autoregressive model makes an assumption that the data records at previous time steps (called as lag variables) can be used to predict the record at the next time step. The relationship between data records is called correlation. Statistical measures are typically used to calculate the correlation between the current record and the records at previous time steps. The stronger the correlation between the current record and a specific lagged variable, the more weight the autoregressive model puts on that variable. If all previous records show low or no correlation with the current one, then the time series problem may not be predictable [19]. Equation 1 shows the mathematical expression for an AR model.

$$R_t = \sum_{i=1}^{p} A_i R_{t-i} + E_t \qquad (1)$$

where R_t is the record at time t and p is the order of the model. For example, an autoregressinve model of order two indicates that the current value of a time series is a linear combination of the two immediately preceding records plus a random error. The coefficients $A = (A_1, ..., A_p)$ are weights applied to each of the past records. The random errors (noises) E_t are

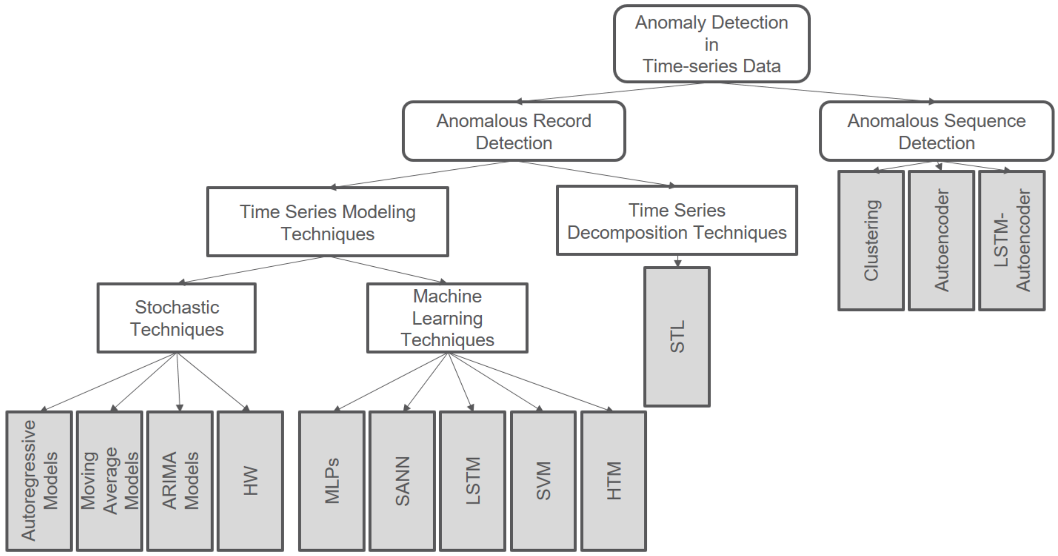

Figure 1: Classification Framework for Anomaly Detection Approaches for Sequence Data

assumed to be independent and following a Normal $N(0, \sigma^2)$ distribution. Given the time series T, the objective of AR modeling is to estimate the model parameters (A, σ^2). The linear regression estimators [20], likelihood estimators [21], and Yule-Walker equations [10] are typical stochastic techniques used to estimate this model parameters.

The AR model is only appropriate for modeling univariate stationary time-series data [10]. Moreover, it does not consider the non-linear associations between the data records in a time series.

Moving Average (MA) models [22]. In these models, a data record at time t is a linear combination of the random errors that occurred in past time periods (i.e. E_{t-1}, E_{t-2},...,E_{t-p}). Equation 2 shows the mathematical expression for an MA model.

$$R_t = \mu \sum_{i=1}^{p} B_i E_{t-i} + E_t \tag{2}$$

Where μ is the series mean, p is the order of the model, and $B = (B_1, ..., B_p)$ are weights applied to each of the past errors. The random errors E_t are assumed to be independent and following a Normal $N(0, \sigma^2)$ distribution.

The MA model is appropriate for univariate stationary time series modeling [10]. Moreover, it is more complicated to fit an MA model to a time series than fitting an AR model. Because in an MA model, the random error terms are not foreseeable [10].

Autoregressive Integrated Moving Average (ARIMA) models [18]. ARIMA is a mixed model, which incorporates: (1) Autoregression (AR) model, (2) an Integrated component, and (3) Moving Average (MA) model. The integrated component stationarized the time series by using transformations like differencing [23], logging [24], and deflating [25]. ARIMA can model time series with non-stationary behaviour. However, this model assumes that the time series is linear and follows a known statistical distribution, which makes it inapplicable to many practical problems [10].

Holt-Winters (HW [26]). This technique uses exponential smoothing [27] to model three features of a time series: (1) mean value, (2) trend, and (3) seasonality. Exponential smoothing assigns to past records exponentially decreasing weights over time. The objective is to decrease the weight put on older data records. Three types of exponential smoothing (i.e., triple exponential smoothing) are performed for the three features of a time series. The model requires multiple hyper-parameters: one for each smoothing, one for the length of a season, and one for the number of periods in a season. Hasani et al. [26] enhanced this technique (HW-GA) using a Genetic Algorithm [28] to optimize the HW hyper-parameters. The HW model is only appropriate for modeling univariate time-series data. Moreover, it does not consider the non-linear associations between the data records in a time series.

In Machine Learning-based modeling techniques, a time series is considered to follow a specific pattern. Examples of these techniques are *Multi Layer*

Perceptron (MLP), *Seasonal Artificial Neural Networs (SANN)*, *Long Short Term Network (LSTM)*, and *Support Vector Machine (SVM)* models for big data and *Hierarchical Temporal Memory (HTM)* for streamed data (i.e., data captured in continuous temporal processes).

Multi Layer Perceptron (MLP) [29]. This technique is a type of Artificial Neural Network (ANN) [30], which supports non-linear modeling, with no assumption about the statistical distribution of the data [10]. An MLP model is a fully connected network of information processing units that are organized as input, hidden, and output layers. Equation 3 shows the mathematical expression of an MLP for time series modeling.

$$R_t = b + \sum_{j=1}^{q} \alpha_j g\left(b_j + \sum_{i=1}^{p} \beta_{ij} R_{t-i}\right) + E_t \tag{3}$$

where R_{t-i} $(i = 1, .., p)$ are p network inputs, R_t is the network output, α_j and β_{ij} are the network connection weights, E_t is a random error, and g is a non-linear activation function, such as logistic sigmoid and hyperbolic tangent.

The objective is to train the network and learn the parameters of the non-linear functional mapping f from the p past data records to the current data record R_t (i.e., $R_t = f(R_{t-1}, ..., R_{t-p}, w) + E_t$). Approaches based on minimization of an error function (equation 4) are typically used to estimate the network parameters. Examples of these approaches are Backpropagation and Generalized Delta Rule [30].

$$Error = \sum_t e_t^2 = \sum_t (R_t - R'_t)^2 \tag{4}$$

where R'_t is the actual network output at time t.

An MLP can model non-linear associations between data records. However, it is appropriate for univariate time series modeling. Moreover, because of the limited number of network inputs, it can only discover the short-term dependencies among the data records.

A Seasonal Artificial Neural Network (SANN) model is an extension of MLPs for modeling seasonal time-series data. The number of input and output neurons are determined based on a seasonal parameter s. The records in the i^{th} and $(i+1)^{th}$ seasonal period are used as the values of network input and output respectively. Equation 5 shows the mathematical expression for this model [10].

$$R_{t+l} = \alpha_l + \sum_{j=1}^{m} w_{1jl} g\left(\theta_j + \sum_{i=0}^{s-1} w_{0ij} R_{t-i}\right) \tag{5}$$

where $R_{t+l}(l = 1, .., s)$ are s future predictions based on the s previous data records $(R_{t-i}(i = 0, ..., s-1))$; w_{0ij} and w_{1jl} are connection weights from the input to hidden and from hidden to output neurons respectively; g is a non-linear activation function and α_l and θ_j are network bias terms.

This network can model non-linear associations in seasonal time-series data. However, it is appropriate for modeling univariate time series. Moreover, the values of records in a season are considered to be dependent only on the values of the previous season. As a result, the network can only learn short-term dependencies between data records.

Long Short Term Network (LSTM) [31]. An LSTM is a Recurrent Neural Network (RNN) [32] that contains loops in its structure to allow information to persist and make network learn sequential dependencies among data records [31]. An RNN can be represented as multiple copies of a neural network, each passing a value to its successor. Figure 2 shows the structure of an RNN [32]. In this Figure, A is a neural network, X_t is the network input, and h_t is the network output.

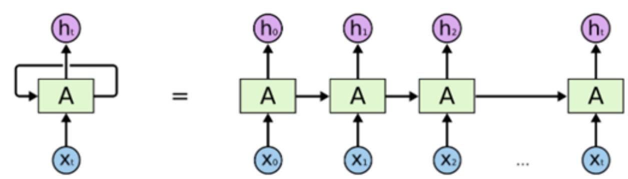

Figure 2: An Unrolled RNN [32]

The original RNNs can only learn short-term dependencies among data records by using the recurrent feedback connections [3]. LSTMs extend RNNs by using specialized gates and memory cells in their neuron structure to learn long-term dependencies.

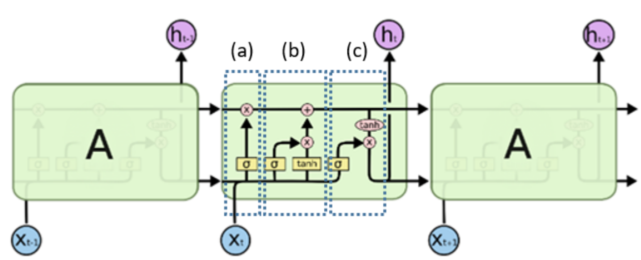

Figure 3: LSTM Structure [32]

Figure 3 shows the structure of an LSTM network. The computational units (neurons) of an LSTM are called *memory cells*. The horizontal line passing through the top of the neuron is called the memory cell state. An LSTM has the ability to remove or add information to the memory cell state by using *gates*. The gates are defined as weighted functions that govern information flow in the memory cells. The gates are composed of a *sigmoid layer* and a *point-wise operation* to optionally let information through. The sigmoid layer outputs a number between zero (to let nothing through) and one (to let everything through).

There are three types of gates, namely, *forget*, *input*, and *output*.

- *Forget gate* (Figure 3 (a)): Decides what information to discard from the memory cell. Equation 6 shows the mathematical representation of the forget gate.

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{6}$$

  where W_f is the connection weight between the inputs (h_{t-1} and x_t) and the sigmoid layer; b_f is the bias term and σ is the sigmoid activation function. In this gate, $f_t = 1$ means that completely keep the information and $f_t = 0$ means that completely get rid of the information.

- *Input gate* (Figure 3 (b)): Decides which values to be used from the network input to update the memory state. Equation 7 shows the mathematical representation of the input gate.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{7}$$

  where C_t is the new memory cell state and C_{t-1} is the old cell state, which is multiplied by f_t to forget the information decided by the forget gate; $\tilde{C}_t$ is the new candidate value for the memory state, which is scaled by i_t as how much the gate decides to update the state value.

- *Output gate* (Figure 3 (c)): Decides what to output based on the input and the memory state. Equation 8 shows the mathematical representation of the output gate. This gate pushes the cell state values between -1 and 1 by using a hyperbolic tangent function and multiplies it by the output of its sigmoid layer to decide which parts of the input and the cell state to output.

$$\begin{aligned} o_t &= \sigma(W_o.[h_{t-1}, x_t] + b_o) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{8}$$

An LSTM network for time series modeling takes the values of p past records (R_{t-i}, $(i = 1, ..., p)$) as input and predicts the value of the current record (R_t) in its output. LSTM modeling techniques can model non-linear long-term sequential dependencies among the data records in univariate/multivariate time series, which makes them more practical to real-world applications. Moreover, LSTMs have the ability to learn seasonality [33]. However, the trained network is a complex equation over the attributes of the data records, which is not human interpretable.

Support Vector Machine (SVM [10]). An SVM model maps the data from the input space into a higher-dimensional feature space using a non-linear mapping (referred to as a Kernel Function) and then performs a linear regression in the new space. The linear model in the new space represents a non-linear model in the original space.

An SVM for time series modeling uses the training data as pairs of input and output, where an input is a vector of p previous data records in the time series and the output is the value of the current data record. Equation 9 shows the mathematical representation of a non-linear SVM regression model.

$$R_t = b + \sum_p \alpha_i \varphi(R_{t-i}) \tag{9}$$

where R_t is the data record at time t, φ is a kernel function, such as Gaussian RBF [34], and R_{t-i} is the i^{th} previous record in the time series.

The SVM modeling techniques can model both linear and non-linear functions for predicting time series values. However, these techniques require an enormous amount of computation, which makes them inapplicable to large datasets [10]. Moreover, the trained model is not human interpretable.

Hierarchical Temporal Memory (HTM [35]). This is an unsupervised technique that continuously models time-series data using a memory based system. An HTM uses online learning algorithms, which store and recall constraints as spatial and temporal patterns in an input dataset. An HTM is a type of neural network whose neurons are arranged in columns, layers, and regions in a time-based hierarchy. This hierarchical organization considerably reduces the training time and memory usage because patterns learned at each level of the hierarchy are reused when combined at higher levels. The learning process of HTM discovers and stores spatial and temporal patterns over time. Once an HTM is trained with a sequence of data, learning new patterns mostly occurs in the upper levels of the hierarchy. An HTM matches an input record to previously learned temporal patterns to predict the next record. It takes longer for an HTM to learn previously unseen patterns. Unlike deep learning techniques that require large datasets to be trained, an HTM requires streamed data. The patterns discovered by this technique are not human interpretable.

4.1.2 Time Series Decomposition Techniques

These techniques decompose a time series into its components, namely level (the average value of data points in a time series), trend (the increasing or decreasing value in the time series), seasonality (the repeating cycle in the time series), and noise (the random variation in the time series) [36, 37]. Next, they monitor the noise component to capture the anomalies. These approaches report as anomalous the data record R_t whose absolute value of noise is greater than a threshold.

These techniques consider the time series as an additive or multiplicative decomposition of level, trend, seasonality, and noise. Equation 10 and 11 shows the mathematical representation of additive and multiplicative models respectively.

$$R_t = l_t + \tau_t + s_t + r_t \tag{10}$$

$$R_t = l_t * \tau_t * s_t * r_t \tag{11}$$

where R_t is the data record at time t, l_t is the level as the average value of data records in a time series, τ_t is the trend in time series, and s_t is the seasonal signal with a particular period, and r_t is the residual of the original time series after the seasonal and trend are removed and is referred to as *noise*, *irregular*, and *remainder*. In this model, s_t can slowly change or stay constant over time.

In a linear additive model the changes over time are consistently made by the same amount. A linear trend is described as a straight line and a linear seasonality has the same frequency (i.e., width of cycles) and amplitude (i.e., height of cycles) [38].

In a non-linear multiplicative model, the changes increase or decrease over time. A non-linear trend is described as a curved line and a non-linear seasonality has increasing or decreasing frequency or amplitude over time [38].

Different approaches are proposed in the literature to decompose a time series into its components. *Seasonal-Trend decomposition using LOESS (STL)* is one of the most commonly used approaches, which is described as follows.

Seasonal-Trend decomposition using LOESS (STL) [40]. This approach uses LOESS (LOcal regrESSion) smoothing technique to detect the time series components. LOESS is a non-parametric smoother that models a curve of best fit through a time series without assuming that the data must follow a specific distribution. This method is a local regression based on a least squares method; it is called local because fitting at point t is weighted towards the data nearest to t. The effect of a neighboring value on the smoothed value at a certain point t decreases with its distance to t. Figure 4 shows an example of the STL decomposition for a liquor sales dataset. This Figure shows the trend, seasonality, and noise components extracted from an original time-series data.

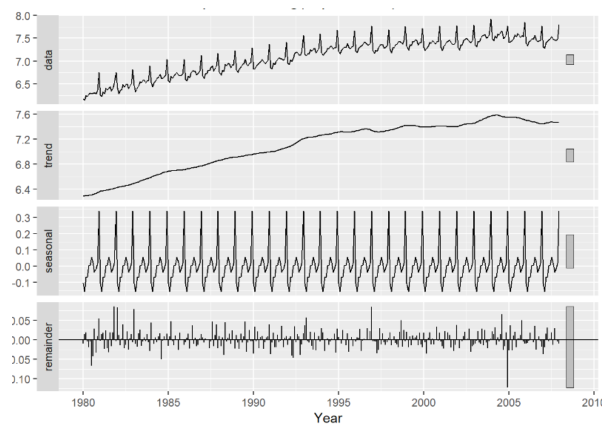

Figure 4: STL Decomposition of Liquor Sales Data [39]

The time series decomposition techniques provide non-complex models that can be used to analyze the time-series data and detect anomalies in the data. However, in real-world applications, we may not be able to model a specific time series as an additive or multiplicative model, since real-world datasets are messy and noisy [38]. Moreover, the decomposition techniques are only applicable to univariate time series data.

4.2 Approaches to Detect Anomalous Sequences

The approaches proposed in the literature to detect anomalous sequences are based on (1) splitting the time-series data into multiple subsequences, typically based on a fixed size overlapping window, and (2) detecting as anomalous those subsequences whose behavior is significantly different from the majority of subsequences in the time series. Examples of these approaches are *Clustering*, *Autoencoder*, and *LSTM-Autoencoder*.

Clustering [37]. These techniques extract subsequence features, such as trend and seasonality. Table 1 shows the time series features from the TS-Features CRAN library [9]. Next, an unsupervised clustering technique, such as K-means [41] and Self-Organizing Map (SOM) [42] is used to group the subsequences based on the similarities between their features. Finally, *internal* and *external* anomalous sequences are detected. An internal anomalous sequence is a subsequence that is distantly positioned within a cluster. An external anomalous sequence is a subsequence that is positioned in the smallest cluster.

Distance-based clustering algorithms cannot derive relationships among multiple time series features in their clusters [43]. Moreover, these techniques only detect anomalous sequences without determining the records/attributes that are the major causes of invalidity in each sequence.

Autoencoder [2]. An autoencoder is a deep neural network that discovers constraints in the unlabeled input data. An autoencoder is composed of an *encoder* and a *decoder*. The encoder compresses the data from the input layer into a short representation, which is a non-linear combination of the input elements. The decoder decompresses this representation into a new representation that closely matches the original data. The network is trained to minimize the reconstruction error (RE), which is the average squared distance between the original data and its reconstruction [44].

The anomalous sequence detection techniques based on autoencoders (1) take a subsequence (i.e., a matrix of m records and d attributes) as input, (2) use an autoencoder to reconstruct the subsequence, (3) assign an invalidity score based on the reconstruction error to the subsequence, and (4) detect as anomalous those subsequences whose invalidity scores are greater than a threshold.

In an autoencoder network for anomalous sequence detection, the input (T_i) and output (T'_i) are fixed-size subsequences. T_i is the i^{th} subsequence that contains w records, w is the window size, and $X_{i,j} = [x^0_{i,j}, ..., x^{d-1}_{i,j}]$ is the j^{th} record in T_i with d attributes. The network output has the same dimensionality as the network input. The encoder investigates the dependencies from the input subsequence and produces a complex hidden context (i.e., d' encoded features). The decoder reconstructs the subsequence from the hidden context and returns a subsequence with shape $(d * w)$. The reconstruction error for this network is defined as follows [44]:

$$RE = \frac{1}{m} \sum_{i=0}^{m-1} (T'_i - T_i)^2 \tag{12}$$

where T_i and T'_i are the i^{th} network input and output and m is the total number of subsequences.

These techniques can learn complex non-linear associations among data attributes in the time series as a result of using a deep architecture with several layers of non-linearity. However, these techniques are not able to model temporal dependencies among the data records in an input subsequence.

LSTM-Autoencoder [2]. An LSTM-Autoencoder is an extension of an autoencoder for time-series data using an encoder-decoder LSTM architecture. As described in Section 3, an LSTM network uses internal memory cells to remember information across long input sequences. As a result, an LSTM-Autoencoder can capture the temporal dependencies among the input records by using LSTM networks as the layers of the autoencoder network.

Figure 5 shows the LSTM-Autoencoder architecture. The input and output are fixed-size time series matrices. $X_{i,j} = [x^0_{i,j}, ..., x^{d-1}_{i,j}]$ is the j^{th} record with d attributes, T_i is the i^{th} time series that contains w records, and w is the window size. The network output has the same dimensionality as the network input. The network is composed of two hidden layers that are LSTMs with d' units. The first LSTM layer functions as an encoder that investigates the dependencies from the input sequence and produces a complex hidden context (i.e., d' encoded time series features, where the value of d' depends on the underlying encoding used by the autoencoder). The second LSTM layer functions as a decoder that produces the output sequence, based on the learned complex context and the previous output state. The TimeDistributed layer is used to process the output from the LSTM hidden layer. This layer is a dense (fully-connected) wrapper layer that makes the network return a sequence with shape $(d * w)$. The reconstruction error for this network is defined as follows [44]:

$$RE = \frac{1}{m} \sum_{i=1}^{m} (T'_i - T_i)^2 \tag{13}$$

where T_i and T'_i are the i^{th} network input and output and m is the total number of subsequences.

These techniques can learn complex non-linear long-term associations among multiple data records and attributes as a result of using a deep network and the memory cells in their architecture. However, these associations are in the form of complex equations that are not human interpretable.

5 Summary

Table 4 summarizes different data quality test approaches for anomalous record and sequence detection. The blank cells mean "not applicable to". We

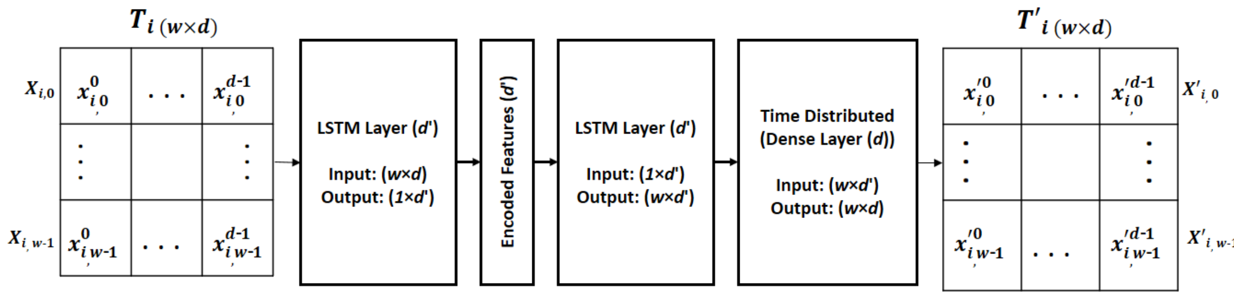

Figure 5: An LSTM-Autoencoder Network

have identified the following open problems in testing the time-series data.

Inapplicability to real-world time series. The stochastic time series analysis approaches only analyze univariate time-series data. Moreover, they assume that the time series is linear and follows a known statistical distribution. As a result, these classical time series modeling approaches do not apply to real-world multivariate time-series data with non-linear associations among data records and attributes. We propose to use a Machine Learning-based approach, which supports non-linear modeling, with no assumption about the statistical distribution of the data.

Unable to detect both anomaly types. Most of the existing stochastic (AR, MA, ARIMA, and SARIMA) and Machine learning-based approaches (MLP, SANN, LSTM, and SVM) can only detect anomalous records in time-series data. The approaches that detect anomalous sequences (clustering, autoencoder, and LSTM-Autoencoder) do not determine the anomalous records that are the major causes of invalidity in each sequence. We propose to assign a suspiciousness score to each record in an anomalous sequence to indicate the level of invalidity of the record in that sequence.

Unable to model long-term dependencies among data records. Most of the existing stochastic and Machine Learning-based approaches are unable to model long-term dependencies between data records. These approaches model the time series as a linear or non-linear function that associates current value of a time series to a small number of its past values. We propose to use an LSTM-based approach with memory cells in its structure that can model long-term dependencies between the data records.

Potential to generate false alarms. The unsupervised learning approaches, such as Autoencoder and LSTM-Autoencoder have the potential to learn incorrect constraints pertaining to the invalid data records and sequences and generate false alarms. False alarms can make the anomaly inspection overwhelming for the domain experts [45]. We propose to use an interactive learning-based LSTM-Autoencoder to minimize the false alarms.

Lacking a systematic approach to set input size. In the existing Anomalous sequence detection approaches, constraints are discovered within an input subsequence, the size of which is typically selected based on a fixed-sized window [46] or by using an exhaustive brute-force approach [47]. Since the window size can considerably affect the correctness of the discovered constraints, fixed-sized windows are not appropriate. Brute-force window-size tuning can be expensive. We propose a systematic autocorrelation-based windowing technique that automatically adjusts the input size based on how far the records are related to their past values.

Lacking explanation. The existing data quality test approaches for sequence data do not explain which constraints are violated by the anomalous sequences. Moreover, they do not determine the records or attributes that are major causes of invalidity of the anomalous sequences. We generate visualization diagrams of two types to describe the detected faults: (1) suspiciousness scores per attribute and (2) decision tree.

References

[1] G. Dong and J. Pei, *Sequence Data Mining*. Springer Science & Business Media, 2007, vol. 33.

[2] T. Kieu, B. Yang, and C. S. Jensen, "Outlier Detection for Multidimensional Time Series Using Deep Neural Networks," in *19th IEEE International Conference on Mobile Data Management (MDM)*, 2018, pp. 125–134.

[3] T. Guo, Z. Xu, X. Yao, H. Chen, K. Aberer, and K. Funaya, "Robust Online Time Series Predic-

Table 4: Data Quality Test Approaches for Sequence Data

Approach	**Time Series Type**	**Anomaly Type**	**Modeling Non-linearity**	**Modeling Seasonality**	**Modeling Long-term Dependencies**
AR	Univariate	Records			
MA	Univariate	Records			
ARIMA	Univariate	Records			
SARIMA	Univariate	Records		✓	
HW	Univariate	Records		✓	
MLP	Univariate	Records	✓		
SANN	Univariate	Records	✓	✓	
LSTM	Mulivariate	Records	✓	✓	✓
SVM	Mulivariate	Records	✓		
HTM	Multivariate	Records	✓	✓	✓
STL	Univariate	Records	✓	✓	
Clustering	Univariate	Sequences	✓	✓	
Autoencoder	Multivariate	Sequences	✓		
LSTM-Autoencoder	Multivariate	Sequences	✓	✓	✓

tion with Recurrent Neural Networks," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2016, pp. 816–825.

[4] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data," vol. 33, pp. 1409–1416, 2019.

[5] H. Lu, Y. Liu, Z. Fei, and C. Guan, "An Outlier Detection Algorithm Based on Cross-Correlation Analysis for Time Series Dataset," *IEEE Access*, vol. 6, pp. 53 593–53 610, 2018.

[6] "El Nino Data Set," https://archive.ics.uci.edu/ml/datasets/El+Nino (Accessed 2020-08-11).

[7] P. D. Talagala, R. J. Hyndman, K. Smith-Miles, S. Kandanaarachchi, and M. A. Munoz, "Anomaly Detection in Streaming Nonstationary Temporal Data," *Journal of Computational and Graphical Statistics*, pp. 1–21, 2019.

[8] Q. Wen, J. Gao, X. Song, L. Sun, H. Xu, and S. Zhu, "RobustSTL: A Robust Seasonal-Trend Decomposition Algorithm for Long Time Series," *CoRR*, vol. abs/1812.01767, 2018. [Online]. Available: http://arxiv.org/abs/1812.01767

[9] "TSfeatures from CRAN Library," https://cran.r-project.org/web/packages/tsfeatures/vignettes/tsfeatures.html (Accessed 2020-05-15).

[10] R. Adhikari and R. K. Agrawal, *An Introductory Study on Time Series Modeling and Forecasting.* LAP LAMBERT Academic Publishing, 2013.

[11] "Yahoo Server Traffic: A Benchmark Dataset for Time Series Anomaly Detection," Mar. 2020. [Online]. Available: https://yahooresearch.tumblr.com/post/114590420346/a-benchmark-dataset-for-time-series-anomaly

[12] "NASA Shuttle from UCI ML Repository," Mar. 2020. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle)

[13] Y. Chen and W. Wu, "Application of One-class Support Vector Machine to Quickly Identify Multivariate Anomalies from Geochemical Exploration Data," *Geochemistry: Exploration, Environment, Analysis*, vol. 17, no. 3, pp. 231–238, 2017.

[14] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," in *ACM SIGMOD International Conference on Management of Data.* Association for Computing Machinery, 2000, p. 93–104.

[15] Z. Cheng, C. Zou, and J. Dong, "Outlier Detection Using Isolation Forest and Local Outlier Factor," in *Conference on Research in Adaptive and Convergent Systems.* Association for Computing Machinery, 2019, p. 161–168.

[16] R. Thomas and J. Judith, "Voting-based ensemble of unsupervised outlier detectors," in *Advances in Communication Systems and Networks*. Springer, 2020, pp. 501–511.

[17] S. Shriram and E. Sivasankar, "Anomaly Detection on Shuttle data using Unsupervised Learning Techniques," in *IEEE International Conference on Computational Intelligence and Knowledge Economy*, 2019, pp. 221–225.

[18] P. M. Maçaira, A. M. T. Thomé, F. L. C. Oliveira, and A. L. C. Ferrer, "Time Series Analysis with Explanatory Variables: A Systematic Literature Review," *Environmental Modelling & Software*, vol. 107, pp. 199 – 209, 2018.

[19] "Autoregression Models for Time Series Forecasting with Python," https://machinelearningmastery.com/ Accessed (23-10-2019).

[20] G. A. F. Seber and A. J. Lee, *Linear Regression Analysis*, 2nd ed. Wiley, 2003.

[21] I. J. Myung, "Tutorial on Maximum Likelihood Estimation," *Journal of Mathematical Psychology*, vol. 47, no. 1, pp. 90–100, 2003.

[22] C. Kuster, Y. Rezgui, and M. Mourshed, "Electrical Load Forecasting Models: A Critical Systematic Review," *Sustainable Cities and Society*, vol. 35, pp. 257 – 270, 2017.

[23] A. Hatua, T. T. Nguyen, and A. H. Sung, "Information Diffusion on Twitter: Pattern Recognition and Prediction of Volume, Sentiment, and Influence," in *Proceedings of the Fourth IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*. Association for Computing Machinery, 2017, p. 157–167.

[24] D. F. Findley, D. P. Lytras, and T. S. McElroy, "Detecting Seasonality in Seasonally Adjusted Monthly Time Series," *Statistics*, vol. 3, 2017.

[25] N. Tomar, D. Patel, and A. Jain, "Air Quality Index Forecasting using Auto-regression Models," in *IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2020, pp. 1–5.

[26] Z. Hasani, B. Jakimovski, G. Velinov, and M. Kon-Popovska, "An Adaptive Anomaly Detection Algorithm for Periodic Data Streams", booktitle="Intelligent Data Engineering and Automated Learning," H. Yin, D. Camacho, P. Novais, and A. J. Tallón-Ballesteros, Eds. Springer International Publishing, 2018, pp. 385–397.

[27] S. Dhamodharavadhani and R. Rathipriya, "Region-Wise Rainfall Prediction Using MapReduce-Based Exponential Smoothing Techniques," in *Advances in Big Data and Cloud Computing*. Springer, 2019, pp. 229–239.

[28] O. Kramer, *Genetic Algorithm Essentials*. Springer, 2017, vol. 679.

[29] M. R. Mohammadi, S. A. Sadrossadat, M. G. Mortazavi, and B. Nouri, "A Brief Review Over Neural Network Modeling Techniques," in *IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, 2017, pp. 54–57.

[30] C. M. Bishop, *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

[31] Y. Yu, X. Si, C. Hu, and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[32] "Understanding LSTM Networks, Recurrent Neural Networks," https://colah.github.io/posts/2015-08-Understanding-LSTMs/ Accessed (21-10-2019).

[33] F. A. Gers, D. Eck, and J. Schmidhuber, "Applying LSTM to Time Series Predictable through Time-Window Approaches," in *Artificial Neural Networks — ICANN 2001*, G. Dorffner, H. Bischof, and K. Hornik, Eds. Springer, 2001, pp. 669–676.

[34] N. I. Sapankevych and R. Sankar, "Time Series Prediction Using Support Vector Machines: A Survey," *IEEE Computational Intelligence Magazine*, vol. 4, no. 2, pp. 24–38, May 2009.

[35] J. Wu, W. Zeng, and F. Yan, "Hierarchical Temporal Memory method for time-series-based anomaly detection," *Neurocomputing*, vol. 273, pp. 535 – 546, 2018.

[36] R. J. Hyndman, E. Wang, and N. Laptev, "Large-scale Unusual Time Series Detection," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, 2015-11, pp. 1616–1619.

[37] N. Laptev, S. Amizadeh, and I. Flint, "Generic and Scalable Framework for Automated Time-series Anomaly Detection," in *21th ACM*

SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2015, pp. 1939–1947.

[38] "How to Decompose Time Series Data into Trend and Seasonality," https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/ Accessed (25-10-2019).

[39] "Time series decomposition," http://course1.winona.edu/bdeppa/ Accessed (25-10-2019).

[40] I. Méndez-Jiménez and M. Cárdenas-Montes, "Time Series Decomposition for Improving the Forecasting Performance of Convolutional Neural Networks," in *Advances in Artificial Intelligence.* Springer International Publishing, 2018, pp. 87–97.

[41] G. Gan and M. Kwok-Po Ng, "k-means Clustering with Outlier Removal," *Pattern Recognition Letters*, vol. 90, pp. 8–14, 2017.

[42] T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[43] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications.* Society for Industrial and Applied Mathematics, 2007.

[44] C. Zhou and R. C. Paffenroth, "Anomaly Detection with Robust Deep Autoencoders," in *23rd ACM International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 665–674.

[45] B. N. Saha, N. Ray, and H. Zhang, "Snake Validation: A PCA-based Outlier Detection Method," *IEEE Signal Processing Letters*, vol. 16, no. 6, pp. 549–552, 2009.

[46] D. Park, Y. Hoshi, and C. C. Kemp, "A MultimodalAnomaly Detector for Robot-Assisted Feeding Using an LSTM-Based Variational Autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[47] B. Wang, Z. Wang, L. Liu, D. Liu, and X. Peng, "Data-driven Anomaly Detection for UAV Sensor Data Based on Deep Learning Prediction Model," in *2019 Prognostics and System Health Management Conference*, 2019, pp. 286–290.