Applied Ecology and Environmental Sciences, 2023, Vol. 11, No. 3, 79-90
Available online at http://pubs.sciepub.com/aees/11/3/2
Published by Science and Education Publishing
DOI:10.12691/aees-11-3-2

Advanced Spatio-Temporal Event Detection System for Groundwater Quality Based on Deep Learning

Divas Karimanzira[1,*], Linda Ritzau[1], Tobias Martin[2], Thilo Fischer[2]

[1]Department of Surface water and Maritime Systems, Fraunhofer IOSB, Am Vogelherd 90, 98693, Ilmenau
[2]TZW: DVGW-Technologiezentrum Wasser, Wasserwerkstraße 2, 01326 Dresden
*Corresponding author: divas.karimanzira@iosb-ast.fraunhofer.de

Received April 16, 2023; Revised June 01, 2023; Accepted June 20, 2023

Abstract It is very important in sensor networks for monitoring, e.g. groundwater quality, to detect sensor failures and anomalous spatio-temporal events such as spills and identify affected areas. Most of the method for anomaly detection do not truly utilize spatial and temporal information. In this paper a novel method based on deep learning (DL) is proposed which truly utilize multivariate spatio-temporal information in anomalous events detection. Anomalous events are quite rare, which makes it very challenging to obtain labeled anomaly datasets. It is therefore purposeful to use an unsupervised approach for sensor anomaly and event detection with labels only being used to set thresholds on prediction errors. Two method for an unsupervised anomaly detection in multivariate spatio-temporal data using deep learning are proposed in this paper. The first framework is composed of a Long Short Term Memory (LSTM) Encoder followed by an LSTM decoder and a LSTM predictor for temporal anomaly detection. In a further step, a Deep Neural Network (DNN) based classifier is used to classify the encoded and trained latent representation to their spatial corresponding to form a temporal and spatial anomaly detector. The second framework is based on a CNN encoder and a LSTM decoder to capture both spatial and temporal features. The encoder component can use either 3D convolutions or Multichannel CNN to capture complex spatial dependencies in each spatial neighborhood. The 3D tensor input for the encoder is formed by stacking the data from the nearest spatial neighbors of each data point. Both methods produce similar results for event detection, detecting different types of anomalies (point, context, etc.). After the training phase to learn the normal system behavior, both methods are capable of detecting anomalies that have never been seen before with a very good accuracy (values ranging between 88% and 96%). To validate the accuracy and efficiency of the DL-based methods, they were compared to a modified ST-DBSCAN algorithm. The results show the superiority of the DL-based methods.

Keywords: *Groundwater Nitrate Prediction, Anomaly detection, Deep learning, Auto encoder, Spatial and temporal anomalies*

Cite This Article: Divas Karimanzira, Linda Ritzau, Tobias Martin, and Thilo Fischer, "Advanced Spatio-Temporal Event Detection System for Groundwater Quality Based on Deep Learning." *Applied Ecology and Environmental Sciences*, vol. 11, no. 3 (2023): 79-90. doi: 10.12691/aees-11-3-2.

1. Introduction

Nitrate monitoring in groundwater is one component of a contamination warning system that continuously measures nitrate concentration. Generally, sensors set up in observation wells are used. Such a monitoring system is equipped with multiple sensors which generates a lot of data, as each sensor produces data continuously, often at one or two minute intervals. This makes it infeasible to have personnel manually and continuously monitor this data and the full benefit of the monitoring system cannot be realized [1]. The best solution is to perform real-time data analysis through automation and improve system availability and reliability. A sensor failure and event detection systems can be designed to monitor the groundwater nitrate measurements in real time and warn the operator in case of sensor failure or anomalous event and also provide spatial information about the region of event. Spatial event detection is an important and challenging problem. Unlike traditional event detection, the task of spatial event detection is to detect the spatial regions (e.g. Clusters of neighboring regions) where urgent events occur. It is very challenging to analyze such data for anomalies because a nitrate monitoring system is complex, and event or changes in nitrate concentration can be a result of different benign causes, e.g., changes in groundwater extraction, system operations, change in land use, and intentional or non-intentional spills [2]. Furthermore, there is no guarantee that the event detection system does not receive inaccurate data from the sensor caused by sensor malfunction or online data transmission issues, which automatically triggers alerts. Therefore, a fully automated sensor failure and event detection system is inevitable. As a result, automated analysis of the data inevitably produces invalid alerts. One criteria for the acceptance of such an event detection system is to avoid

false alarms. Thus, the ability to avoid false alarms and to detect anomalous events should be considered in the design of the event detection system.

In this paper, anomalous event detection refers to the problem of finding instances or patterns in data that deviate from normal behavior [3]. Hence, the notion of normal behavior has to be quantified objectively. The anomaly detection algorithm aims at identifying signal changes which may include abrupt transient shift (spikes and dips), abrupt distributional shift (bi-level changes), and gradual distributional shift - persistent anomalies denoted by slow positive and negative trends [3]. We can categories anomalies in data into three categories:

- Outliers - Short anomalous patterns that appear in a non-systematic way in the collected data, usually arising due to noise or faults, for example, due to communication errors
- Events/change - These patterns appear with a systematic and sudden change from previously known normal behavior. The duration of these patterns is usually longer than outliers. In environmental monitoring, extreme weather conditions are examples of events.
- Drifts - slow, unidirectional, long-term changes in data. This usually happens due to the onset of a fault in a sensor

Generally, algorithms fall into three key categories as shown in Figure 1:

(1) Unsupervised for classifying anomalies with training on unlabeled data and therefore most generally applicable. Several algorithms based on distance such as KNN, Means, statistical methods such as histogram-based outlier score, principal component analysis, classification-based such SVM and Isolation Forest [4], and angle-based outlier detection [5] can be applied.
(2) Supervised for classifying anomalies with training on labeled data. Unbalanced classification algorithms such as Support Vector Machines (SVM) or Artificial Neural Networks (ANN) can be used for this purpose and
(3) Hybrid, a mix of both schemes, which include semi-supervised learning, self-supervised learning etc. The basic idea of the popular algorithms such as Auto-Encoders ([6], [7], [8], Gaussian Mixture Models, Kernel Density Estimation used for semi-supervised learning is that a model of the normal class is learned and any deviations from that model can be said to be anomalies.

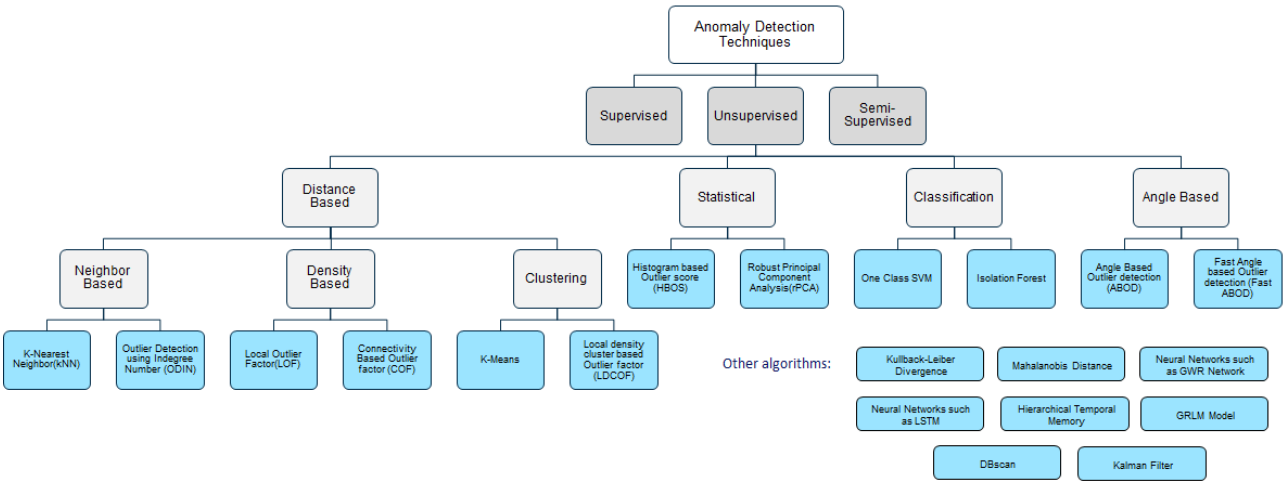


Figure 1. Categorization of anomaly detection algorithms

Using supervised learning for anomaly detection often implies cleaning the data in a pre-processing, extracting meaningful features and reducing the dimensionality to convert the data into usable data. These tasks are all very time-consuming and they may require domain knowledge, have knowledge about the characteristics of the data and the anomalies. This again is associated with other inherent issues such as the fact that there is often a thin line between normal and anomalous instances, or that the data might contain noise due to sensor malfunctioning or wrong measurements that may look similar to an anomalous event. Another common issue is to do with the fact that normal and anomalous data are often imbalanced [9]. It is obvious that in anomaly detection scenarios the available amount of observed normal data is huge compared to the very few moments of anomaly observations. Another issue, which should not be neglected is variability in the groundwater nitrate monitoring system. Sensors are added to, modified or removed from the system, position of sensors are changed, etc., which implies that the model has to continuously adapt to the current network set up. Unfortunately, conventionally, this requires model adaptation through retraining of the model from scratch. Training a model from scratch can consume a large amount of time and computational resources. We will apply transfer learning (TL) to transfer knowledge from the existing model to the new one without creating a new model from scratch.

Further challenges in the case of event detection in groundwater monitoring systems include, 1) Incorporating spatial and/or temporal information, 2) Integrating information from multiple features or data streams, 3) Distinguishing between multiple event types and 4) Computational complexity. There have been many studies on finding spatiotemporal anomalies. For example, in Birant and Kut [10] a three-step neighborhood-based

Spatio-Temporal (ST)-Outlier detection mechanism to identify the spatiotemporal outliers was proposed. In the first step, a DBSCAN algorithm is applied to identify the spatial neighborhoods. The temporal context of spatial-outlier objects are then checked by comparing them to temporal neighboring objects. In [11], a four-step approach was used to identify spatiotemporal outliers: classification (clustering), aggregation, comparison and verification. They aimed to address the semantic and dynamic properties of geographic phenomena for ST-outlier detection. All these methods first apply spatial (or non-temporal) context to find spatial outliers using a distance-based clustering. Then, existence of temporal outliers are detected by comparing the other spatial objects using temporal neighborhoods. Mostly DBSCAN, or locality based outlier detection algorithms such as LOF [12] are used to find neighborhoods. As such, they cannot detect collective anomalies. As mentioned before, distance-based methods are computationally expensive and not suitable for multivariate datasets [12].

Recently, deep learning-based methods are increasingly getting attention in anomaly detection, e.g. ([13], [14], [15], [16], [17], [18], and [19]). A review of deep learning methods for anomaly detection is given in [20]. In [13] an LSTM network-based encoder-decoder scheme for anomaly detection in univariate time series datasets was proposed. It learns to reconstruct normal time series data and uses reconstruction error to detect anomalies.

[8] Build an encoder composed of deep convolutional neural network and Restricted Boltzman Machine to extract features from videos. The extracted features are fed into an LSTM based prediction system to predict the next video frame in the learned feature space. Then, the difference between the prediction and actual observation in the feature space is used to detect anomalies. A semi-supervised deep auto encoder is used in [21] for outlier detection in multivariate clinical observation data from Electronic Health Records (EHR).

[19] Proposed a spatiotemporal architecture to detect anomalies in videos. The framework contains a spatial feature extractor and temporal encoder-decoder component. The spatial encoder component comprises two convolutional and two de-convolutional layers. They use a three-layer convolutional long short-term memory (LSTM) network as temporal encoder-decoder component.

Another anomaly detection in videos is in [17]. They use a deep spatio-temporal 3D convolutional auto encoder schema to detect falls in videos. The fall detection problem is formulated as a one class classification problem. A semi-supervised learning approach is used for learning. Furthermore, [22, 23] developed a Spatio-Temporal Auto-Encoder (STAE) to jointly exploit spatial and temporal evolution patterns for representation learning in video sequences.

A deep learning based unsupervised anomaly detection approach DeepAnT for time series data was presented in [16]. It is based on 1D deep convolutional neural network to predict univariate time series data. A prediction-based approach where a window of time series is used as a context and the next time stamp is predicted. The anomaly detector module uses the prediction error and a pre-defined threshold value to tag each data point as normal or abnormal.

In [24] a graph-based neural network and an LSTM are applied to deal with spatial and temporal information in multivariate time series.

In this paper, we focus on the problem of spatial and temporal event detection using deep learning. Two method for an unsupervised anomaly detection in multivariate spatio-temporal data are proposed. The first framework is composed of a Long Short Term Memory (LSTM) Encoder followed by an LSTM decoder and a LSTM predictor for temporal anomaly detection. In a further step, a Deep Neural Network (DNN) based classifier is used to classify the encoded and trained latent representation to their spatial corresponding to form a temporal and spatial anomaly detector. The LSTM predictor expands the reconstruction error of abnormalities, making it easier to distinguish abnormal events. At the same time, the reconstruction model based on the decoder enhances the ability to predict future time steps from regular events, which ensures robustness to noise. The second framework is based on a CNN encoder and a LSTM decoder to capture both spatial and temporal features. The encoder component can use either 3D convolutions or Multichannel CNN to capture complex spatial dependencies in each spatial neighborhood. The 3D tensor input for the encoder is formed by stacking the data from the nearest spatial neighbors of each data point. In both methods, a two-step approach is followed for anomaly detection. In a first step, the LSTM is employed to learn the normal time series patterns and to predict future values. This is followed by the anomaly detection in the second step, which is performed by computing anomaly scores from the prediction errors. Our goal is to identify if an event has occurred and characterize the event by pinpointing the affected subgroup of the data (e.g. spatial area, time duration).

The major contributions of this work include:

- Unsupervised anomaly detection which is designed specifically for non-image multivariate spatio-temporal data.
- A novel way of pre-processing the non-image multivariate spatio-temporal data by using the nearest spatial neighborhood, and
- Modification of the ST-DBSCAN method for multivariate non-spatial attributes.

2. Materials and Methods

2.1. Dataset

In this study, data collected from a regional groundwater quality monitoring network was used. The monitoring network consists of 162 observation wells as depicted in the monitoring network consists of 162 observation wells. The dataset covers a time period of 8 years from October, 2010 to February, 2019. However, the data had many missing features and values. So, only features with sufficient data history were selected. For each observation well, were temporal and spatial context attributes represented by the timestamp and geological position (latitude and longitude as well as the measuring depth), respectively. Besides that, the dataset contains ten groundwater quality records, each consisting of the features (sensor) values, observation well id in which the

measurements were taken. Contextual attributes include date, well_id, latitude, and longitude. In this study, latitude, longitude information are ignored and only the well information is used as spatial context variable. The sensors at each node included Nitrate, Calcium, Magnesium, Groundwater level, Redox potential, Sulphate, Temperature, Electro conductivity (EC) and pH

2.1. Methods

The goal is to detect any emerging events (e.g. nitrate spill), pinpoint the affected spatial area, and characterize the type of event. Rather than monitoring individual locations, we examine groups of locations and time periods. The procedure for anomaly detection follows in *three steps*. Data is collected from all heterogeneous sensors and preprocessed to generate feature values. Hereby, the sliding window technique is applied to build the multivariate spatial-temporal input data for the framework. By using the multistep overlapping subsequences from m-nearest spatial neighborhood of each data point, we build a 3-dimensional data matrix, which can represent the spatial and temporal dependency within the dataset. The important parameters of this algorithm are window size T and step size s. They should be chosen carefully based on the underlying dynamics of the dataset and the goal of anomaly detection problem. The length of each sub-sequence is equal to the window size. Using sliding window technique, for a long sequence with length L, the number of extracted subsequences can be given as: num.of subseq.= (L−T+1)/s, which gives the maximum number of subsequences we can possibly extract for a given T and s.

The collected data is used in the *second step* to train the anomaly detectors and detect potential anomalies. In the *third step*, the cause of the detected anomalies are estimated.

2.1.1. Data Processing

Data is collected from the groundwater water quality monitoring network in the form of spatio-temporal multivariate data of water quality parameters $c_{i,m}^{t}$ for spatial location s_i for each data stream (feature) D_m. There is a set of data records (observations) where each observation has a time-stamp, location information, and possibly other attributes.

Let $X = \left\{x^n\right\}_{n=1}^{N}$ denote a multivariate time series dataset composed of N subsequences of multivariate observations. Each subsequence has the same number of time steps, or window size, which is generated from a long multivariate time series data. Let each subsequence $x(n)$ has T time steps, and each observation at time step t, is a d dimensional vector which represents the number of observed features (number of univariate time series). The dataset X has dimensions of (N, d, T), where $x(n) \in R^{dxT}$. Each sample from multivariate time series dataset can be represented as

$$x^n = \begin{bmatrix} x_{11}^n & \cdots & x_{1T}^n \\ \vdots & \ddots & \vdots \\ x_{d1}^n & \cdots & x_{dT}^n \end{bmatrix} \quad (1)$$

Each multivariate data point x^n has a spatial attribute s^n attached to it and comes from a different spatial location. We denote multivariate spatio-temporal dataset as $D_{ST} = \left\{\left(X^{(i)}, s^{(i)}\right)\right\}_{i=1}^{S}$ which contains $N^{(i)}$ multivariate data points from S different spatial regions. The multivariate spatio-temporal data matrix X_{ST} can be represented as a three-dimensional matrix, as shown in Figure 2. It is considered as a group of many multivariate subsequences related to multiple spatial regions and representing observations from the same time window with the same timestamps. m represents the number of spatial neighborhoods to include in the anomaly detection process. In the case of the groundwater monitoring network, m defines the "range" at which observation wells are related to one another. If m is identified, it might be able to tell the graph distance at which observation wells tend to be come uncorrelated with one another.

This is computed using the (partial) spatial autocorrelation functions (SPACF). In a similar manner to the (partial) temporal autocorrelation function, the (partial) spatial autocorrelation function relates each observation to its (k)th order neighbors. In the spatial context, the (k)th order neighbors of observation (y_i) is the set of observations (y_j) that are first reached in (k) steps. This means that the graph distance between observation (y_j) and (y_i) is exactly (k):

$$y_{ik} : \min\left(\left\|y_j - y_i\right\|\right) = k \quad \forall j = 1, 2, \ldots, n \quad (2)$$

Thus, the (k)th order spatial autocorrelation function is:

$$\rho_k = cor\left(y, W^k y\right) \quad (3)$$

where (W^k) is the adjacency matrix for (k)-minimal neighbors. The (k)th-order partial spatial autocorrelation function is:

$$\rho_k^{\cdot} = cor(y, W^k y | W^{k-1} y, W^{k-2} y, \ldots, W^1 y) \quad (4)$$

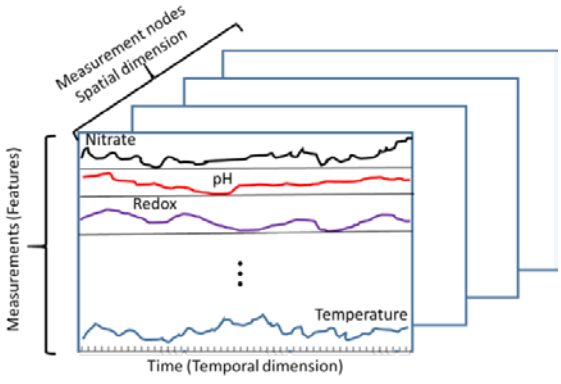
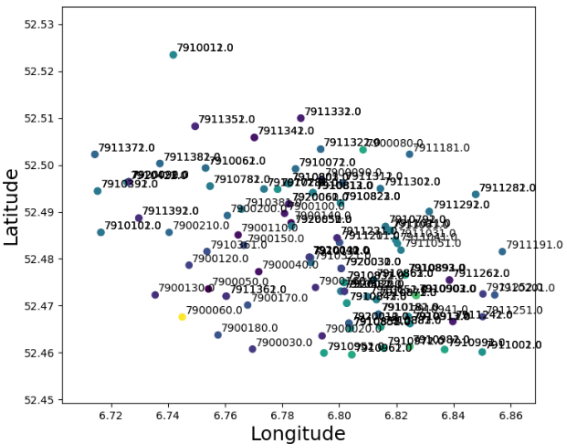


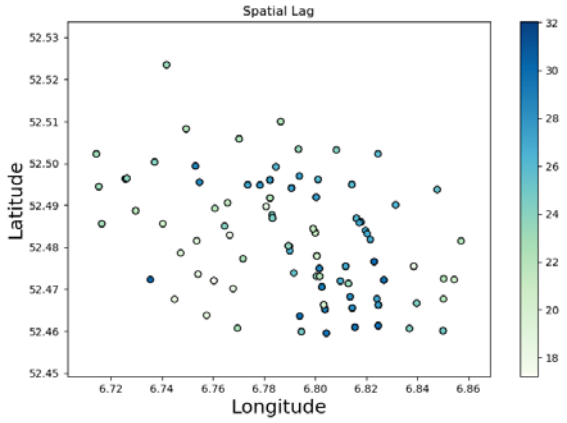
Figure 2. Illustration of the three-dimensional multivariate spatio-temporal data matrix structure

Several features were measured at quarter yearly rhythm and some values were missing. Therefore, the dataset was imputed by piecewise linear interpolation to monthly timesteps. The k-Balltree algorithm is applied to find all the k-neighbourhood wells for each observation well using the latitude and longitude information and the haversine formula [24].

To get a first impression if places that are close to each other are similar in regards to the spatial and non-spatial attributes (i.e., the relations between the observation wells), the spatial similarity and attribute similarity for nitrate concentration between every pair of observation wells were calculated using Moran II [25]. The results in Figure 3b show strong correlations between some of the neighboring observation wells.



a)



b)

Figure 3. a) Groundwater monitoring network consisting of 162 observation wells. b) Spatial and attribute similarities between observation wells in the model region based on nitrate concentration

2.1.2. Anomaly Detection Algorithms

Two frameworks based on end-to-end modelling as illustrated in Figure 4 and 5 are introduced in this paper. The auto encoder architecture is used to learn efficient data representation in an unsupervised manner [25]. There are three components to an auto encoder: an encoding (input) portion that compresses the data, in the process learns a representation (encoding) for the set of data, a component that handles the compressed data (size reduction), and a decoder (output) portion that reconstructs the learned representation as close as possible to the original input from the compressed data while minimizing the overall loss function. Long short-term memory (LSTM) is a neural network architecture capable of learning order dependencies in sequence prediction problems. A LSTM network is a type of recurrent neural network (RNN). The RNN mainly suffers from vanishing gradients. Gradients contain information, and over time, if the gradients vanish, then important localized information is lost. This is where LSTM is handful as it helps remember the cell states preserving the information. LSTM networks are good fit for classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series.

The first approach shown in Figure 4 consists of a LSTM Encoder to extract temporal features and deep neural network (DNN) classifier [26] to learn the spatial features to detect spatiotemporal anomalies. Again in this case the spatiotemporal anomalies are detected by the classifier in case it cannot successfully assign a correct spatial context label (location information) for the given sequence. In such a case it can be assumed that the encoded sequence was generated by a process that do not comply with temporal and spatial regularities of the given world. [28] Emphasized the superiority of a composite model consisting of a LSTM encoder-decoder and a LSTM future predictor to overcome the limitation of the individual models. Therefore, we will use a LSTM Auto encoder and LSTM Future Predictor which are trained in parallel to extract temporal context from the data as in [29]. The composite model include a shared encoder LSTM, and two LSTMs for decoding. One of the decoder is used to decode the representation generated by encoder into the input sequence, and the other one is used to decode the same representation to predict the future multivariate time series.

The LSTM Encoder produces latent representation of temporal data as output, which are then fed into the DNN classifier. This in turn is used to extract spatial features and classify input data to identify anomalous input sequences within given data. To be able to identify anomalous input sequences, the classifier is trained for predicting correct spatial location. Therefore, the unsupervised anomaly detection problem is formulated as multiclass classification problem by training the classifier to learn regions which each input sequence comes from. The aim of this step is to build a final classifier which is successful in assigning the correct location label to each input sequence and at the same time would be able to detect spatio-temporal anomalies which do not confirm with overall trend within each spatio-temporal neighborhoods.

The second approach illustrated in Figure 5 is made up of a spatial feature extractor followed by a convLSTM temporal encoder-decoder for learning the encoded spatial features. The hybrid autoencoder network is composed of a 3D convolutional neural network (CNN) based spatio-temporal encoder and a convolutional Long Short-Term Memory (ConvLSTM) network-based spatio-temporal decoder. It is designed to be trained in a truly unsupervised fashion for anomaly detection in non-image spatio-temporal datasets. We know that in a time series

data set, data points with two adjacent timestamps are likely to have a higher similarity than data points with more distant timestamps. It is also true for spatio-temporal datasets that neighboring regions may have some strongly positively correlated patterns, such as traffic jam, climate change, and human activity. The hybrid deep learning framework is able to exploit contextual features of neighboring regions for anomaly detection in the absence of labels for normal or abnormal

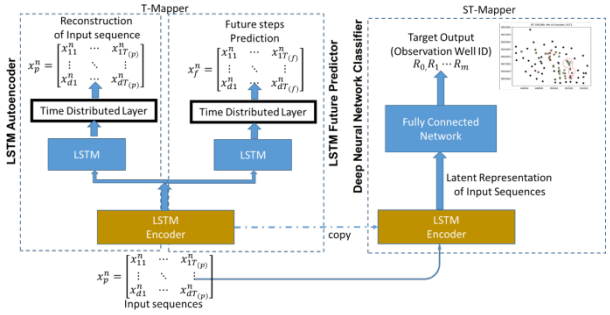


Figure 4. Framework 1: LSTM autoencoder and forecaster as basis for training the encoder which is then used for spatial and temporal anomaly detection by a deep neural network (DNN) classifier

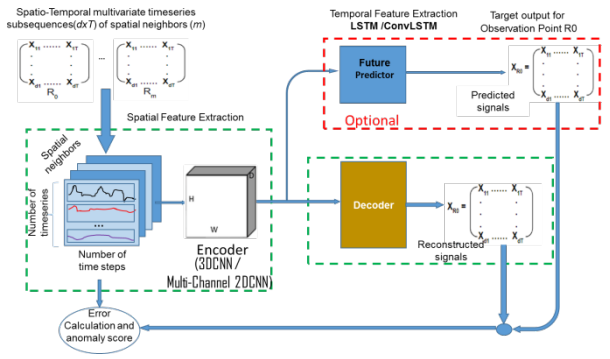


Figure 5. An Auto encoder made up of a CNN (3DCNN or Multichannel 2DCNN) decoder and a LSTM or ConvLSTM decoder and forecaster to decode spatial and temporal features for anomaly detection

Both proposed frameworks are composed of three steps: The first stage is the pre-processing of the multivariate spatio-temporal data so that the deep autoencoder network can exploit the spatial and temporal contexts jointly. The second stage is the data reconstruction stage, which is executed by a deep hybrid autoencoder network. The third stage is the anomaly detection stage, which is performed based on the reconstruction error.

In the first stage, the multivariate data from m nearest spatial neighbors are used to represent spatial dependency between different spatial regions as described in data preprocessing. A sliding window is applied to build the multivariate spatial-temporal input data in form of a 3D tensor for the 3DCNN encoder. By using the multistep overlapping subsequences from m nearest spatial neighborhood of each data point, a 3-dimensional tensor is built, which can represent the spatial and temporal dependency within the dataset. The important parameters of the spatial and temporal sliding window algorithm are window size T and step size s.

The second stage is for reconstruction and is composed of a 3DCNN for encoding the spatiotemporal features and a Convolutional Long Short-term Memory (ConvLSTM) network for decoding. 3D convolutional operations are applied on multivariate spatio-temporal data to better preserve the temporal features along with the spatial features. The input data are re-constructed as a 3-dimensional cuboid by stacking multivariate data frame. First, the models learn the "normal state" of the input data without separating the anomalous events and the non-anomalous situations. To have useful model skill and reliable threshold results, prediction intervals (PI) are derived to determine the model uncertainty. Prediction Intervals include all types of uncertainties. PI is defined based on p-quantiles as the interval from the lower ($PI_L = \frac{1-p}{2}$) to the upper limit $PI_u = \frac{1+p}{2}$) of the predictions, in which the true value is expected with a high probability (p). For the models**,** the uncertainty is presented as p=0.10 prediction interval from several bootstrapping runs. The upper and the lower bounds (PI_L) of the confidence band is computed as

$$PI_L = \bar{x} \pm 1.645\sqrt{\sigma^2 + MSE} \qquad (5)$$

$\bar{x}$ is the mean and σ^2 is the variance of the N-bootstrap runs, and MSE is the mean squared error of the fitted models. The bootstrapping procedure follows two steps: 1) computing a population of statistics e.g., mean squared error and then 2) calculating the confidence interval. A population of statistics is created by running the Bayesian optimization for hyperparameter search 100 times. Each time a new model with different hyperparameters is found and its metrics (MSE and variance) calculated. In the second step, the confidence interval is calculated using the resulting statistics. Anomalous events are then detected by way of deviation from the normal state, i.e., the time when the model is outside the PIs.

2.1.3. Identification of the Causes of Anomalies

In the third step, it can be identified which variety of data is the cause of anomaly by calculating the contribution of each variety of data to the degree of anomaly and observing how each contribution changes.

In order to validate the accuracy and efficiency of the deep neural network based architectures for spatio-temporal anomaly detection, it is worthwhile to compare the clustering results from the proposed method to those from classical methods. According to the format of the available data, a density-based method (e.g., ST-DBSCAN [10]) was preferred over a space–time scanning method (e.g., SaTScan). Density-based methods have the advantage that they can take a large number of cases into account and are able to detect irregularly shaped clusters. In this respect, the only comparable method is the ST-DBSCAN. The Spatio-Temporal Density-Based Spatial Clustering of Applications with Noise (ST-DBSCAN) algorithm [10] is based on DBSCAN [30]. It can be used to cluster spatial–temporal data according to its non-spatial, spatial and temporal attributes. Second, DBSCAN cannot detect some noise points when clusters of different densities exist. ST-DBSCAN assigns to each cluster a density factor so that some noise points when clusters of different densities exist can be detected. Unlike the LOF

algorithm, it can score data points and identify clusters. Even though **ST-DBSCAN** is able to uncover clusters based on locational and temporal distances, it cannot reflect similarities and differences in the other observed variables during the clustering process. The ST-DBSCAN algorithm was modified to consider similarities and differences in observations during the clustering process. In the work of [10] on ST-DBSCAN, a method called *Retrieve_Neighbors(obj, Eps1, Eps2*) is used for finding the neighbors of obj. For use in this paper, the ST-DBSCAN algorithm was modified with a third threshold parameter Eps3, which is an array of values of all features as follows:

```
Retrieve_Neighbors(obj, Eps1, Eps2, Eps3=[]):
Neighbors = []
For i=1 to |D|:
If dist1(obj, oi) <= Eps1 &
  dist2(obj, oi) <= Eps2 &
  dist3(obj, oi) >= Eps3:
  Neighbors += oi
Return Neighbors
```

2.1.4. Hyperparameter Search

The final structure and hyperparameters of the CNNs are obtained using Bayesian optimization (BayesOpt) [32]. Convolutional neural networks must have their architecture specified in order to be trained. Options for the training process include learning rate, window size, and L2 regularisation strength. It can be highly challenging and time-consuming to choose and tune hyperparameters, especially the combination of the CNN and the LSTM models requires a number of parameters to be tuned. For the selection of the hyperparameters of this neural network configurations be made a bit more informed, a good approach is Bayesian optimization [32]. The Bayesian optiumization algorithm pick a combination of hyperparameter values (our belief) and train the machine learning model with it. It then get the evidence (i.e. score of the model).The belief is updated so that it can lead to model improvement.

The process is terminated when a stopping criteria is met. The advantage of Bayesian optimization is that it can be used to optimize non-differentiable, discontinuous, and time-consuming functions. Besides the specification of the neural network architecture and deciding the options of the training algorithm, Bayesian optimization is also used to select the most important predictor variables.

3. Experiments

The models were implemented in the Python 3.8 environment with Keras framework using the Tensorflow backend. To have comparable results between the three methods, possible common parameters were determined in the same way. For example, the DL method based on the 3DCNN encoder requires the number of spatial neighbors, which is comparable to the minPts parameter required by the modified ST-DBSCAN. Furthermore, the DL method based on the LSTM Auto encoder and an LSTM Future Predictor requires the target clusters for the spatial classification. These cluster correspond to the one produced by the modified ST-DBSCAN algorithm.

Bayesian optimization is also used for tuning and selection of the hyperparameters of the CNN and the LSTM models. 10-fold cross-validation is used during the Bayesian optimization, where the data is randomly partitioned into 10 subsets. The model fitted to the remaining 9 subsets is then validated using each subset in turn. In this way, models with better generalizability could be established. Several runs per model could be executed to produce an ensemble of models (every run produces a different parameter combination). For each model type, the optimization parameters included the *window size*, *input features*, *batch size*, *learning rate*, *number of nodes* in the layers, and the *number of layers*.

In the composite framework the LSTM Auto encoder and an LSTM Future Predictor were trained in parallel with a shared encoder component. The LSTM Auto encoder, had one hidden layer with 100 neurons and the LSTM future prediction model had two hidden layers each with 200 and 100 hidden neurons all with Rectified Linear Unit (ReLU) activation functions, respectively. For the DNN Classifier, the number of inputs equalled the number of the outputs of LSTM Encoder. The output layer of the DNN classifier with a dimension determined by the number of clusters of observation wells, used a softmax function for categorical multi-class classification. Backpropagation algorithm was used for training the networks and the ADAM optimizer algorithm was used to optimize the mean squared error for the LSTM models and the categorical crossentropy was used as the loss function for multiclass classifier. 10 epochs with batch size of 64 were run for the LSTM models in training. The number of cluster of the observation wells required as labels was determined using the modified ST-DBSCAN. Like the classical DBSCAN method, the modified ST-DBSCAN identifies clusters with arbitrary shapes and determines the number of clusters according to spatial, temporal and feature similarities.

After the encoder has been trained, it was put in front of the DNN classifier as a temporal context extractor for the input data. The classifier is designed to predict the location label of each multivariate input subsequence in a supervised training settings. Hereby, the DNN classifier is trained using observation wells location information as labels to enforce the deep neural network classifier to learn spatial context.

The performance of the composite network was measured using precision and recall. Precision and recall are defined as the number of true positive results divided by the number of all positive results (true positives+ false positives), and the number of true positive results divided by the number of actual positive results, respectively. The accuracy is then given by the total number of true positive and true negative cases divided by all number of cases. Based on these definitions, the classifier gave a total accuracy of 99.8%. Table 2 shows the performance of the composite network for each class label.

For the second framework with the 3DCNN encoder, 2 CNN blocks, each of which has a 3D convolutional layer, followed by a 3D max-pooling layer with pool size of 2×2×2 and strides of 1×2×2 with padding, built the encoder network. The first block had 64 and the second block had 32 feature maps with padding and no striding. The kernel size was set to 3×3×5 for both convolutional

layers, as these values are found to produce the best result for the dataset. To allow the deep neural networks converge faster, an activation layer with Rectified Linear Unit (ReLU) non-linearity, ReLU(x)=max(x,0) was added to every convolutional layer.

Two ConvLSTM layers with the number of feature maps set to 32 and 64, respectively to preserve the symmetry of the autoencoder framework were composed together to form the decoder network. Their activation functions were set to hyperbolic tangent 2D convolution operations are applied over spatial and temporal dimensions using the kernel size of 3×2 and the stride of 1×1 with padding. Batch normalization (BN) is known to speed up the training of deep neural networks, therefore it is applied to each of the ConvLSTM layers. The final layer is a fully connected neural network. It reconstruct the target output. For compatibility reasons, the fully connected layer with ReLU activation function is a time distributed dense layer which applies the same fully connected operation to every time step. The dense layer had 10 nodes equal to the number of univariate time series (reconstruction features). Furthermore, the model is optimized using the Adam optimizer with learning rate of 0.0001, regularized by weight decay with L2 penalty multiplier of 1×10^{-4}) and dropout regularization for the two ConvLSTM layers was 0.25. The model is trained to minimize the following mean absolute error (MAE) loss function. Using Bayesian optimization, the final network structure of the autoencoder was found with the hyperparameters as shown in Table 1.

Table 1. Key hyperparameters

	Name of layers	Filters/Kernel size/Pool size	Parameter of layers
Encoder Network	Conv3d	64@3x3x5	Activation = ReLU
	MaxPooling3D	2x2x2	Strides=1x2x2
	Conv3d	32@3x3x5	Activation = ReLU
	MaxPooling3D	2x2x2	Strides=1x2x2
Decoder network	ConvLSTM2D	32@3x2	Strides = 1x1, activation = tanh, dropout = 0.25
	BatchNormalization		
	ConvLSTM2D	64@3x2	Strides = 1x1, activation = tanh, dropout = 0.25
	BatchNormalization		
	Reshape		
Fully connected Network	TimeDistributed Dense		Units = 512, Activation = ReLU
	TimeDistributed Dense		Units, 128,Activation = ReLU
	TimeDistributed Dense		Units, 9, Activation = ReLU

For the modified ST-DBSCAN, the required parameters Eps1, Eps2, Eps3, MinPts, and $\Delta\epsilon$ and were set to, Eps1 = 1, Eps2 = 0.25, MinPts = 15, $\Delta\epsilon$=*0.001*. The heuristics discussed in [31] were used to find the appropriate values of the parameters. By specifying the two thresholds one for spatial distance and another one for temporal distance separately, ST-DBSCAN can take temporal similarities into account more explicitly during the clustering process.

According to the type of application, whether its early detection of cascading effects of nitrate concentration spreading from one area to another or early detection of changes in the nitrate concentration in the individual wells, the data need to be divided accordingly, i.e. spatial or timely. In the first instance, the dataset is divided into observation wells for training, validation and testing. This enables the early detection of for example a nitrate spill and detection time can be used as an evaluation criteria for performance. In the second instance, it is purposeful to use the data from all the observation wells for training, validation and testing but dividing the timeseries temporary into training, validation and testing. Both instances were setup, whereby in the first instance the models are trained from 162 observation wells, which are divided into in a 10-fold manner for training and validation and in the second instance, the models are trained with a training dataset of 12376 spatio-temporal multivariate data points from 162 different observation wells obtained by a sliding window size of 7 months and sliding step size of 1 month. Groundwater processes are slow, hence the large window size to capture some changes. After training, a test set of 4368 data points was applied and the reconstruction and prediction errors calculated.

4. Results

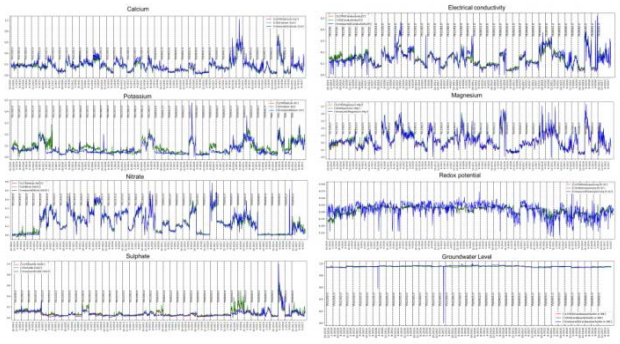


Figure 6. Model test results of the LSTM Composite network und the 3DCNN Auto encoder for the observation wells, identified by their IDs are given. The measured values for the different sensors (features) are plotted in blue

It is difficult to visualize the results because of the problems multidimensionality of (number of wells x number of features x timesteps) = 162x10x112. For the first instance, the results of the two deep learning models will be shown in the following. In Figures 6, the results of the reconstruction and prediction capability for the 42 test observation wells of the deep learning models can be seen for example features ('Groundwater level in NN', "Conductivity at 25°", 'pH-Wert', 'Water temperature', 'Magnesium (Mg)', 'Sulfate (SO4)', 'Calcium (Ca)', 'Potassium (K)', 'Nitrate (NO3)', 'Redox potential Eh (E)') for all the test observation wells identified by their ID e.g. 7912926. All values are min-max normalized into the

range of [0, 1]. The plots of the features shows a series of subsequence from the different observation wells, measured and reconstructed. The results show that the model trained very well with a validation loss of 0.0001 and accuracy of 0.995 for the LSTM Composite model and test loss of 0.00024 and test accuracy of 0.9993 for the 3DCNN Auto encoder model. The models are able to capture the structure of the real data and the resulting reconstruction error margins are quite good to enable the early detection of anomalous events for each test well.

From the results the anomaly/event detection procedure was run and the reconstruction errors compared with the threshold value determined by the PI evaluation. Even though labeled dta is not necessary for the algorithm, we created some labels manually for evaluating the models, which we classified into point, trend, fluctuations, non typical to the region and level change. 618 positions were labeled as anomalous in the dataset. There is no differentiation between the types of anomalies (Point, Context, Trends etc.)). Several quantitative information can be interpreted for the quality of the detection, including recall, precision and F1 score as shown in Table 1. From the 618 samples of anomaly, 504 were predicted correctly by the 3DCNN Auto encoder and 502 correctly by the LSTM composite network, which is an accuracy of the detectors of 82% and 81.2%, respectively. The type “Fluctuation” was confused in some cases as “non-typical” and “point” types. It has thus the highest misclassification with and accuracy in prediction of 62%.

Table 2. Quantitative measures for anomaly detection of the two algorithms

Type	Precision	Recall	F1-Score
Non typical	76.3	78.3	77.3
Fluctuations	67.5	61.3	64.2
point	82.6	86.3	84.4
trend	90.9	90.9	88.8
Level change	93.2	89.4	89.4

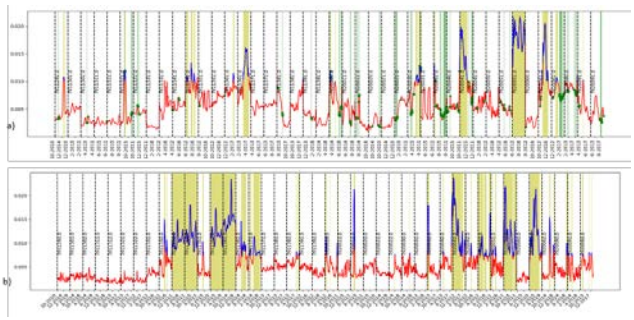


Figure 7. Anomaly detection results. Subsequences marked in blue and in the shaded areas indicate anomalous events a) LSTM Composite network und b) 3DCNN Auto encoder. Yellow marked areas are temporary anomalies and green marked represent spatial anomalies of the LSTM composite network

Figure 7 a and b show the results of the anomaly detection by the two models. Anomalous subsequences are marked in blue and the shaded areas. As described in the methodology section, the reconstruction errors are the mean square error for each subsequence of all the 10 features for each observation well. It can be seen that all suspected areas of irregularities in the spatial and temporal dimensions are discovered, which is but very dependent on the choice of the anomaly threshold value. Figure 7a shows the temporal anomalies of the LSTM autoencoder and future predictor (yellow). The spatial misclassifications which represent the spatial anomalies can be seen in Figure 7a marked in green after DNN classification. Both methods can detect all sorts of anomalies (Point, Context, Trends etc.) as can be seen in Figure 7 a and b. From the eight point anomalies recorded by the experts, the two methods managed to detect all of them and accuracy of 100 percent.

It can be seen from the above results that both methods are excellent in anomaly detection. They found similar areas for anomaly, whereby the 3DCNN encoder – based method combines the spatial and temporal anomalies in one. Closer look by an expert at the different anomalies can tell what type of anomaly exactly they are. For example Figure 8a show an extraction from anomaly diagram. Its shows spatial and context anomalies for observation wells 79110010 and 79110020. In this case the water temperatures of the two neighboring observation wells are quite different, which theoretically should not be. Regional dissimilarities are also shown in Figure 8b (collective anomalies). Closer look at the values of the feature shows that even though the LSTM could predict the individual timeseries for the test wells, the classifier could not classify the timeseries to the region (circled) because of the different behavior which is obvious if Figure 8b. Another example results is the detection of point anomalies in nitrate concentration as shown in Figure 8c. Level changes are also detected as events as shown in Figure 8d.

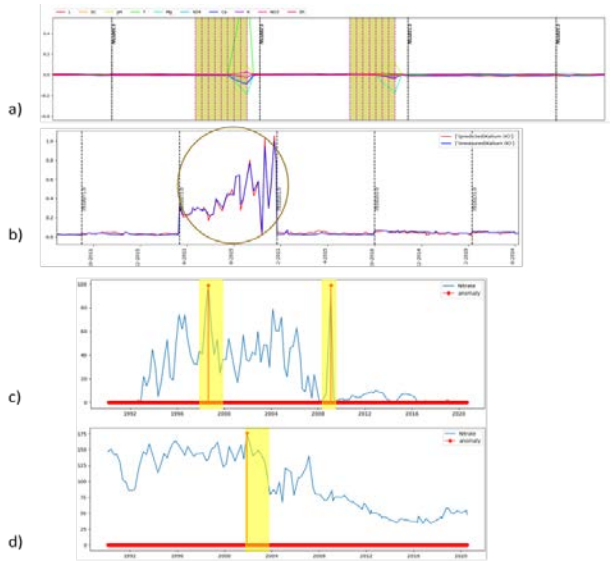


Figure 8. Spatial and context anomalies for neighboring wells a) water temperature and b) Potassium, c) point anomalies in nitrate concentration and d) level change in nitrate concetration. In b and c, the red line shows the beginning of the anomalies

As described in step 3 of the procedure, the causes of anomalies are identified as shown in Figure 9. The stacked bar chart shows the contributions of each variety (feature) to the anomalous event. The plot shows the date of the anomalous event, the contributions and the location (given by the ID, e.g. 79113210) of the event. From the contributions, the causers of the anomalies is identified as

Potassium (K). Probable location of the cause anomalous events can also be detected as in Figure 9 were the Potassium concentration in site 7910181 is maximum.

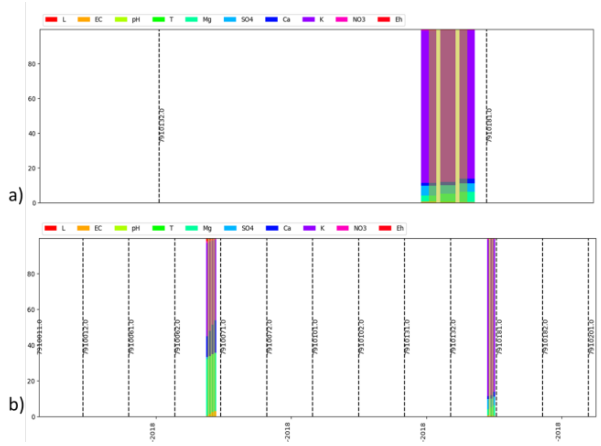


Figure 9. Anomaly detection results. a) Feature contribution to the anomalies in region 791013210 (i.e. anomaly causes) and c) Feature contribution to the anomalies for two neighboring regions 7910062 and 7910061

The two models were compared again based on the setup of the second instance. The results of the models are shown in the following Figure 100 and Figure 111. It can be seen based on the examples of Mg, Nitrate and Sulphate that both models can regionalize estimating the behavior of unknown wells from the ones used for training. Again both methods learn very well to reconstruct and predict the unknown test timeseries.

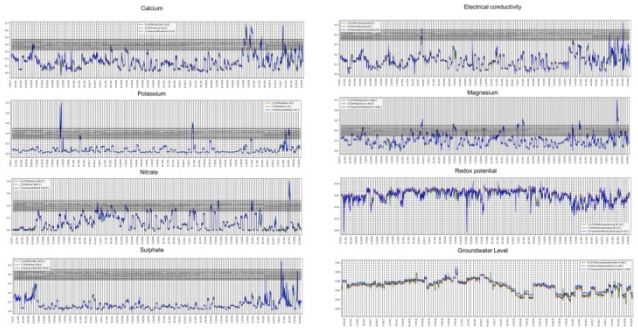


Figure 10. Model test results of the LSTM Composite network und the 3DCNN Auto encoder for the observation wells, identified by their IDs are given. The measured values for the different sensors (features) are plotted in blue

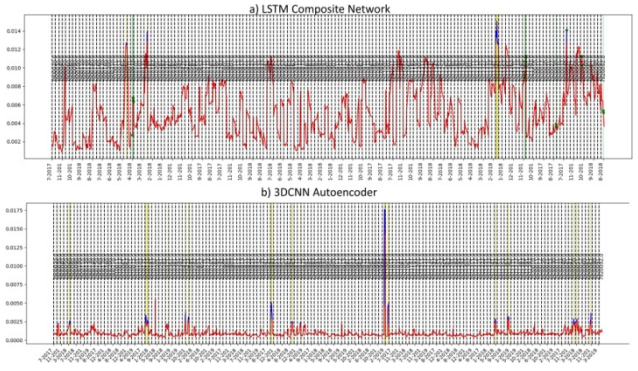


Figure 11. Anomaly detection results. Subsequences marked in blue and in the shaded areas indicate anomalous events. a) LSTM Composite network und b) 3DCNN Auto encoder

The modified ST-DBSCAN was run on the subsqeuences created in the process of training and testing the deep neural network architectures described previously for better comparison. The results of the modified ST-DBSCAN are shown in Figure 122 a and b.

In Figure 122 a, the observations identified as anomalies are plotted in black. These observations have scores values in spatial, temporal and attribute behavioural dimensions above the required threshold values. The non-spatial attribute included the time and the feature variables (('Groundwater level in NN', "Conductivity at 25°",'pH-Wert', 'Water temperature', 'Magnesium (Mg)', 'Sulfate (SO4)', 'Calcium (Ca)', 'Potassium (K)', 'Nitrate (NO3)', 'Redox potential Eh (E)')). Thus, the modified ST-DBSCAN can create groups with points that are spatially near each other and that has similar non-spatial attributes.

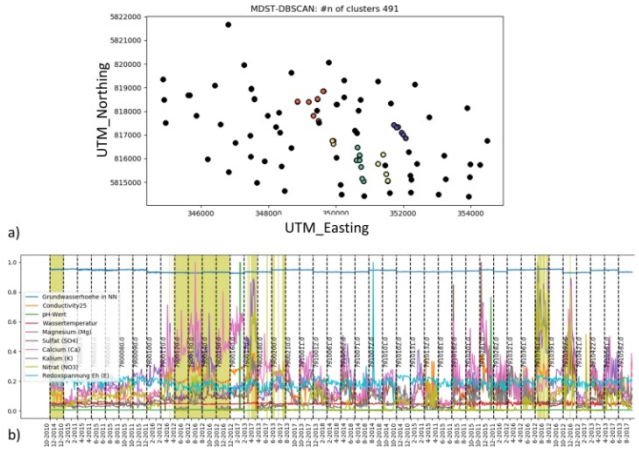


Figure 12. Anomaly detection results of the modified ST-DBSCAN. a) Spatio-temporal and feature based clusters obtained by running the modified ST-DBSCAN. b) Subsequences marked in blue and in the shaded areas indicate anomalous events

5. Discussions

Due to the complexity of the underlying groundwater system, it is quite challenging to detect events in the spatio-temporal and feature dimensions manually. All the three methods are true spatio-temporal anomaly detectors and they all produced acceptable results. The all got their advantages and disadvantages. The deep learning (DL) based methods need a lot of effort in the data preparation and training process. The first DL method based on the LSTM autoencoder, LSTM future predictor and DNN Classifier learns the underlying system very well with a validation mean squared error of 0.0001 and a the classification accuracy of the timeseries subsequences to their corresponding observation wells of 99%.This method determines the spatio-temporal anomalies in two stages, which is associated with two sources of errors. The other inductive bias in this model is the assumption that the same physics apply to all input sequences irrespective of which spatial neighborhood it comes from. The second DL method based on the 3DCNN encoder and ConvLSTM decoder explores the spatio-temporal anomalies comprehensively and produces the best result of the three methods with a validation mean squared error of 0.0001 and find all events as far as the human can

charge. Two parameters determine the accuracy of anomaly detection of the DL-based methods. These are the prediction accuracy and the anomaly threshold. Therefore, one of the major challenges of the DL-based methods is setting up these two parameters. In the experiments, the anomaly threshold was set to sigma six of the training error.

The modified ST-DBSCAN is a multidimensional spatiotemporal clustering method with its implementation based on the traditionally reliable ST-DBSCAN algorithm. It is less complex than the DL-based methods. Besides the threshold for the spatial and temporal dimension, it requires many more thresholds depending on the number of feature and finding the appropriate thresholds becomes more challenging. In literature, some efforts have been made to automate the parameter selection process in DBSCAN. These techniques such as the KNN distance plot are easily extendable to the modified ST-DBSCAN. Besides the thresholds, the clustering result of the modified ST-DBSCAN algorithm depends on which point is selected first in the clustering process. This is another important fact, which needs to be addressed in a future study.

6. Sensitivity Analysis

In further experiments with the DL-based models, a trade-off between prediction accuracy of the models and the results of anomaly detection was experienced. It showed that a DL-based model trained to minimize the prediction mean squared error was not always the best for anomaly detection. By optimizing the model for prediction, a kind of overfitting occurs which cannot be prevented by early stopping as the MSE continues to decrease on both training and validation dataset. It result in the model trying to fit even the anomalous observations to lower the prediction error. This is an unavoidable phenomenon, because the DL-based model training is unsupervised with respect to anomalies (i.e., model has no target or feedback to guard against fitting the anomalies. To solve this problem, the model was tuned manually to detect anomalies on the validation set. However, this comes at the cost of an increase in the prediction mean squared error value. In the manual tuning, one or more of the key parameters (lookback value, the number of LSTM units in the LSTM layers, or the number of LSTM layers) need to be reduced.

As described in the methodology section, the modified ST-DBSCAN requires several parameters: Eps1, Eps2, MinPts, and Delta-Epson. The values of the threshold are by the problem to solve (e.g. physical distance). In this paper, a k-distance graph was plotted and the distance threshold taken from the value of the elbow of the plot. This method is described in [31]. The average distance between each point and its k nearest neighbors is calculated, where k = the MinPts which is set by the user. The average k-distances are then plotted in ascending order on a k-distance graph and the optimal value for ε is take from the point of maximum curvature (i.e. where the graph has the steepest slope). In the plot in Figure 133 for the groundwater observation wells, there is a strong bend at 500m. So for k/minpts = 4, any point which have k^{th} neighbour distance greater than 500m will be considered noise/non-core point.

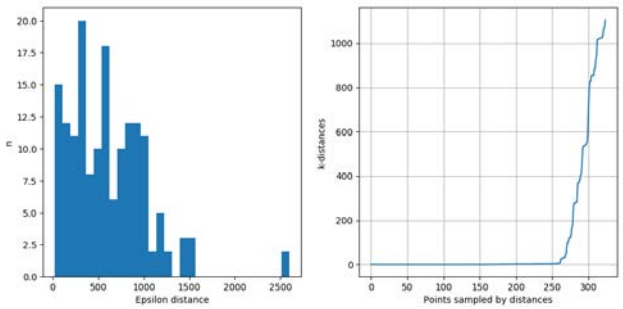


Figure 13. a) ε plot b) a k-distance graph

For the minPts, sensitivity experiments showed that the larger values yield significant clusters. Experiments also showed that too high values of the thresholds merges the clusters, putting the majority of the points in the same cluster. In case the threshold are chosen too small, a large number of points will not be clustered.

7. Conlusions

This paper focused on the detection of spatio-temporal anomalies in a multivariate timeseries dataset using a zero positive anomaly learning. As anomalies are rare and often come in different forms, the method requires no labeled dataset for the anomalous cases. Manually labeled data was only used for performance evaluation and not for training. In all distance/clustering-based algorithms, the biggest challenge is to combine the contextual features along the spatial and temporal dimensions in a meaningful way. Two methods are compared in this paper. The first architecture is consists of a CNN for spatial feature representation and a LSTM (ConvLSTM) autoencoder for learning temporal feature evolution. The second architecture is composed of a LSTM Encoder to extract temporal features and deep neural network (DNN) classifier to learn the spatial features to detect spatiotemporal anomalies. Spatiotemporal anomalies are detected by the classifier in case it cannot successfully assign a correct spatial context label (location information) for the given sequence. In such a case it can be assumed that the encoded sequence was generated by a process that do not comply with temporal and spatial regularities of the given world. The composite model was able to persistently detect spatio-temporal anomaly sequences well beyond the LSTM based prediction models. To further get improvements on spatial classifier, the model was extended by applying convolutional neural network base spatial context extractor using finer grained neighborhood data. The hybrid autoencoder network is composed of a 3D convolutional neural network (CNN) based spatio-temporal encoder and a convolutional Long Short-Term Memory (ConvLSTM) network-based spatio-temporal decoder. It is designed to be trained in a truly unsupervised fashion for anomaly detection in non-image spatio-temporal datasets. The framework is able to exploit contextual features of neighboring regions for anomaly detection in the absence of labels for normal or abnormal. Finally the two models are compared to a modified ST-

DBSCAN algorithm. This method is quite sensitive to the tuning parameters, what makes it very difficult to handle. Besides the thresholds, the clustering result of the modified ST-DBSCAN algorithm depends on which point is selected first in the clustering process. This is another important fact, which needs to be addressed in the future.

Acknowledgements

We acknowledge all the colleagues who supported in preparing data for the NIMO Projekt and all colleagues, institutions, or agencies that aided the efforts of the authors.

References

[1] Lanzolla A, Spadavecchia M. Wireless Sensor Networks for Environmental Monitoring. *Sensors*. 2021; 21(4): 1172.

[2] Yu, G., Wang, J., Liu, L. The analysis of groundwater nitrate pollution and health risk assessment in rural areas of Yantai, China. BMC Public Health 20, 437 (2020).

[3] Kenneth, O (2020), Real-Time Anomaly Detection for Multivariate Data Stream}, https://kenluck2001.github.io/blog_post/realtime_anomaly_detection_for_multivariate_data_stream.

[4] Mensi, Antonella & Franzoni, Alessio & Tax, David & Bicego, Manuele. (2021). An Alternative Exploitation of Isolation Forests for Outlier Detection. Structural, Syntactic, and Statistical Pattern Recognition, pp.34-44, 10.1007/978-3-030-73973-7_4.

[5] Xiaojie, li & Lv, Jian & Cheng, Dongdong. (2015). Angle-Based Outlier Detection Algorithm with More Stable Relationships. Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Volume 1 pp.433-446.

[6] Borghesi, Andrea & Bartolini, Andrea & Lombardi, Michele & Milano, Michela & Benini, Luca. (2019). Anomaly Detection Using Autoencoders in High Performance Computing Systems. Proceedings of the AAAI Conference on Artificial Intelligence. 33. 9428-9433. 10.1609/aaai.v33i01.33019428.

[7] Qian, S. Ying and B. Wang, "Anomaly Detection in Distributed Systems via Variational Autoencoders," *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2020, pp. 2822-2829.

[8] A. Munawar, P. Vinayavekhin and G. De Magistris, "Spatio-temporal anomaly detection for industrial robots through prediction in unsupervised feature space", *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, pp. 1017-1025, Mar. 2017.

[9] Kong J., Kowalczyk W., Menzel S., Bäck T. (2020) Improving Imbalanced Classification by Anomaly Detection. In: Bäck T. et al. (eds) Parallel Problem Solving from Nature – PPSN XVI. PPSN 2020. Lecture Notes in Computer Science, vol 12269. Springer, Cham.

[10] Birant, D. and Kut, A. (2007). St-dbscan: An algorithm for clustering spatial–temporal data. Data & Knowledge Engineering, 60(1):208 – 221. Intelligent Data Mining.

[11] Cheng, T.; Li, Z. A Multiscale Approach for Spatio-temporal Outlier Detection. Trans. GIS. 2006, 10, 253–263.

[12] Aggarwal, C.C. Spatial Outlier Detection. In Outlier Analysis, 2nd ed.; Springer Nature: New York, NY, USA, 2017; pp. 345–367.

[13] Malhotra, P.; Ramakrishnan, A.; Anand, G.; Vig, L.; Agarwal, P.; Shro_, G. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. ICML 2016, Anomaly DetectionWorkshop. arXiv 2016, arXiv:1607.00148v2.

[14] H. Yang, B. Wang, S. Lin, D. Wipf, M. Guo and B. Guo, "Unsupervised extraction of video highlights via robust recurrent auto-encoders", *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, pp. 4633-4641, Dec. 2015.

[15] Estiri and Murphy, Y. S. Chong and Y. H. Tay, "Abnormal event detection in videos using spatiotemporal autoencoder", *Proc. Int. Symp. Neural Netw.*, pp. 189-196, 2017.

[16] M. Munir, S. A. Siddiqui, A. Dengel and S. Ahmed, "DeepAnT: A deep learning approach for unsupervised anomaly detection in time series", *IEEE Access*, vol. 7, pp. 1991-2005, 2019.

[17] Nogas,J. S. S. Khan and A. Mihailidis, "DeepFall: Non-invasive fall detection with deep spatio-temporal convolutional autoencoders", *J. Healthcare Inform. Res.*, vol. 4, pp. 50-70, Mar. 2020.

[18] Y. Karadayi, M. N. Aydin and A. S. Öğrencí, "Unsupervised Anomaly Detection in Multivariate Spatio-Temporal Data Using Deep Learning: Early Detection of COVID-19 Outbreak in Italy," in *IEEE Access*, vol. 8, pp. 164155-164177, 2020.

[19] Chong and Tay, 2017.

[20] Pang, G., Shen, C., Cao, L. & Hengel, A. V. D. Deep learning for anomaly detection: A review. *ACM Comput. Surveys (CSUR)* 54, 1–38 (2021).

[21] H. Estiri and S. N. Murphy, "Semi-supervised encoding for outlier detection in clinical observation data", *Comput. Methods Programs Biomed.*, vol. 181, Nov. 2019.

[22] Lv, Hui & Cui, Zhen & Wang, Biao & Yang, Jian. (2022). Spatio-Temporal Relation Learning for Video Anomaly Detection. arXiv:2209.13116 [cs.CV].

[23] Liu T, Zhang C, Niu X, Wang L (2022) Spatio-temporal prediction and reconstruction network for video anomaly detection. PLoS ONE 17(5): e0265564.

[24] Tian, Z., Zhuo, M., Liu, L. *et al.* Anomaly detection using spatial and temporal information in multivariate time series. *Sci Rep* 13, 4400 (2023).

[25] P. V. Ingole and M. K. Nichat, "Landmark based shortest path detection by using Dijkestra algorithm and Haversine formula", *Int. J. Eng. Res. Appl.*, vol. 3, no. 3, pp. 162-165, 2013.

[26] *Li, Hongfei; Calder, Catherine A.; Cressie, Noel (2007). "Beyond Moran's I: Testing for Spatial Dependence Based on the Spatial Autoregressive Model". Geographical Analysis. 39 (4): 357–375.*

[27] D. D'Avino, D. Cozzolino, G. Poggi and L. Verdoliva, "Auto encoder with recurrent neural networks for video forgery detection", *Proc. IS&T Int. Symp. Electron. Imag. Media Watermarking Secur. Forensics*, pp. 92-99, 2017.

[28] P. Perera and V. M. Patel, "Learning deep features for one-class classification", *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5450-5463, Nov. 2019.

[29] M. Gupta, J. Gao, C. C. Aggarwal and J. Han, "Outlier Detection for Temporal Data: A Survey", IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 9, pp. 2250-2267, Sept. 2014.

[30] Srivastava, N., Mansimov, E., and Salakhutdinov, R. (2015): Unsupervised learning of video rep-resentations using lstms. International Conference on Machine Learning (ICML), 2015.

[31] Ester, M., H. P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, AAAI Press, pp. 226-231. 1996.

[32] Fernando Nogueira (2014), Bayesian Optimization: Open source constrained global optimization tool for Python, 2014, url = " https://github.com/fmfn/BayesianOptimization".

[33] Nadia Rahmah and Imas Sukaesih Sitanggang (2016), Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra *IOP Conf. Ser.: Earth and Environmental. Science.* 31 012012.