Studying properties of water data using manifold-aware anomaly detectors

1st Tino Paulsen
Machine Learning Group
Leuphana Universität Lüneburg
Germany
tino.paulsen@leuphana.de

2nd Ulf Brefeld
Machine Learning Group
Leuphana Universität Lüneburg
Germany
ulf.brefeld@leuphana.de

***Abstract*—Deep learning methods, especially the family of autoencoder architectures, exhibit state-of-the-art detection rates in anomaly detection tasks. Additionally, recent results show that learning latent spaces with deep architectures on Riemannian manifolds may further improve performances as well as related interpolation tasks. In this paper, we study the use of Riemannian manifolds with variational autoencoders (VAEs) for anomaly detection on data from water providers. Besides traditional embeddings in Euclidean space, we study embeddings in Poincaré discc, spheres, and Stiefel manifolds, where in general, the Poincaré disc is often preferred for data with hierarchical structures, embeddings in spheres suggests itself for cyclical structures and the Stiefel manifold is well suited for time-dependent data well. Data from water providers clearly meets all three criteria and we report on empirical results with the different manifolds as latent spaces and compare their detection performance to that of standard Euclidian embeddings.**

***Index Terms*—anomaly detection, Riemannian manifolds, non-Euclidean geometry**

I. Introduction

Water is possibly the most basic and, at the same time, precious resource in the world. Water is not only needed for our daily hydration, and thus survival, but also preserves and drives cultural and economic development; without water, there is no agriculture, no fish, no rivers for transportation of goods, and no industry.

Modern water providers deploy computer-monitored hardware to be able to control every aspect of their water treatment. Any intrusion from the outside into the system poses a real threat to the public as it may have severe impact on the quality of the drinking water. Water providers thus belong to the critical infrastructure of any country and need to be protected against threats accordingly. Since modern attackers adapt their strategies to finally launch a successful attack, a constant and never changing defence clearly falls short for lack of adaptivity. Instead, the system needs to be intelligent and learn from historic data to adapt quickly to present and future attacks. A simple strategy to detect intrusions in computer-based systems is called anomaly detection, where a model of normality is generated from normal system behaviour. Since attacks deviate from normality by definition, the model identifies this deviating behaviour as such and can automatically trigger counter measures.

Variational autoencoder (VAEs, [1]) have shown to perform well in a variety of anomaly detection problems ([2]–[4]). Autoencoders aim to detect and exploit manifold structures in the data and solve this unsupervised problem by a supervised setup: The idea is to embed input examples into a lower dimensional latent space and to reconstruct the original inputs from that code, so that a supervised loss function between original input and reconstruction can be optimized. Usually, autoencoders are implemented by neural networks where the components responsible for computing the embedding and reconstruction are called encoder and decoder, respectively.

By contrast, variational autoencoders interpret the latent space as parameters of a variational distribution. Thus, the decoder acts as a generator in a statistical sense, that is initialised with an input example and produces non-deterministic reconstructions given an input. VAEs are thus considered generative models.

To have VAEs (and autoencoders in general) perform well, manifold structures are exploited so that similar input examples are mapped to similar, neighbouring points in the latent space. Traditionally, Euclidean distance is used to compute this relatedness in latent space, however, a Euclidean assumption may not be appropriate for all data, particularly when those data exhibit manifold structures.

However, even for scenarios entirely set in Euclidean space, Arvanatidis et al. [5] showed that Euclidean distances may be warped which renders the measure consequentially unreliable. Inspired by this, different Riemannian manifolds have been studied as robust alternatives for learning latent spaces. Examples like hyperbolic geometry with instantiations as Poincaré balls [6] or hyperboloid spaces [7] have been studied, especially for data containing hierarchical structures because the negative curvature of the manifolds allows the space to expand further, the more distant it is from the origin, leaving enough room for clusters to be well separated.

Another interesting model using spherical geometry has been created by Davidson et al., [8], which is preferable when the data contains cyclical structures, like graph-structures with loops. The reason is that such loops can also exist in the latent space in spherical geometry. Figure 1 shows the three different geometries and their influence on the behaviour of straight lines. Hence, a Euclidean latent space should only assumed if the data contains a grid-like structure.

In this paper, we study whether VAEs on Riemannian manifolds allow for better detection rates on SWaT water data than traditional Euclidean assumptions. We focus on the three aforementioned manifolds, the Poincaré ball, a model of spherical geometry as well as a Stiefel manifold, a characterisation of the orthonormal bases of a vector space. All three manifolds provide characteristics that fit well to water data and SWat in particular due to an unusual mixture of continuous and discrete variables as well as its autoregressive nature. Our empirical results suggest that replacing the standard Euclidian way of computation by a Riemannian manifold generally leads to higher detection rates of attacks for most manifols. Particularly the Stiefel manifold seems to match the unique structure of SWaT very well and clearly outperforms its competitors.

The remainder of this paper is structured as follows. Section II summarises related work and Section III reviews the concepts of Riemannian manifolds. Section IV presents our learning methodologies. We report on the empirical setup in Section V and present the results in Section VI. Section VII concludes.

II. Related Work

Anomaly detection for water data has recently been drawing a lot of attention in the community and there exist a substantial body of related work that have been tried out on water data, particularly SWaT and WADI [9]. For brevity, we focus on only the subset of approaches that employ autencoders [2], [3], [10].

One of the best performances on SWaT stems from Xie et al. [4], who combine 1D convolutional and gated recurrent units to construct a network to learn the dependencies in the data so that statistical deviations in new data can be computed afterwards using the network output. Another very interesting method has been proposed by Fährmann et al. [10] who use lightweight LSTM-VAE architectures on SWaT and WADI to achieve computationally efficient models while preserving most of the predictive power of much larger architectures.

Zhou et al. [3] challenge the standard assumption that training data resembles a perfect model of normality by proposing a robust variant of autoencoders for anomaly detection. Based on the assumption that some anomalies may exist also in the training data, they utilise a decomposition of the training data using proximal gradients in fixed intervals during training to filter those anomalies out, and only then continue to assemble their model of normality.

Another paper that is concerned about the robustness of autoencoders has been presented by Arvanitidis et al. [5], who inspects the latent space of autoencoders. They observe that the distance function, which is assumed to be Euclidean, often is not really Euclidean anymore. Instead, the latent space is deformed through the training process and their contribution aims to measure this deformation of the latent space.

Instead of measuring the deformation, Mathieu et al. [6] propose to use a Riemannian manifold directly as latent space; in their case they deploy the Poincaré ball of hyperbolic geometry. In contrast to Arvanitidis et al. [5], the model is restricted to not change the nature of the latent space. This is achieved through switching from encoding positions to encoding velocities and utilising the velocities to move around the specified manifold.

Nagano et al. [7] propose to improve training on manifolds. By focusing on the hyperboloid model, they also choose a member of the hyperbolic geometry, but also generalize their results to other manifolds. Their hyperbolic models share the property that the latent space fits data with hierarchical structures well, and, as the space grows with distances, offers enough 'newly claimed' space for a growing density of clusters. This strategy is also possible in other geometries, such as hyperspheres [8], that constitute a good choice for latent spaces when the data meets cyclical structures.

An interesting work by Tran et al. [11] does not deal with VAEs but proposes to use Stiefel manifold together with variational Bayes. Their extension to the Stiefel manifold using a time series application performss well in comparison to the standard Euclidean model. Following this result, we utilise the Stiefel manifold for the autoregressive nature of water data.

In this paper, we study different manifolds for anomaly detection on SWaT, the 'standard' Euclidean one, two typical Riemannian manifolds, i.e. models for hyperbolic and spherical geometry and additionally the Stiefel manifold as a representative of the matrix manifold family. The Stiefel manifold can also be seen and used as a Riemannian manifold, but is contained in the more precise grouping of matrix manifolds, so we utilise the canonical version of it. These manifolds serve us as latent spaces for LSTM-VAEs, which are suited for the sequential nature of the data.

III. Specific Riemannian manifolds

We introduce necessary concepts from Riemannian geometry before turning to different specific Riemannian manifolds. For a more thorough treatment of Riemannian manifolds we confer to [12]. Every at least differentiable manifold $\mathcal{M}$ possesses for each point $\boldsymbol{z} \in \mathcal{M}$ a tangent space $\mathcal{T}_{\boldsymbol{z}}\mathcal{M}$ of same dimensionality of the manifold. A Riemannian manifold is a manifold $\mathcal{M}$ equipped with a Riemannian metric $\mathfrak{g}_{\boldsymbol{z}}$ which assigns every point $\boldsymbol{z}$ of the manifold a smoothly varying inner product:

$$\mathfrak{g}(\boldsymbol{z}) = \langle \cdot, \cdot \rangle_{\boldsymbol{z}} : \mathcal{T}_{\boldsymbol{z}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{z}}\mathcal{M} \to \mathbb{R}. \tag{1}$$

The Riemannian metric can be rewritten to be represented as a tensor which we denote by $G(\boldsymbol{z})$:

$$\forall \boldsymbol{u}, \boldsymbol{v} \in \mathcal{T}_{\boldsymbol{z}}\mathcal{M}, \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\boldsymbol{z}} = \mathfrak{g}(\boldsymbol{z})(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\intercal G(\boldsymbol{z})\boldsymbol{v}. \tag{2}$$

From the Riemannian metric one can construct a norm on $\mathcal{T}_{\boldsymbol{z}}\mathcal{M}$ given by $\| \cdot \|_{\boldsymbol{z}} = \sqrt{\langle \cdot, \cdot \rangle_{\boldsymbol{z}}}$. A measure can be defined using the metric tensor:

$$d\mathcal{M}(\boldsymbol{z}) = \sqrt{|G(\boldsymbol{z})|}d\boldsymbol{z} \tag{3}$$

with $d\boldsymbol{z}$ being the Lebesgue measure. Note that shortest paths between two points on manifolds are not necessarily straight lines between them, but should be though of curves since curvature deforms the space and hence, also shortest paths.

Geodesics are the generalisation of "straight" lines on Riemannian manifolds. In general, a parameterised curve on a

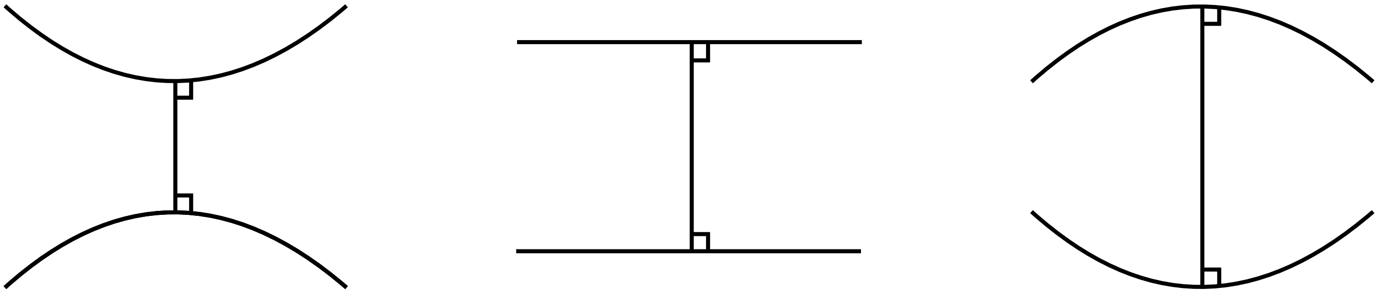

Fig. 1. A visualisation of three different geometries, from left to right: Hyperbolic geometry, Euclidean geometry and spherical geometry. The influence of curvature on the behaviour of ”straight” lines can be observed, in hyperbolic geometry the negative curvature drives the lines apart, in Euclidean the curvature of zero does not influence the lines, in spherical geometry the positive curvature drives the lines together.

Riemannian manifold is denoted by $\gamma : t \mapsto \gamma(t) \in \mathcal{M}$. Let γ be a connecting curve between two points $\boldsymbol{z}, \boldsymbol{y} \in \mathcal{M}$ of length

$$L(\gamma) = \int_0^1 \sqrt{\|\gamma'(t)\|_{\gamma(t)}}. \tag{4}$$

The shortest path between $\boldsymbol{z}$ and $\boldsymbol{y}$ is then given by $\gamma^* = \arg\min L(\gamma)$ with $\gamma(0) = \boldsymbol{z}, \gamma(1) = \boldsymbol{y}$ and called the *geodesic* from $\boldsymbol{z}$ to $\boldsymbol{y}$. This length of the shortest joining curve is often used to measure distances on Riemannian manifolds. For the specific manifolds used in this work the solutions to find geodesics are analytically known, however in the general case geodesics can also be found by optimising the associated differential equation [13].

Euclidean geometry, the standard workhorse of machine learning models, naturally possesses a description as Riemannian manifold. The corresponding metric tensor is, straight forwardly, the identity matrix $\mathbb{I}$. On the Euclidean manifold, the position described by a vector and the velocity to reach that position from the origin coincide, is given by its metric. This is not the case for other manifolds, resulting in the need for operators to move on the manifold. For unrolling a velocity $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{z}}\mathcal{M}$ from a tangent space, we utilise the exponential map $\exp_\mu(\boldsymbol{v})$ to unroll it at point μ on the manifold. The inverse logarithm map $\boldsymbol{v} = \log_\mu(\boldsymbol{z})$ takes two points $(\mu, \boldsymbol{z})$ and calculates the velocity $\boldsymbol{v}$ needed to move from μ to $\boldsymbol{z}$. Both operators follow the associated geodesic γ^* between μ and $\boldsymbol{z}$. Parallel transport is not restricted to be done only along geodesics, it is however possible and useful in the proposed models of this work.

We study three different manifolds for embedding purposes as latent spaces, namely the Poincaré ball, a model of hyperbolic geometry, the sphere manifold and the Stiefel manifold.

A. Poincaré ball

The Poincaré ball $\mathbb{B}^d_c$ is a model of hyperbolic geometry, which is especially useful for embedding data with hierarchical structures, as demonstrated by [6]. This is due to the property of the space expanding when influenced by negative curvature, resulting in larger distances the more points move away from the origin. As hierarchical data can be represented as trees, embeddings of those trees tend to get sprawled close together in Euclidean, but not in hyperbolic space as the growing space offers more area for the leaves. The metric tensor of the Poincaré ball is given by:

$$\mathfrak{g}^c_b(\boldsymbol{z}) = (\lambda^c_{\boldsymbol{z}})^2 \ \mathfrak{g}_e(\boldsymbol{z}), \quad \lambda^c_{\boldsymbol{z}} = \frac{2}{1 - c||\boldsymbol{z}||^2}, \tag{5}$$

where $\lambda^c_{\boldsymbol{z}}$ is the conformal factor and $\mathfrak{g}_e$ is the Euclidean metric tensor.

The exponential map is given by:

$$\exp^c_\mu(\boldsymbol{v}) = \mu \oplus (\tanh \sqrt{c}\frac{\lambda^c_{\boldsymbol{z}}||\boldsymbol{v}||}{2} \frac{\boldsymbol{v}}{\sqrt{c}||\boldsymbol{v}||}) \tag{6}$$

where $\oplus$ is denoting the Möbius addition [14]. The Möbius addition composes velocities, which is possible in spaces that are equipped with a gyrovector-structure, which the hyperbolic and spherical manifold do. The inverse operation for the exponential map, the logarithm map is given by:

$$\log^c_\mu(\boldsymbol{y}) = \frac{2}{\sqrt{c}\lambda^c_{\boldsymbol{z}}} \tanh^{-1}(\sqrt{c}|| - \boldsymbol{z} \oplus_c \boldsymbol{y}||) \frac{-\boldsymbol{z} \oplus_c \boldsymbol{y}}{|| - \boldsymbol{z} \oplus_c \boldsymbol{y}||} \tag{7}$$

Distance on the Poincaré ball is given by:

$$d^c(\boldsymbol{z}, \boldsymbol{y}) = \frac{1}{\sqrt{c}} \cosh^{-1}(1 + 2c\frac{||\boldsymbol{z} - \boldsymbol{y}||^2}{(1 - c||\boldsymbol{z}||^2)(1 - c||\boldsymbol{y}||^2)}) \tag{8}$$

B. Spherical geometry

In contrast to the negative-curvature hyperbolic space, spherical geometry is influenced by positive curvature. The spherical manifold is denoted by $\mathbb{S}^d_c$. This results in space becoming more dense, to the extend of forming spheres, thus allowing for loops. These loops make the space attractive for embedding data which exhibits cyclical structures, as shown by [8]. As points can now be connected by a path that loops around the sphere, geodesics have to choose the quicker path. From this follows an important detail of the exponential, the injectivity radius. This radius specifies the neighbourhood in which the exponential map is a diffeomorphism [12].

Spherical geometry can be realised in many different notations, as one can choose to either work intrinsically on the sphere or restrict a $n + 1$ dimensional euclidean space to the sphere. The Riemannian metric of a sphere of radius r as canonically given by [12]:

$$r^2\mathfrak{g}_s(x/r) = \frac{4r^4(dx_1^2 + \ldots + dx_n^2)}{(||x||^2 + r^2)^2} \tag{9}$$

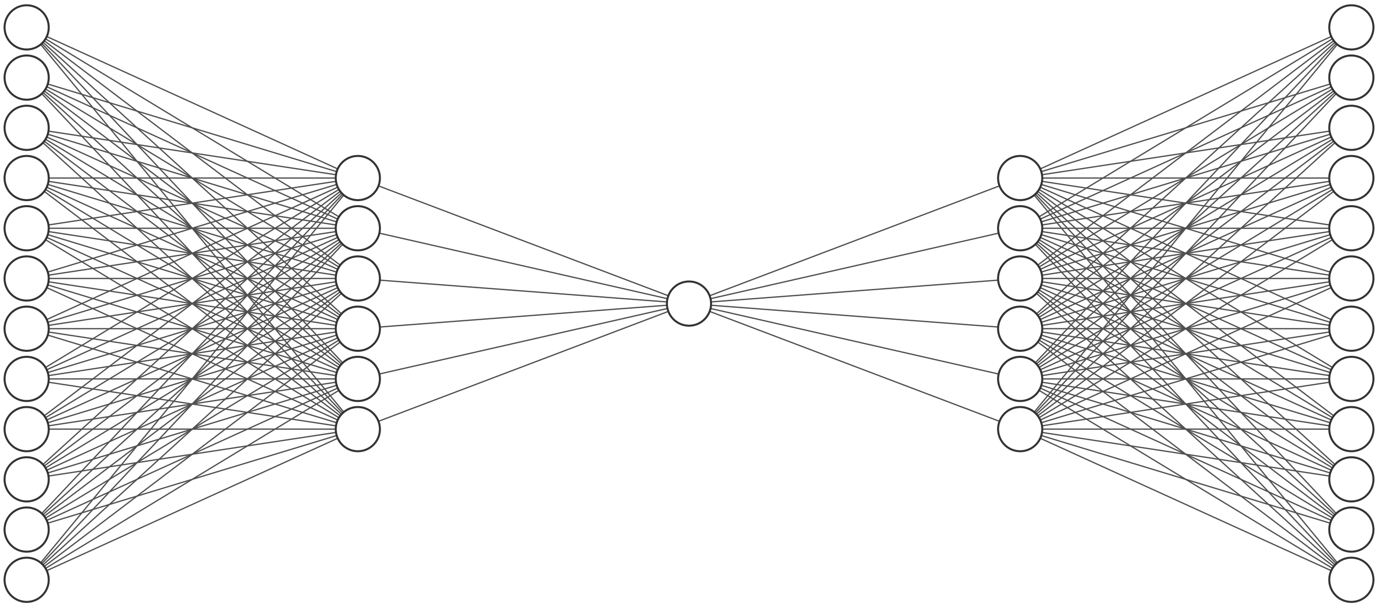

Fig. 2. A schematic of our model. The autoencoder architecture of reducing the dimensionality can be clearly seen, in our case there is a single entity in the middle of the model indicating the chosen manifold $\mathcal{M}$. The sampling in the latent space is on the manifold.

This is the metric of the stereographic projection model, which is used in this work. One advantage of this is that distances are now measured again on a flattened surface, so they are relatively simple to measure by just measuring the curve of the geodesic restricted in euclidean space. A possible way to calculate this is a standard polar representation of spheres and calculate the arc length therein.

C. Stiefel manifold

The Stiefel manifold is parameterising the orthonormal bases of a vector space. It is defined as:

$$\mathbb{V}_k(\mathbb{R}^n) = \{A \in \mathbb{R}^{n \times k} : A^\intercal A = I_k\} \tag{10}$$

with $n < k$. Another description of the Stiefel manifold is the set of orthonormal k-frames. One can define a Stiefel manifold over other spaces then $\mathbb{R}$, we only use this version of the manifold. This is a fundamentally different manifold from both the hyperbolic and the spherical one, following from the definition over vector spaces of $\mathbb{R}$. There is no immediately shared understanding of curvature with both aforementioned manifolds. It offers advantages over a normal vectors space as the Stiefel manifold can be more adaptive.

In comparison to the spherical manifold, the visualisation of 10 would also resemble a spherical condition like 9. The sphere manifold restricts the points to exist on the manifold, i.e. the sphere. Contrasting that, in a Stiefel manifold the condition ensures that the matrix characterises the orthonormal bases. If one visualises the norm, it appears similar, but it functions in a wholly different manner.

It should be noted that there exist versions of this manifold, one can not only choose a different space than $\mathbb{R}$, it is also possible to choose the inner product [15]. We choose the canonical one over the Euclidean one to accentuate the difference in the latent space.

IV. Methodology

A. Variational Autoencoder

In the scenario of unsupervised learning, variational autoencoder are a popular approach to learn a model of normality and utilise that for anomaly detection [1]. The key idea is to train a neural network to reproduce the input as target whilst reducing the model dimensionality typically in the middle of the network, creating a bottleneck. Given a dataset $\boldsymbol{X}^d$ of d dimensions, a variational autoencoder learns a mapping $\varphi : \boldsymbol{X}^d \mapsto \boldsymbol{Z}^k$ to a latent space $\boldsymbol{Z}^k$ called an encoder. This latent space has $k \ll d$, creating the aforementioned bottleneck. For each dimension of the latent space a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ is parameterised by approximating a mean μ and a variance σ using the decoder φ, resulting in a variational embedding of the data. From these distributions, $z^k \in \boldsymbol{Z}^k$ samples are drawn and decoded via an mapping $\psi : \boldsymbol{Z}^k \mapsto \boldsymbol{X}^d$. This decoder tries to reconstruct the original data $x^d \in \boldsymbol{X^d}$, we denote the reconstruction by $\hat{x}^d$.

As the target is known, we can quantify the deviation of the reconstruction using a loss, often the mean squared error is used:

$$\text{MSE} = \frac{1}{N} \sum_{1}^{N} (\hat{x}_i - x_i)^2 \tag{11}$$

The complete objective for optimisation is typically given by this reconstruction loss combined with a regularisation term for the distributions, in a variational autoencoder usually the Kullback-Leibler divergence is used:

$$\text{KL}(P, Q) = \sum_{x \in X} p(x) (\frac{p(x)}{q(x)}) \tag{12}$$

The prior is a standard normal Gaussian prior $\mathcal{N}(0,1)$. Combining the KL-divergence and the reconstruction loss results in the evidence lower bound (ELBO).

In order to utilise the VAE for anomaly detection the fact that, given a well-trained model, normal samples are reconstructed quite well is exploited. Anomalous examples are not reconstructed as well, resulting in a higher reconstruction error. A threshold is chosen which classifies the examples as normal or anomalous, typically a percentile of the reconstruction losses of the training or validation data.

B. LSTM

As the water data is of sequential nature, it needs to be handled accordingly. The appropiate neural network architecture for this was already introduced in 1997 [16], the long short-term memory cells. This architecture also addresses an important problem when training recurrent neural networks, the problem of vanishing gradients. When training a recurrent neural network with backpropagation, especially if the error is traced through sequences, the gradient of the error can vanish. This follows from being multiplicated by the learning rate, typically a small quantity, repeatedly.

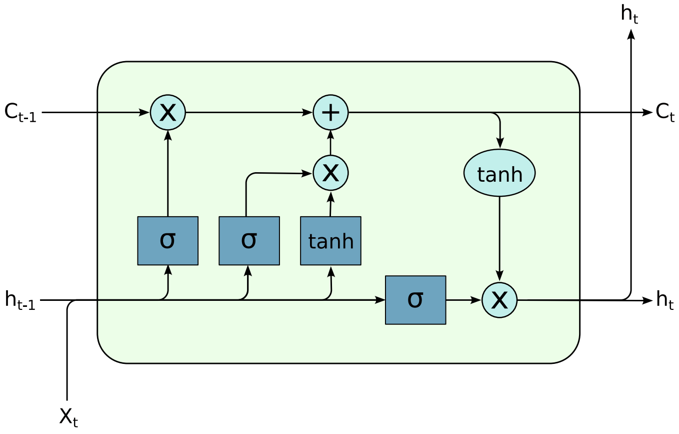

Fig. 3. An example of a LSTM cell. The σ here denotes the sigmoid activation function, x is the input, c_t is the cell state and h_t the hidden state. t denotes the time step of the optimisation.

The remedy devised by [16] is a cell architecture that learns how much information to remember and how much to forget while also leaving the gradient in its original form, thus effectively working against the vanishing gradient problem. In order to achieve this, different gates are used to modify the amount of information being passed on, as well as a hidden state and cell state that is modified between different activations of the cell. For more details we refer to [16].

LSTMs are the best-practice approach to handling sequences with temporal dependencies. Because of its sequential nature water data is often handled with LSTMs [10], [17].

C. VAEs on manifolds

In this section we describe how to utilise a Riemannian manifold as a latent space for a VAE. A visualisation of this process is given in figure 4. Usually a VAE learns a mean μ and a variance σ from the data into in order to parameterise a Gaussian distribution. σ is pushed through a *softplus* function to ensure positive variance.

On a manifold, we do not sample μ directly, but a velocity $\boldsymbol{v}_\mu$ that leads from the origin to a point used as mean on the manifold. We use the exponential map $\exp_0(\boldsymbol{v}_\mu)$ to reach μ on the manifold. The variance σ is parallel transported from the origin to μ. It is not directly assumed to be estimated at μ because then scale would be dependent on the location of μ, which would create a delay in optimisation which can be negated this way. Also, it eases the calculation of the Kullback-Leibler divergence because $p(x)$ and $q(x)$ are in the same tangent space at $\mathcal{T}_0\mathcal{M}$.

At μ, using the transported σ, a sample is drawn from a normal distribution. For this we use the wrapped normal distribution, i.e. we sample velocities $\boldsymbol{v}_z$ and use the exponential map $\exp_\mu(\boldsymbol{v_z})$ to unroll the velocity onto the manifold [7]. There are other generalisations of Gaussian distributions for specific manifolds, [18], [19] show a entropy-maximising generalisation. However such a generalisation is not readily available for all used manifolds. To ensure comparability during training and inference time a wrapped normal approach is chosen for all models.

When a sample $\boldsymbol{z}$ is generated, we use the logarithm map $\log_0(\boldsymbol{z})$ to calculate $\boldsymbol{v}_z$, which is given to the decoder. This scheme ensures that the embeddings generated by the encoder are in the chosen Riemannian manifold, which allows the models to profit from the different behaviour of space.

V. Experimental Setup

A. Dataset description

The dataset used is the SWaT dataset from [9]. It is a downscaled but fully functional testbed that mimics a real-world water purification plant. Contained in the dataset is not only the data from the actuators and sensors, but also from the network traffic. In this work we focus solely on the physical data, as our aim is to identify fitting manifolds to embed the data to, the network information would quite probably need another type of manifold, which exceeds the scope of this work.

The dataset contains a week of normal behaviour which result in 496,800 data points over 51 variables. These features span different continuous variables like flow meters, level and pressure sensors, and discrete variables like binary signals if pumps and valves are open or activated. Four additional days of operation were recorded, adding 449,919 data points during which time 36 attacks were executed. 28 of those attacks were on singular points, 8 were multiple attacks simultaneously. They manipulated sensor values and/or directly manipulated the functioning of the testbed. Duration and interval were mixed, for more details we refer to [9].

B. Data preprocessing

For the data processing and feature selection we follow [10], which in turn follow [2]. They propose a similarity test between the probability distributions of the training and the validation set to ensure that the training set and validation set are efficient when used in the usual training procedure, e.g. early stopping on the validation loss. Additionally, they use the test to show that the validation set is a good proxy for the test set, but don´t utilise this information, as the test set should not be known ahead of test time. Variables that have mismatching distributions are removed from the data sets, as training on these is assumed to hurt the model. That means specifically that we remove the following features: AIT201, AIT202, AIT203, P201, AIT401, AIT402, AIT501, AIT502, AIT503, AIT504, FIT503, FIT504, PIT501, PIT502, PIT503. Additionally the first 6 hours of the training data were removed, as the system needed that time to stabilise itself. The training data was split using an 80% - 20% split into training and validation data. All of training, validation and test set was normalised by removing the mean and scaling to unit variance of the training data.

As the data naturally exhibits temporal dependencies and the model is specific towards that fact, the data is prepared using sliding windows of size ω. Concretely, we slide a fixed-size window over the datasets and from that process generate sequences, which we use as samples. A common side effect of sliding windows is the loss of $\omega - 1$ samples, as just completely filled windows are tolerated. Once again following [10], we choose a window size of $\omega = 4$ and a stride of 1, so the created windows can overlap.

The reconstruction loss is aggregated over features and windows, resulting in a calibration of the model only to the exactness of windows or respectively the window size ω. In the test set we label a window as anomalous if it contains one sample which is marked as attack, thus creating more sensitive testing scenario.

C. Model specifications

We follow the experimental setup of [10] and use lightweight LSTM-VAEs, in our case we modify the latent space to be on a specific manifold. The set of manifolds is $\mathcal{M} = \{\mathbb{R}, \mathbb{B}, \mathbb{S}, \mathbb{V}\}$ and we denote the models with $\mathbb{R}$–VAE, $\mathbb{B}$–VAE, $\mathbb{S}$–VAE and $\mathbb{V}$–VAE for the latent space choices of Euclidean, Poincaré ball, spherical and Stiefel manifold respectively. Architectures of the neural networks used are given in the following table:

TABLE I
THE ARCHITECTURES OF THE USED VAES. THE BATCH SIZE IS NOT WRITTEN EXPLICITLY TO EASE READABILITY.

Layer	Output	Activation
LSTM	(4,32)	ReLU
Dense(μ)	16	Identity
Dense(σ)	16	Softplus
Sampling on $\mathcal{M}$	16	Identity
LSTM	(4,32)	ReLU
Dense	(4,36)	Identity

The used batch size is 128, the latent space has the dimensionality of 16 and the hidden dimension of 32. Optimisation is done using the Adam optimiser [20], with a learning rate of 0.05, and betas of $\beta = (0.7, 0.9)$, as the authors indicated this typically shows better performance when using `geoopt` [21], as it needs to retract gradients to the manifold. Also following the strong suggestion of [21], the used data type was `Float64`, as moving on manifold can be demanding on the numerical precision. All models were trained for 50 epochs, with selection of the best validating model enabled.

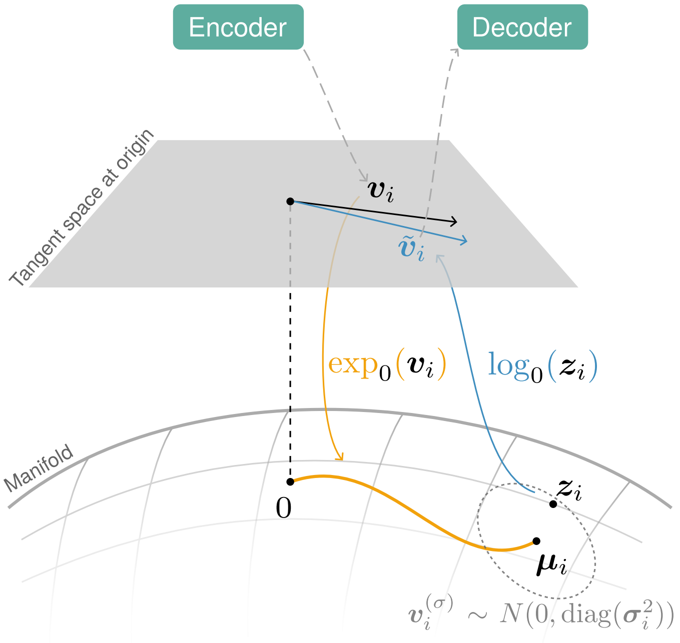
Encoder
Decoder
Tangent space at origin
$\boldsymbol{v}_i$
$\tilde{\boldsymbol{v}}_i$
$\exp_0(\boldsymbol{v}_i)$
$\log_0(\boldsymbol{z}_i)$
Manifold
0
$\boldsymbol{z}_i$
$\boldsymbol{\mu}_i$
$v_i^{(\sigma)} \sim N(0, \mathrm{diag}(\boldsymbol{\sigma}_i^2))$

Fig. 4. A visualisation of the sampling procedure in the latent space on a manifold. First, using a velocity $\boldsymbol{v}_i$ estimated by the encoder is pushed onto the manifold using an exponential map $\exp_0(\boldsymbol{v}_i$ to reach μ_i. At μ, using the parallel transported σ, a sample $\boldsymbol{z}_i$ is drawn from a Gaussian. For this $\boldsymbol{z}_i$ the logarithm map $\log_0(\boldsymbol{z}_i)$ is calculated and returns the velocity $\tilde{\boldsymbol{v}}_i$ that leads from the origin to $\boldsymbol{z}_i$, which is given to the decoder.

For implementation we use PyTorch, an automatic differentiation framework [22]. Regarding the manifolds, we use the unofficial implementation of [23] by [21], which the authors describe astutely as a "manifold-aware `pytorch.optim`". The package utilises the stereographic projection model for both the Poincaré ball and for the sphere manifold. They are provided a curvature of -1.0 and 1.0, respectively. The Stiefel manifold does not share a choice of curvature, as it is defined over $\mathbb{R}$, but it has a choice of inner product. We choose the canonical inner product over the Euclidean inner product.

D. Reconstruction scores

The loss already mentioned in 11 is slightly modified for handling sequences instead of singular data points as samples. The average over the window and over the features is taken, resulting in a single loss value for a window, usually termed reconstruction loss or reconstruction score. This reconstruction score is not only used for optimisation of the model, as it measures the deviation from the learned normality, we use it for anomaly detection. All reconstruction scores above a certain threshold are classified as anomaly. This threshold can be either found empirically or by some heuristic, we

TABLE II
Precision, Recall and F1 scores on SWaT

	SWaT		
Models	Precision	Recall	F1
$\mathbb{R}$–VAE	0.932	0.643	0.761
$\mathbb{B}^{-1.0}$–VAE	0.931	0.643	0.760
$\mathbb{S}^{1.0}$–VAE	0.932	0.643	0.761
$\mathbb{V}$–VAE	0.959	0.635	**0.764**

orient ourselves at [10] and use the 99th percentile of the reconstructed training data.

VI. Results

We evaluate the models on following metrics: precision, recall and F1-score. Given the true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), the metrics are given by:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

The empirical results are presented in table II. Our modification is the choice of latent space from the set of manifolds $\mathcal{M} = \{\mathbb{R}, \mathbb{B}, \mathbb{S}, \mathbb{V}\}$.

Using a Stiefel manifold as latent space in $\mathbb{V}$–VAE outperforms the other models. The increase in performance is highly significant with respect to a t-test on the reconstructions given by the models toward each of the other models, with test statistic ranging from 7 to 367. This could be due to the nature of a Stiefel manifold as characterisation of orthonormal bases of a vector space - thus improving the adaptability of the space as compared to a normal vector space. A Stiefel manifold latent space seems to be more suited for the structures of water data, which are naturally of mixed structures.

Utilising a Poincaré disc as latent space as in the $\mathbb{B}$–VAE does appear detrimental to the performance. As hyperbolic geometry is especially suitable for hierarchical structures in data, this seem to hint at the mismatch for water data. The embeddings of the model seem to challenge the proper reconstruction even more, as more distance must be learned and properly placed in the latent space.

The model using spherical geometry, $\mathbb{S}$–VAE, seems to perform equivalent to the benchmark model. Allowing the model to embed cycles does not seem to offer advantages to the model, however it also does not set the model back. As the SWaT data contains a week of normal behaviour data, the question remains open wether cycles of longer datasets like weekly or monthly rythms would be better embedded in this geometry.

The model using Euclidean geometry, $\mathbb{R}$–VAE, is the comparison model to [10] and our benchmark model. In comparison our models report higher precision and lower recall. A special interest is often shown not only for the general performance indicated by the F1-score, but also for the precision and recall as to recognise the tendencies of the model in specific directions of mistakes. As we do not include the network traffic data of SWaT it is more challenging to detect all anomalies, still this is a clear shift even of our benchmark model compared to [10].

Our main comparison is between our own models, but to ensure applicability for models we modify the architecture from [10], especially the architecture they call ”Lightweight-LSTM-VAE-S”. The authors of [10] report slightly higher numbers, even though the architecture is strictly congruent, this could follow from numerical details of the implementation. As the numbers are comparable, we assume the model to be valid and follow through with our inspection of the modifications.

It should be noted that there exist models which report a higher performance, like [4], which use a combination of statistical deviation analysis, 1D-convolutional neural networks and gated recurrent units. This is naturally more computation and storage demanding, which our work purposefully needs to avoid. As we operate with this limitation in mind, we orient and compare with [10], as their ”lighweight” models are applicable in wider circumstances.

VII. Conclusion

Water facilities are critical to protect and a crucial component of government infrastructure. Anomaly detection systems are a last-layer safety measure against malfunctions and attacks. For the virtual part of the cyber-physical system postulated by a water facility many tools exist like Intrusion-Detection-Systems (IDS). However for the physical part, there is need for smart, adaptive defence systems that need to be exceptionally robust, as false-positives can result in direct costs.

We studied the use of LSTM-VAEs as anomaly detectors together with different manifold assumptions, because recent research has shown an increase in robustness and performance when the latent space is fitting to the structure contained in the data [5]. Secondly, hierarchical structures are known to match assumptions in hyperbolic geometry, while for cyclical structures spherical geometries are preferred and grid-like data is well suited for the standard Euclidean geometry [6], [8]. We conducted a series of experiments to answer the question whether manifold assumptions are justified for the structures contained in water data.

The empirical results showed that the Stiefel manifold outperforms its competitors, particularly the standard Euclidean way of computation. This could indicate that the characterisation of orthonormal bases of vector spaces is especially matches charactersitic traits of water data. The lack of improvement of the spherical and Poincaré manifold seem to indicate a smaller prevalence of hierarchies and cyclical structures in the data. As water data often contains a difficult mixture of discrete and continuous variables as well as an autoregressive nature, the improved adaptability of a Stiefel manifold over a Euclidean vector space is plausible.

With this in mind and the challenges posed by water data, it may be possible to find a better suited manifold outside of the known, analytically solved ones, by directly exploiting traits of the mixed structure in the data. This manifold, however, yet remains to be found.

Acknowledgements

This work is funded in parts by the German Federal Ministry of Education and Research (BMBF) under the project 16KIS1205, and by the European Union under grant EFRE/85202549. The dataset was provided by iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design. We thank Samuel Fadel and Moritz Wohlstein for assistance regarding the images.

References

[1] D. P Kingma, M. Welling, *et al.*, "Auto-encoding variational bayes," in *Proceedings of the International Conference on Learning Representations (ICLR)*, vol. 1, 2014.

[2] M. Kravchik and A. Shabtai, "Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2179–2197, 2022. DOI: 10.1109/TDSC.2021.3050101.

[3] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 665–674.

[4] X. Xie, B. Wang, T. Wan, and W. Tang, "Multivariate abnormal detection for industrial control systems using 1d cnn and gru," *IEEE Access*, vol. 8, pp. 88 348–88 359, 2020. DOI: 10.1109/ACCESS.2020.2993335.

[5] G. Arvanitidis, L. K. Hansen, and S. Hauberg, "Latent space oddity: On the curvature of deep generative models," *arXiv preprint arXiv:1710.11379*, 2017.

[6] E. Mathieu, C. Le Lan, C. J. Maddison, R. Tomioka, and Y. W. Teh, "Continuous hierarchical representations with poincaré variational auto-encoders," *Advances in neural information processing systems*, vol. 32, 2019.

[7] Y. Nagano, S. Yamaguchi, Y. Fujita, and M. Koyama, "A wrapped normal distribution on hyperbolic space for gradient-based learning," in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 4693–4702.

[8] T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak, "Hyperspherical variational auto-encoders," *arXiv preprint arXiv:1804.00891*, 2018.

[9] J. Goh, S. Adepu, K. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," Oct. 2016.

[10] D. Fährmann, N. Damer, F. Kirchbuchner, and A. Kuijper, "Lightweight long short-term memory variational auto-encoder for multivariate time series anomaly detection in industrial control systems," *Sensors*, vol. 22, no. 8, 2022, ISSN: 1424-8220. DOI: 10.3390/s22082886. [Online]. Available: https://www.mdpi.com/1424-8220/22/8/2886.

[11] M.-N. Tran, D. H. Nguyen, and D. Nguyen, "Variational bayes on manifolds," *Statistics and Computing*, vol. 31, pp. 1–17, 2021.

[12] J. M. Lee, *Introduction to Riemannian Manifolds*, 2nd ed. Springer, 2018.

[13] M. P. Do Carmo and J. Flaherty Francis, *Riemannian geometry*. Springer, 1992, vol. 6.

[14] A. Ungar, *A Gyrovector Space Approach to Hyperbolic Geometry*. Morgan & Claypool Publishers, 2009.

[15] Y. Chikuse, *Statistics on special manifolds*. Springer Science & Business Media, 2003, vol. 174.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] K. Faber, M. Pietron, and D. Zurek, "Ensemble neuroevolution-based approach for multivariate time series anomaly detection," *Entropy*, vol. 23, no. 11, 2021, ISSN: 1099-4300. DOI: 10.3390/e23111466. [Online]. Available: https://www.mdpi.com/1099-4300/23/11/1466.

[18] X. Pennec, "Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements," *Journal of Mathematical Imaging and Vision*, vol. 25, pp. 127–154, 2006.

[19] S. Hauberg, "Directional statistics with the spherical normal distribution," in *2018 21st International Conference on Information Fusion (FUSION)*, IEEE, 2018, pp. 704–711.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] M. Kochurov, R. Karimov, and S. Kozlukov, *Geoopt: Riemannian optimization in pytorch*, 2020. arXiv: 2005.02819 [cs.CG].

[22] A. Paszke, S. Gross, S. Chintala, *et al.*, "Automatic differentiation in pytorch," 2017.

[23] G. Becigneul and O.-E. Ganea, "Riemannian adaptive optimization methods," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=r1eiqi09K7.