Using a VAE-SOM architecture for anomaly detection of flexible sensors in limb prosthesis

Zikai Zhu a,1, Peng Su b,1, Sean Zhong c, Jiayu Huang a, Suranjjan Ottikkutti b, Kaveh Nazem Tahmasebi b, Zhuo Zou a,*, Lirong Zheng a,*, DeJiu Chen b,*

a School of Information Science and Engineering, Fudan University, Shanghai, China
b Department of Engineering Design, the Royal Institute of Technology, Stockholm, Sweden
c Department of Computer and Information Science, Linköping University, Linköping, Sweden

A B S T R A C T

Flexible wearable sensor electronics, combined with advanced software functions, pave the way toward increasingly intelligent healthcare devices. One important application area is limb prosthesis, where printed flexible sensor solutions enable efficient monitoring and assessing of the actual intra-socket dynamic operation conditions in clinical and other more natural environments. However, the data collected by such sensors suffer from variations and errors, leading to difficulty in perceiving the actual operational conditions. This paper proposes a novel method for detecting anomalies in the data that are collected for measuring the intra-socket dynamic operation conditions by printed flexible wearable sensors. A discrete generative model based on Variational AutoEncoder (VAE) is used first to encode the collected multi-variant time-series data in terms of latent states. After that, a clustering method based on the Self-Organizing Map (SOM) is used to acquire discrete and interpretable representations of the VAE encoded latent states. An adaptive Markov chain is utilized to detect anomalies by quantifying state transitions and revealing temporal dependencies. The contributions of the proposed architecture conclude as follows: (1) Using the VAE-SOM hybrid model to regularize the continues data as discrete states, supporting interpreting the operational data to analytic models. (2) Employing adaptive Markov chains to generalize the transitions of these states, allowing to model the complex operational conditions. Compared with benchmark methods, our architecture is validated via two public datasets and achieves the best F1 scores. Moreover, we measure the run-time performance of this lightweight architecture. The results indicate that the proposed method performs low computational complexity, facilitating the applications on real-life productions.

1. Introduction

Limb prostheses offer a solution to help the amputees restore their ability to walk after amputations. The performance of such wearable system is strongly affected by the design of prosthetic socket, which constitutes the interface between the user stump (natural) and the prosthetic device (artificial). In general, a well-designed prosthesis should have a socket providing appropriate fitting, suitable load transmission, and effective operation stability and control. However, due to varying bio-mechanical conditions among amputee stumps, as well as many unforeseen dynamic effects (e.g. volume fluctuation and tissue evolvement) in operation, it is a challenging task to create a comfortable prosthetic socket. The prosthetic socket usually involves a set of virtual and physical services. These services cover integration in mechanical, electrical, and information engineering [1,2], providing simulated and physical information (e.g., simulated and sensory data via the test and verification) to optimize and upgrade the prosthetic sockets. For example, a typical design and test workflow integrated mechanical and intelligent services can summarize as follows [3]: First, bio-mechanical modeling and simulation of overall gait dynamics; Next, analysis and virtualization of possible pressure load conditions; Then, physical prototype testing with operational data collection; To this end, one critical step is to equip the prostheses with advanced perception system that helps monitor and assess the dynamic operational conditions within prosthetic sockets that are traditionally not directly observable. In particular, monitoring operational data plays a critical role in strengthening the design, development, and deployment of the

* Corresponding author.
E-mail addresses: zzh20@fudan.edu.cn (Z. Zhu), pensu@kth.se (P. Su), seazh870@student.liu.se (S. Zhong), jiayuhuang21@m.fudan.edu.cn (J. Huang), suranjjan@kth.se (S. Ottikkutti), kavent@kth.se (K.N. Tahmasebi), zhuo@fudan.edu.cn (Z. Zou), lrzheng@fudan.edu.cn (L. Zheng), chendj@kth.se (D. Chen).

Corresponding author.
E-mail addresses: zzh20@fudan.edu.cn (Z. Zhu), pensu@kth.se (P. Su), seazh870@student.liu.se (S. Zhong), jiayuhuang21@m.fudan.edu.cn (J. Huang), suranjjan@kth.se (S. Ottikkutti), kavent@kth.se (K.N. Tahmasebi), zhuo@fudan.edu.cn (Z. Zou), lrzheng@fudan.edu.cn (L. Zheng), chendj@kth.se (D. Chen).

1 These authors contributed equally to this work.
These results indicate that the architecture is wearable devices friendly.

The rest of this paper is organized as follows: Section 2 introduces the related concepts and studies on anomaly detection techniques. Section 3 formalizes the problems by splitting them into spatial and temporal perspectives. Section 4 applies the algorithm for a case study by using the collected data from a limb prosthesis socket test and a similar public dataset to evaluate its performance. Finally, the conclusion of the effectiveness of the proposed algorithm is given in Section 5.

2. Related work

Anomaly detection plays a critical role in realizing the self-awareness and self-calibration of the Cyber-Physics System (CPS), such as bio-devices, autonomous driving systems, and industrial robots. Conventional methods focus on detecting time-series anomalous data by using statistical analysis, such as Auto Regression Moving-Average (ARMA) [8]. The performance of such classical time-series analysis methods depends on the availability of a large amount of data with strong seasonality. Moreover, these conventional approaches are insufficient to cope with the multi-dimensional data due to the complexity of data features. To process such data, a feature extractor, such as wavelet transform (WT), Fourier transform, Principal Component Analysis (PCA) and spectral kurtosis, would be used to reduce the data dimension. Unlike the above methods, Deep Neural Networks (DNNs) can simultaneously achieve dimensional reduction and anomaly detection benefiting from the complex network topology. Three categories of DNN techniques are used to detect anomalies: classification, prediction, and reconstruction [9].

A binary classifier is trained by a dataset with predefined labels, which contains normal and anomalous data [10]. The disadvantage is that it is tough to collect various anomalous data due to the complexity of anomaly models in the CPS, leading to the limited usage of anomaly detection. Some kinds of classifiers can improve the flexibility, e.g., hierarchical neural networks [11], which refine the binary to multiple anomaly types. However, these methods assume the anomaly types have been revealed and collected in the dataset. These methods suit naive operational conditions due to the certainty of anomaly modeling.

The sequence predictors, such as Recurrent Neural Network (RNN), Hidden Markov Model (HMM), and Long Short Term Memory (LSTM), forecast and compare the predicted values with the corresponding data. Their deviation reflects whether the data is an anomalies. Recent works [3,9,12–14] show that the sequence predictors have advantages in the single time-series analysis with a large-scale training dataset. However, the performance is still limited to cope with the multi-dimensional time-series data. One possible solution is cascading the LSTM/RNN and simultaneously predicting the multiple dimension data [15,16] at the same time. However, the threshold for flagging the anomaly data relies on massive data amount and prior knowledge of the anomalous behavior.

Reconstruction methods [17–21] use the generative models, including Generative Adversarial Network (GAN) and AutoEncoder (AE), to detect anomalies according to the deviation between reconstruction results and original data. These approaches also address the preprocessing of high-dimensional input data, achieving dimension reduction and anomaly detection simultaneously through complex network topology. The hypothesis is that when data contains anomalies, the output of encoders changes into an unexpected value. Consequently, the result of decoders has a significant discrepancy with the original data. The limitation of this method is that it assumes the value of anomaly data would rarely occur in the training dataset [15,18,22], which makes it hard to cover the contextual anomalies, whose value is expected but in the wrong context. In addition, this method still suffers from the posterior collapse [19]: A powerful decoder for distinguishing anomalous data is leading to lower efficiency for anomaly detection.
We also investigate the anomaly detection practices applied on industrial information. In [23], the supervised methods such as Stochastic Gradient Descent (SGD), Support Vector Machines (SVM), and K-Nearest Neighbor (K-NN) are employed in the industrial network to detect anomalous data. However, these methods rely on labeled data which is labor-intensive and time-consuming to collect in industrial applications. As a specific application of information integration in healthcare [1,2], prosthetic systems can be equipped with intelligent detection devices without affecting the user’s daily life. Some anomaly detection methods have been used in the limb prosthesis [24,25]. However, these works usually require expert knowledge and operation assumption, lacking the readability analysis between the collected data and the observed conditions.

Inspired by the applications of VAE and SOM on data generalization and reconstruction in industrial-related fields [26–29], we propose combining these techniques to address anomaly detection of multi-dimensional data. By synthesizing the advantages of VAE and SOM, this paper aims to contribute to a semi-supervised learning architecture to detect anomalous data.

3. Methodology

3.1. Problem formalization

Obtaining accurate dynamics results of pressure distribution in a socket-stump interface prove to be a difficult task. Efforts have been made to estimate the static pressure at localized points in time, such as during donning. These pressures measured through patient trials have been compared against estimates through Finite Element Analysis (FEA) simulations [30]. However, little research has been done to estimate the dynamic distribution of pressure in the socket-stump pressure, which is necessary for anomaly detection. Some researchers [31–33] have demonstrated dynamic pressure distribution in the patients’ socket-stump interface. These experiments, however, require modification or refabrication of the socket to demonstrate the pressures. Additionally, the diversity of patients leads to the difficulty of finding a uniform pattern for anomaly detection. In this paper, we propose VAE-SOM architecture, as a generalized model for high dimensional data, based on the pressure distribution of the socket to detect anomalous data.

3.2. Dimension reduction and discretization

The piston force of the socket-stump interface generates time-series signals that can be divided into spatial and temporal properties. A suitable dimensional reduction method must be chosen for these multi-variant signals.

3.2.1. Learning probabilistic models by VAE

We model the data $X$ collected from the prosthetic sockets as following:

$$X^i = \{x^i_1, x^i_2, \ldots, x^i_s\}, X^i \subset X, i \in \{1, \ldots, s\}$$

where $X^i$ refers to a sequence that records the pressure variable gaits of the sensor $i$ with sampling stamps $\{1, \ldots, s\}$. $s$ indicates the total amount of sensors in the prosthetic sockets. For any timestamp $m$, sensor observations data $X$ can be modeled as follows:

$$X_m = \{x^1_m, x^2_m, \ldots, x^s_m\}, X_m \subset X, m \in \{1, \ldots, t\}$$

To obtain the generalized distribution of sensors pressure at any timestamp, we utilize VAE [34,35] in the following manner to analyze the data (as shown in Fig. 2). The encoder of VAE with parameters $p_θ$ is used for dimension reduction from the observations $X_m$ to latent space $z_m$. While the decoder with parameters $q_φ$ is used for reconstructing $z_m$ to $\hat{X}_m$. The encoder can be modeled as follows:

$$p_θ(z_m) = \int p_φ(z_m|x_m)p_θ(x_m)dx$$

where $z_m \in \mathbb{R}^d$ is the corresponding latent states (usually $d < s$), while the encoder compresses the complex high-dimensional data distribution (Eq. (2)) to a more tractable low dimensional distribution of $z_m$. In the decoder phase, latent variable $z_m$ is recovered to $\hat{x}_m$, which can be indicated as follows:

$$q_φ(\hat{x}_m) = \int q_φ(\hat{x}_m|z_m)q_φ(z_m)dz$$

Following the Eqs. (3) and (4), we derive the KL Divergence, which measures the discrepancy between $p_θ(x)$ and $q_φ(z)$, as the optimization criteria $\mathcal{L}(θ, φ, x_m)$ for training the VAE, which can be formalized as follows:

$$\mathcal{L}(θ, φ, x_m) = - D_{KL}(p_θ(z_m|x_m) \parallel q_φ(z)) + \mathbb{E}_{q_φ(z)}[log q_φ(z_m)]$$

The latent space $z = \{z_1, z_2, \ldots, z_n\}$ can be regularized as Gaussian Distribution by using parameterization tricks [34] which can be modeled as follows:

$$z_m ~ p_φ(z_m|x_m) = \mathcal{N}(\mu_m, \sigma_m^2)$$

By using above methods, we convert the multi-variant variables to the Multi Gaussian Distributions with various $\mu, \sigma$. These parameters support a robust latent representation of high-dimensional data while minimizing the discrepancy between the data distribution and the latent space by optimizing the KL Divergence. Such approaches generalize and explore the spatial property of high-dimensional input data containing uncertainty and stochastic. We have summarized state-of-the-art works which meet the problems of predicting temporal high-dimensional data due to the intractable complicated data features. However, the latent representation proposes a mapping from high-dimensional data to low-dimensional vectors and reserves input data information as much as possible, facilitating reliable models of temporal dependency and resolving the above issues.

3.2.2. Discretization for latent space

Although VAE obtains information on the high dimensional data by regularizing them as distributions, the time-dependency modeling still has not been solved, especially considering the contextual anomalies, which contain a normal value with unreasonable time. HMM infers the hidden states transformation based on multi-variant observations [3]. However, the method is limited by over-fitting, low precision, and hard generalization. To resolve these problems, we cluster the massive hidden data by SOM [36] to discrete and reduce the latent state space [37,38]. SOM can perform a non-linear mapping from the given high-dimensional input space to a usually one or two-dimensional map of neurons, which is denoted as $X \times N$ with one output neuron $l$ at each grid point. Each neuron of SOM has a $d$-dimensional weight vector. During the training phase, the neuron weights will update themselves by moving weight vectors toward the latent vectors as the following equation:

$$w_i(t+1) = w_i(t) + h_l(t) \cdot (z_m - w_i(t))$$

where $h_l(t)$ is the neighborhood kernel that determines the degree to which neuron $l$ updates toward the latent vector at time $t$. Typically, it is a Gaussian function centered on the winner neuron $c$. For each time step, every neuron calculates the distance (for example, Euclidean distance) between its weight vector $w_i$ and latent vector $z_m$. Therefore, the neuron closest to the input is called the winner neuron, and this phase can be presented as:

$$h_l(t) = a(t)exp\left(-\frac{||p_l - p_i||^2}{2\sigma(t)^2}\right)$$

Thus, the neighborhood function is defined as Eq. (8) where $t$ is the iteration times, $a(t)$ and $\sigma(t)$ are learning rate and neighborhood radius, respectively. And they all decrease monotonously with time $t$. $p_i \in \mathbb{R}^2$ and $p_k \in \mathbb{R}^2$ are positions of the winner neuron $c$ and neuron $l$ in
where $N$, $M$, and $K$ adapt to observed data to find the active states and their distribution.

Possible capacity of anomalies is done by comparing to the baseline.

3.3. Adaptive Markov chain modeling

We use a finite state, time-homogeneous Markov chain [39] to model the baseline (normal) dynamics in the discrete state space after VAE’s encoder compression and SOM clustering. The detection of anomalies is done by comparing to the baseline.

The unknown set of active states forms the most significant difference from classical Markov chain modeling. The SOM output provides a possible capacity of $M \times N$ states. But it is uncertain that each state can be active because $M$ and $N$ are subjective. Therefore, our model must adapt to observed data to find the active states and their distribution.

A Markov chain on a set of finite states $S$ is completely characterized by a $N_S \times N_S$ state transition matrix $P$ where $N_S = |S|$ is the number of states in $S$ and:

$$\sum_{j=0}^{N_S-1} P_{ij}(t) = 1, \quad 0 \leq P_{ij}(t) \leq 1,$$

where $i, j \in \{0, 1, \ldots, N_S - 1\}, t \in \mathbb{N}^+$. The temporal dynamics of state distribution $\pi(t)$ (a $1 \times N_S$ row vector) follows the following recursion:

$$\pi(t+1) = \pi(t)P(t)$$

In terms of baseline establishment, which corresponds to the normal stable operation, we are more interested in the limiting or steady-state distribution $\pi$ of the Markov chain satisfying:

$$\pi = \pi P$$

where $P = \lim_{n \to \infty} \frac{1}{n} \sum_{m=1}^{n} \prod_{t=1}^{m} P(t) = (\pi^T \ldots \pi^T)^T$. If $P$ is approximated by observed data, then $\pi$ can be obtained by algebraically solving the following over-determined system of linear equations:

$$\begin{cases} (I_{N_S} - P^T)e^T = 0_{N_S} \\ e_{N_S} = 1 \end{cases}$$

where $I_{N_S}$ indicates an identity matrix, $e_{N_S}$ is an $N_S$-dimensional unit column vector. This system of equations can be solved with the reduced form of QR decomposition.

Assume $N_0 = M \times N \geq N_S$ ($N_S$ being the unknown number of actual positive recurrent states of the Markov chain), we construct a Markov chain object with initial state update count matrix $U = \theta_{N_0 \times N_0}$, the actual state count vector $AC = \theta_{N_0}$, and an empty state transition history. Then with each $i \to j$ state transition, we increase the actual count $AC_i$ for the state $i$ and the $U_{ij}$ element of $U$. At any time $t$ into the simulation, the Markov chain object can be queried to return an empirical $L \times L$ state transition matrix which is a proper $L \times L$ slice of the full $N_0 \times N_0$ transition matrix $F$ formed by affinely combining the identity matrix $I_{N_0}$ and the actual state update count matrix $U(t)(N_0 \times N_0)$ by the following formula:

$$F_{is}(t) = \frac{1}{1+AC(t)}[I_{is} + U_{is}(t)] \forall i \in \{0, 1, \ldots, N_0 - 1\}$$

where $F_{is}(t), I_{is}$, and $U_{is}(t)$ are the $i$th row of $F(t), I_{N_0}$, and $U(t)$ respectively. $AC(t)$ is the actual count of state $i$ at time $t$. The non-zero elements of $AC(t)$ identify the active states and determine the proper slicing.

Since the set of active states is unknown, we implement a pruning function on the Markov chain object that prunes short-lived transient states after sufficient simulations by deleting them from the state transition history and updating the $U$ matrix and $AC$ vector accordingly. The probability threshold for pruning is empirical and problem specific. The frequency of conducting such pruning is also discretionary. As more data are fed to update the object followed with proper pruning, eventually $L \to N_S$ and $P \to (\pi^T \ldots \pi^T)^T$. In practice, some small variations will occur in the $\pi$s of the matrix. A baseline object consisting of the two-dimensional map. Therefore, the closer to the winner neuron, the more updated the neuron. After enough training iterations, similar neurons cluster together without supervised information.

We define the quantization error at timestamp $m$ as the minimum Euclidean distance between the latent vector and $w_i$:

$$L_{SOM} = \arg \min_{i \in M \times N} ||z_m - w_i||^2$$

Input data would be compressed by the encoder part of VAE into the latent space and recovered by the decoder. The decoder is responsible for evaluating the quality of the latent space. We measure the loss of VAE-SOM as follows:

$$L = L_{VAE} + L_{SOM}$$

where $L_{VAE}$ refers to the optimization criteria of VAE in Eq. (5) and $L_{SOM}$ refers to the loss of SOM in Eq. (9). A well-trained VAE can generate a fine-grained latent space. To fully represent the latent space, a higher-dimensional SOM is needed, which in turn leads to the problem of hard training for Markov chains because of massive states. Therefore, the trade-off between the performance of VAE and SOM is balanced by tuning their parameters, and this issue will be discussed in Section 4.

Fig. 2. Topology of VAE-SOM. The encoder with parameters $p_i$ is used for dimension reduction from input space to latent space, and the decoder with parameters $q_j$ is used for reconstructing $z_m$ or weight of winner neuron to $X$.

In the SOM part, the blue dots denote the latent variable $z_m$; the pink dot and solid line denote the clustering results by using SOM; the solid orange line denotes an example of the time sequence transformation of different states, which is modeled by adaptive Markov chain.
of $N_\gamma$ active states and its distribution $\pi$ is thus formed to characterize the stable normal dynamics.

A permanent structural change is identified by a change of the set of active states $S$ or its stable distribution $\pi$ or both. A transient anomaly is identified by a temporal detour of state trajectory outside of $S$ or a change of the empirical transition matrix $P$ compared with $(x^T \cdots x^T)^T$. Among these changes, we identify state trajectory deviation from $S$.

### 3.4. Anomaly detection

After the above steps, we have modeled the spatial and temporal perspective of multi-dimensional time-series data. Considering the difficulty of collecting the anomalous data under complex operational conditions, we use semi-supervised learning [40] to train the VAE-SOM architecture as well as the adaptive Markov chain. Following this idea, we build the training dataset with partially labeled data $X^a$, a subset of normal data in our case. The unlabeled data and the rest of the normal data $X'$ are used for building a test dataset $(X^a \cap X') = \emptyset$. Based on the assumption that the spatial and temporal properties of normal data show the strong-relevant, such as similar clusters or distributions, this semi-supervised architecture can improve its performance by exploring the training dataset $X^a$.

We use the transition probabilities among latent vectors to detect anomalous data by assigning a time window with length $T$. The log-likelihood from any latent sequence $\{z_i, \ldots, z_{i+T}\}$ can derive from the Eq. (13) by reforming the steady-state transition matrix of these states $[x_i, \ldots, x_{i+T-1}] \subseteq \pi$. We define a threshold $r$ which can be collected from the training phase of adaptive Markov chain:

$$r = \arg\min \{ -\alpha \log \sum_{n \in \pi} \pi_n \}$$

where $\alpha$ refers to a coefficient to control the sensitivity of the anomaly detector. When the transition probability is less than the threshold, it refers to such temporal features rarely occurring in the normal states. In this case, we highlight it as anomalous data.

### 4. Verification and validation

#### 4.1. Overview of dataset

To verify this architecture, we utilized two distinct datasets. Flexible Sensor Pressure Dataset (FSP) obtained by bio-mechanical simulation in conjunction with FEA simulation is used for dynamics pressures estimation [3,41] (Fig. 3). The piston force, acting on the hip joint, translated to the femur and then the socket-stump interface, is calculated in a biomechanical simulation to represent the pressures in real patients. In this dataset, large variations and asymmetry is usually an indication of discomfort or sensor failure [42]. By being able to detect variations from normal dynamics such as pressure distribution, it may be possible to detect discomfort or sensor failure in socket. These variations could also be characterized as anomalies in the function of the prosthetic and hence encourage designs to mitigate such anomalies.

To further demonstrate the effectiveness of VAE-SOM, we use a public dataset, Simulated Falls and Daily Living Activities Dataset (SFDLA) [43]. This SFDLA dataset contains similar features to our customized dataset. Motion sensor units collect data from the human body (head, chest, waist, right wrist, right thigh, and right ankle) with a sampling frequency of 25 Hz and an average duration of about 15 s.

#### 4.2. Experimental setup and evaluation metrics

**4.2.1. Data preprocessing**

In the FSP dataset, 5120 samples collected from unworn comfortable sock were used as train data and 25600 samples with anomaly spikes were used as test data. In the SFDLA dataset, we merged the data collected by 6 sensing units and obtained action data with 54 features. Because each action data contains several seconds of static state data before and after the action, we only used the action data within a 6-second window around the acceleration peak point. By shuffling the order of these action data, we got two action sequences. The action sequence with 30000 samples were used as train data, and the 150000 samples with anomaly spikes were used as test data. The details of the two datasets are shown in Table 1.

Different features in the above datasets were collected from different sensors, which results in different value ranges, so we normalized every feature between [0,1]. To detect context anomaly, we set a default rolling window with a window size of 10 and a step size of 1.

#### 4.2.2. Architecture

In our experiments, the input to VAE-SOM is a multi-dimensional time series. As discussed in Section 3, both the encoder and decoder parts are fully connected layers, and their topological structures are mirror-symmetrical. In the training phase, both the encoder and decoder parts work to obtain a high-performance encoder, and in the testing phase, only the encoder part works (as shown in Fig. 1). The SOM part is connected to the latent space, and the weight dimension of SOM neurons is consistent with the encoded latent vectors. The SOM is a square grid whose size ranges from $3 \times 3$ to $13 \times 13$. Adam is used as the optimizer with 0.001 as the learning rate.

#### 4.2.3. Baseline models and evaluation metrics

We select One-Class Support Vector Machine (OC-SVM) [10], and LSTM [12], aligned to the classification and prediction approach, as baseline models to compare the performance. Both OC-SVM and LSTM are commonly used basic models in anomaly detection tasks. OC-SVM learns the distribution area of the training dataset in the feature space and detects anomalies by calculating whether the test data sequence is in this area. LSTM, on the other hand, can forecast future data based on previous input sequences and detects anomalies by comparing the difference between its prediction and the actual test data. We also use the Hidden Markov Models (HMM), which detect anomalous data of the FSP dataset by using semi-supervised learning [3], to compare the performance with the VAE-SOM architecture. The value of True Positive (TP) is incremented by one if the previously recorded anomaly is included in the sequence that detected the presence of an anomaly;
Fig. 4. Test data and its output in VAE-SOM. The top image is the visual representation of the test data, the middle image is the result of the latent vector going through the decoder, and the bottom image is the decoding result of the SOM (9 × 9) weights mapping from the latent vector in order.

otherwise, the value of False Positive (FP) is incremented by one. Conversely, if the time sequence detected as normal does not overlap with previously recorded anomalies, the value of True Negative (TN) is increased by one; otherwise, False Negative (FN) is increased by one.

\[
\text{Precision} = \frac{TP}{TP + FP} \\
\text{Recall} = \frac{TP}{TP + FN} \\
F1 \text{ score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}
\]

According to the above definition, we can obtain the formulas shown in Eq. (17). In real-life scenarios, anomalies are not a common state, so the anomaly detection task focuses on the model’s accuracy in detecting anomalies and the proportion of detected anomalies in all recorded anomalies. Therefore, precision and recall are used as evaluation metrics to represent the performance of different models in the above two aspects. F1 score is the harmonic mean of Precision and Recall, which reflects the overall performance of different models and is the most crucial metric in an unbalanced dataset.

4.3. Results and analysis

4.3.1. Discrete representation and interpretability

In order to test the reconstruction performance of VAE-SOM, some experiments are performed on the FSP dataset with a 9 × 9 SOM. After training the model, the testing data sequence is encoded as a latent vector sequence. Because each latent vector maps to a winner neuron in the SOM, the latent vector can be replaced by the weight of the corresponding winner neuron. Therefore, we obtain a sequence of latent vectors and a sequence of weights.

The decoding results of the latent vector sequence and weight sequence are shown in Fig. 4, and the mean square error is used to calculate the difference between the two reconstruction results and the input data. SOM is a mapping from a continuous latent vector space to alternatives with a limited number, so the reconstruction error per sample increases from 0.0010 to 0.0059. Due to the utilization of 81 different patterns instead of the infinite number of latent vectors, the average reconstruction error increased by 0.0049. And the finite discrete states make the basis for time-series anomaly detection in VAE-SOM.

Table 2

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Precision</th>
<th>Recall</th>
<th>F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSP</td>
<td>OC-SVM</td>
<td>0.959</td>
<td>0.611</td>
<td>0.746</td>
</tr>
<tr>
<td></td>
<td>LSTM</td>
<td>0.803</td>
<td>0.960</td>
<td>0.874</td>
</tr>
<tr>
<td></td>
<td>HMM</td>
<td>0.825</td>
<td>0.961</td>
<td>0.887</td>
</tr>
<tr>
<td></td>
<td>VAE-SOM</td>
<td>0.982</td>
<td>0.868</td>
<td>0.922</td>
</tr>
<tr>
<td>SFDLA</td>
<td>OC-SVM</td>
<td>0.810</td>
<td>0.618</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td>LSTM</td>
<td>0.582</td>
<td>0.997</td>
<td>0.735</td>
</tr>
<tr>
<td></td>
<td>VAE-SOM</td>
<td>0.841</td>
<td>0.938</td>
<td>0.887</td>
</tr>
</tbody>
</table>

In order to show the interpretability of this architecture, the mapping result of the input (taking one channel of FSP dataset as an example) in a SOM is shown in Fig. 5. The size of the SOM is set to 3 × 3, so nine colors are used in Fig. 5 to represent the different winner neuron corresponding to each input. After training this architecture, we can construct a Markov transition matrix of the 9 SOM output neurons by fixing all the parameters of VAE-SOM and making all the training data pass the model in order again. When the window length is 2, the negative log-likelihood of a sequence can be calculated as \(-\log(p_{s_{i}} \times p_{s_{i-1}})\) and shown as green dotted line in Fig. 5. So the negative log-likelihood of the sequence containing anomalies is \(-\infty\) because the transmission probability is 0. After training, the threshold, represented by a red line, is set to the minimum negative log-likelihood and is used to detect anomalies in the testing phase according to the negative log-likelihood of the test sequence.

4.3.2. Anomaly detection performance

After learning on the training set, we examine the anomaly detection ability of these four models on the testing set and list the evaluation results in Table 2.

As shown in Table 2, VAE-SOM model has the highest F1 score and is better than our previous HMM-based work. The highest F1 score means that the compositive performance of VAE-SOM in both datasets is the best. Especially in the FSP dataset, the F1 score of VAE-SOM can reach 92.2%, and the precision rate can reach 98.2%. For the LSTM model, its recall rate is the highest, but precision is lower, which means that the TP and FP of the detection results are both high. Although most true anomalies are successfully detected, frequent error
Fig. 5. An example of anomaly detection with VAE-SOM. The dimension of the SOM is set to $3 \times 3$, and the window length is 2. The dashed black line indicates what the data recorded in channel 2 should have been. The dots represent the current value of the data, and a large gap from the dashed black line can be regarded as anomalous. The 9 colors of dots are used to distinguish the various mappings of these data in 9 SOM output neurons. Negative log-likelihood and threshold are plotted with green and red dashed lines, respectively. And the negative log-likelihood at both anomalies is larger than the threshold.

Table 3
Evaluation results with different SOM dimension and window size.

<table>
<thead>
<tr>
<th>SOM dimension</th>
<th>Window size</th>
<th>Precision</th>
<th>Recall rate</th>
<th>F1 score</th>
<th>F1 score (mean ± std.dev)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3 \times 3$</td>
<td>2</td>
<td>100.00%</td>
<td>11.34%</td>
<td>0.2037</td>
<td>0.3168 ± 0.0727</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>100.00%</td>
<td>18.42%</td>
<td>0.3111</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>100.00%</td>
<td>21.31%</td>
<td>0.3513</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>76.65%</td>
<td>27.16%</td>
<td>0.4011</td>
<td></td>
</tr>
<tr>
<td>$5 \times 5$</td>
<td>2</td>
<td>100.00%</td>
<td>38.29%</td>
<td>0.5537</td>
<td>0.6654 ± 0.0616</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>99.82%</td>
<td>55.78%</td>
<td>0.7157</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>94.71%</td>
<td>51.00%</td>
<td>0.6631</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>83.55%</td>
<td>58.63%</td>
<td>0.6691</td>
<td></td>
</tr>
<tr>
<td>$9 \times 9$</td>
<td>2</td>
<td>100.00%</td>
<td>80.99%</td>
<td>0.8995</td>
<td>0.9159 ± 0.0115</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>99.98%</td>
<td>83.82%</td>
<td>0.9119</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>98.21%</td>
<td>86.81%</td>
<td>0.9216</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>98.59%</td>
<td>88.02%</td>
<td>0.9306</td>
<td></td>
</tr>
<tr>
<td>$13 \times 13$</td>
<td>2</td>
<td>99.70%</td>
<td>82.62%</td>
<td>0.9036</td>
<td>0.9267 ± 0.0141</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>99.53%</td>
<td>87.21%</td>
<td>0.9296</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>99.46%</td>
<td>87.71%</td>
<td>0.9222</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>98.32%</td>
<td>90.32%</td>
<td>0.9415</td>
<td></td>
</tr>
</tbody>
</table>

4.3.3. SOM dimension and window size

The size of the SOM and the size of the window are both hyperparameters that affect the performance of VAE-SOM. To investigate the effect of these two parameters, we test this architecture on the FSP dataset, and the results are shown in Table 3.

As the window size increases, the precision shows a downward trend in all SOM sizes, while the recall rate gradually increases, and the F1 score also shows an upward trend. Under the same window size, with the increase of SOM size, the recall rate and F1 score show an upward trend. Moreover, as the SOM dimensions gradually increase, the F1 score variance between different window sizes gradually decreases, making the model more stable.

Above phenomena is due to the widespread presence of small-probability transitions in the transition matrix, which means that transitions between two states less likely to happen. When a test data is converted from one state to another due to an anomaly, the transition probability between the new state and the previous and posterior states maybe not 0. Therefore, when the window size is too short, this kind of anomaly will be ignored, and only sequences with zero transition probability are detected, resulting in extremely high precision but low recall rate. When the window size increases, the model can get more information at a time, more anomalies are checked out, and the recall rate will increase. The SOM dimension represents the number of discrete states. The larger the dimension, the finer the latent vectors are divided and the closer it is to $N_s$, which helps detect anomalies. When the SOM dimension is too small (such as $3 \times 3$), the data division is too vague, so only sequences with zero transition status can be distinguished, resulting in high precision and low recall rate.

4.3.4. Lightweight deployment of the VAE-SOM architecture

In real-life practice, run-time performance is also a critical factor that should be a concern. Specifically, lower power consumption is required to improve the portability of the anomaly detection system. Therefore, such a system tends to deploy in microelectronic devices with low computational energy consumption and fewer hardware resources. This poses limitations on the computational complexity and scalability of the anomaly detection devices, making lightweight architecture design necessary.
To evaluate the performance of VAE-SOM architecture in terms of lightweight during run time, we conducted experiments on the FSP dataset. All experiments are based on the same CPU (Intel i7-10750H). The encoder and decoder parts of VAE are 4-layer fully connected structures, and the size of SOM is set to 9 × 9. In order to achieve lightweight architecture after training, we prune the decoder network, which is trivial to detect anomalous data in the inference phase. FLOPs (Floating point operations) and latency are used as metrics to evaluate the computational complexity of the proposed method and the LSTM benchmark.

The evaluation results of the proposed architecture and benchmark are shown in Table 4. After pruning the decoder, the parameters of VAE-SOM architecture are 17.4% lower than LSTM. This means that the unique structure of VAE-SOM makes it more lightweight and has more robust anomaly detection capabilities. During the inference phase, the latency of the proposed architecture decreases up to 28 times than LSTM models, indicating this architecture can have superior real-time performance on microelectronic devices with lower computing resources and achieve lower power consumption.

5. Conclusion and future work

This article presents an anomaly detection approach for Industrial Information Integration Engineering (IIIE) in the context of bio-medical devices. We propose the VAE-SOM architecture for anomaly detection of multi-dimensional time series data in limb prostheses. Compared with existing works, the VAE of the proposed architecture extracts low-dimensional latent features from the multivariate time series data. Next, the SOM discretizes these continuous latent features, interpreting the operational data to analytic models. The adaptive Markov chains model the temporal transitions and adjust transition matrices depending on the run-time environment. Furthermore, We also present the possibility of the application for deploying in microelectronic devices. The results present a high precision and excellent run-time performance.

To further exploit our architecture and reduce the training data for various motions, such as running and gait shifting with different users, we propose extending the current adaptive Markov chain to switch these motions with a re-adaptive operator. In that case, any new update can be loaded when the new object is detected. New stable dynamics prepare to become active at the next re-adaptation after iterations. This procedure can also be triggered automatically by the KL divergence of stable state distribution between the current baseline object and the new one. Once this divergence reaches a certain threshold, a re-adaptation is triggered, and the new object becomes the new baseline object. By operating in this way, we have a series of baseline objects reflecting long-term structural changes (multiple communication classes) over time. And short-term temporal anomalies are detected by comparing them with the most recently enacted baseline object.

CRediT authorship contribution statement

Zikai Zhu: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. Peng Su: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. Sean Zhong: Formal analysis, Methodology. Jiayu Huang: Validation, Investigation, Writing – original draft. Suranjan Ottikkutti: Writing – review & editing, Visualization, Data curation, Resources. Kaveh Nazem Tahmasebi: Writing – review & editing. Zhuo Zou: Conceptualization, Methodology, Validation, Resources, Writing – review & editing, Supervision. Lirong Zheng: Conceptualization, Methodology, Validation, Resources. DeJiu Chen: Conceptualization, Methodology, Validation, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by the research project Socket-Sense, funded by the European Union’s Horizon 2020 under grant agreement No. 825429, the National Natural Science Foundation of China under grant No. 62011530132 (NSFC-STINT project), No. 62076066, and Shanghai Municipal Science and Technology Major Project No. 2021SHZDZX0103.

References


[41] Socket sense open access data. URL https://zenodo.org/record/7400478#.ZFuVcaBBye0.
