Time Series Anomaly Detection with Multiresolution Ensemble Decoding

Lifeng Shen [1], Zhongzhong Yu [2], Qianli Ma [2, 3], James T. Kwok [1]
[1] Department of Computer Science and Enginering, Hong Kong University of Science and Technology, Hong Kong
[2] School of Computer Science and Engineering, South China University of Technology, Guangzhou
[3] Key Laboratory of Big Data and Intelligent Robot (South China University of Technology), Ministry of Education
lshenae@cse.ust.hk, yuzzhong2020@foxmail.com, qianlima@scut.edu.cn, jamesk@cse.ust.hk

Abstract

Recurrent autoencoder has been one of the useful models for time series anomaly detection where outliers/abnormal segments can be identified due to their high reconstruction errors. However, existing recurrent autoencoders still easily suffer from the error accumulation issue during decoding. This paper explores a simple yet efficient strategy to alleviate this issue. Specifically, we propose a sequence-to-sequence network called Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED). Its core is to use lower-resolution information to help long-range decoding at layers with higher resolutions. This is achieved by jointly learning multiple recurrent decoders where each decoder is with a different decoding length. By introducing a coarse-to-fine fusion mechanism, temporal information can be shared across these decoders. Finally, a single ensemble output can be obtained. To stabilize our training process, we further introduce a multiresolution shape-forcing loss on the outputs of decoders whose decoding length is shorter than the original input length. Extensive empirical studies based on nine real-world benchmarks for time series anomaly detection demonstrate that the proposed dynamic ensemble decoding can outperform recent strong baseline methods.

Introduction

Anomaly detection aims to identify anomalous patterns from data. In particular, time series anomaly detection has received considerable attention in the past decade (Gupta et al. 2014; Cook, Misirli, and Fan 2020). Time series data can be easily found in many real-world applications. One example is cyber-physical systems such as smart buildings, factories, and power plants (Chia and Syed 2014; Ding et al. 2016), in which there are a large number of sensors. Efficient and robust time series anomaly detection can help monitor system behaviors such that potential risks and financial losses can be avoided. However, detecting outliers from time series data is challenging. First, finding and labeling of anomalies are very time-consuming and expensive in practice. Moreover, time series data usually have complex nonlinear and high-dimensional dynamics that are difficult to model. To alleviate the first issue, time series anomaly detection is usually formulated as a one-class classification problem (Ruff et al. 2018; Zhou et al. 2019), in which the training set contains only normal samples.

Existing time series anomaly detection techniques can be categorized as either predictive methods (Chen et al. 2018) or reconstruction-based methods (Malhotra et al. 2016; Kieu, Yang, and Jensen 2018). Classical predictive models including autoregressive moving average (ARMA) (Wold 1938) and autoregressive integrated moving average (ARIMA) (Yu, Jibin, and Jiang 2016) build linear regressors and the prediction error is used as anomaly score. More recently, recurrent neural networks (RNNs) (Zhang et al. 2019) and other deep predictors (Filonov, Lavrentyev, and Vorontsov 2016) are also similarly used. However, these methods largely depend on the models' extrapolation capacity (Yoo, Kim, and Kim 2019). On the other hand, reconstruction-based methods learn a compressed representation for the core statistical structures of normal data, and then reconstruct the input from this compressed representation. Then all points within a given time series are detected where large reconstruction errors indicate outliers. Reconstruction methods are popular in many practical applications, e.g., anomalous rhythm detection (Zhou et al. 2019) and network traffic monitoring (Kieu, Yang, and Jensen 2018). In this paper, we focus on reconstruction methods.

In deep learning, recurrent neural network (RNN) has been a key model for temporal modeling. Combined with the strong temporal modeling capacity of RNN, the recurrent auto-encoder (RAE) (Malhotra et al. 2016) has demonstrated good performance in time series anomaly detection. Following the sequence-to-sequence framework (Sutskever, Vinyals, and Le 2014), RAE consists of an encoder and a decoder. The reconstruction error at each time step is used as an anomaly score. The main difference with standard sequence-to-sequence models is that RAE decodes the time series in time reverse order, which is usually easier for the decoder to regenerate the input time series from the encoder's compressed representation. reverse order will be However, RAE can have difficulties in decoding long time series due to error accumulation. Very recently, a number of autoencoder variants have been proposed for time series anomaly detection. Yoo, Kim, and Kim (2019) developed the recurrent reconstructive network (RRN), which uses self-attention and feedback transition to help capture the temporal dynamics. A regularizer on the encoder-decoder states is also introduced on training. Kieu et al. (2019) used ensemble learning (Dietterich 2000) and proposed the recurrent autoencoder ensemble. Both the encoders and decoders consist of several recurrent neural networks with sparse skip connections. On inference, the median reconstruction error from all decoders is used as the anomaly score. However, these recurrent autoencoder variants still suffer easily from error accumulation.

In this paper, we propose the Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED), which is also based on the sequence-to-sequence model. The key idea is to model temporal patterns at different resolutions, and then use the lower-resolution temporal information to guide decoding at a higher resolution. This can be understood from the fact that lower-resolution temporal patterns often represent macro characteristics of the time series (such as trend patterns and seasonality). This can be used to avoid overfitting on the complex and highly nonlinear local patterns at a higher resolution, and alleviate the accumulation of errors during decoding.

Inspired by (Kieu et al. 2019), the proposed model is also an ensemble of decoders. However, the difference is that the proposed decoders capture the input time series' temporal information at multiple resolutions. This is achieved by controlling the number of decoding steps in the decoders. With a short decoding length, the decoder has to focus on the macro temporal characteristics; whereas a decoder with a long decoding length can capture more detailed local temporal patterns. By using multiple decoding steps, we can increase the diversity of decoders which would potentially help the ensemble performance. Furthermore, instead of simply averaging the decoder outputs as in (Kieu et al. 2019), they are integrated in a flexible and robust manner. Specifically, we introduce a multiresolution shape-forcing loss to encourage the decoders to match the input's global temporal shape at multiple resolutions, in which the similarity is defined by the dynamic time warping (DTW) distance (Sakoe and Chiba 1978). Finally, the output from the highest resolution (whose decoding length equals the length of the whole time series) is used as the final ensemble output.

Our main contributions can be summarized as follows:

- We present a novel recurrent autoencoder RAMED, which has multiple decoders with different decoding lengths. By introducing a shape-forcing reconstruction loss, decoders can capture global temporal characteristics of time series at multiple resolutions.
- We introduce a fusion mechanism to integrate multiresolution temporal information from multiple decoders.
- We conduct extensive empirical studies on time series anomaly detection. Results demonstrate that the proposed model outperforms recent strong baselines.

Related Work

Time Series Autoencoders

Sequence-to-sequence is a popular framework to encoder sequential data, which has been widely explored for many NLP applications, e.g., machine translation (Bahdanau, Cho, and Bengio 2015). This framework has also been extended to many time series applications (e.g., time series prediction, clustering, and anomaly detection). In anomaly detection, this framework needs to be modified by reconstructing a reversed input sequence/time series during decoding. This requires that its encoder and decoder have the same decoding lengths. As mentioned before, recurrent neural network-based methods including Recurrent Auto-Encoder (RAE) (Malhotra et al. 2016), Recurrent Reconstructive Network (RRN) (Yoo, Kim, and Kim 2019) are two recent representative time-series autoencoders. Both of them follow the standard setting of the sequence-to-sequence framework. They decode the input time series from a compressed hidden representation vector. Our proposed method is similar to them since our model is also built upon the sequence-to-sequence framework. But the difference is that our model is an ensemble model that consists of multiple rnns, while both RAE and RRN are based on single-layer RNN or hierarchical variants.

Kieu et al. (2019)'s RAE-ensemble model is one work mostly related to us. Rather than simply pooling multiple regressors' outputs to obtain a final ensemble output in (Kieu et al. 2019), our work focus on employing multiresolution temporal information to improve existing recurrent autoencoder ensemble framework. This design intuitively provides a coarse-to-fine decoding process. At the lowest resolution layer (corresponding to the decoder with the shortest length), macro-temporal characteristics are encouraged to learn. Subsequently, the learned information will be passed to the next layer (with a higher decoding resolution) such that the decoder with a high-resolution decoding requirement would not overfit shorter-term patterns.

Apart from the above work, convolution-based autoencoders are also explored. Zhou et al. (2019) recently propose a BeatGAN for time series anomaly detection. By combining simple temporal convolution-deconvolution module and adversarial training, it achieves a good performance on the detection of anomalous beats from electrocardiogram (ECG) readings. In our experiments, we will choose it as one of our main baselines.

Multiresolution Temporal Modeling

There are many works exploring multiresolution temporal structure in time series. Hihi and Bengio (1996) developed a hierarchical RNN to capture multiresolution temporal information by integrating multiple delays and time scales in different recurrent neurons. Recently, Chung, Ahn, and Bengio (2017) further proposed a hierarchical multiscale rnn to model the multiscale structure within text sequences. Similar to us, Liu et al. (2019) also explores a coarse-to-fine modeling procedure in their hierarchical LSTM. However, Liu et al. (2019)'s work focus on time series imputation. Give a series of observations, they utilized the coarser information to impute the missing values. Our work is different from this setting since we collect multiple decoders and each of them reconstructs input at different resolutions. Through a multiresolution fusion mechanism, a single final output can be obtained. To the best of our knowledge, we are the first to introduce multiresolution temporal modeling into a recurrent autoencoder ensemble.

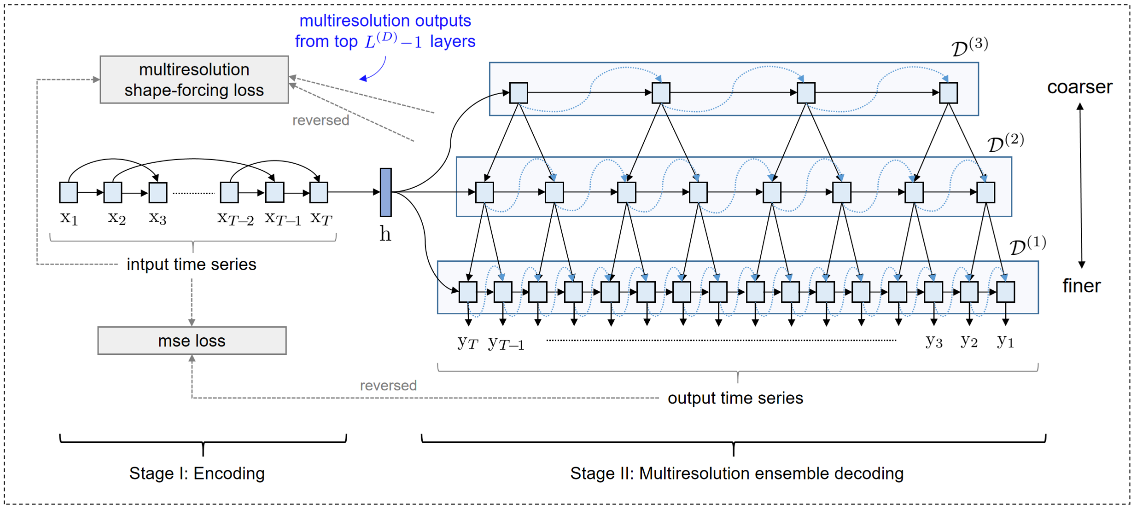

Figure 1: The proposed Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED), with $L^{(E)} = 1$.

Proposed Architecture

Let $\mathbf{X}^{(i)} = [\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \dots, \mathbf{x}_{T^{(i)}}^{(i)}]$ be the ith of time series. $T^{(i)}$ is the length of $\mathbf{X}^{(i)}$. The goal of time series anomaly detection is to generate an anomaly score/label for each input $\mathbf{x}_t^{(i)}$ in $\mathbf{X}^{(i)}$. To simplify notations, we will drop the superscript i in the sequel.

Model Overview

Figure 1 shows the proposed Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED) model. Following the standard sequence-to-sequence framework, there are two stages: (i) Encoding, which compresses the input time series $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]$ to a hidden feature representation $\mathbf{h}$. The encoder is usually a recurrent network, such as a LSTM; (ii) Decoding, which reconstructs the input $\mathbf{X}$ from $\mathbf{h}$. Again, the decoder is usually a recurrent network. In ROMED, we obtain multiple reconstructions by running a number of recurrent networks on $\mathbf{h}$. These decoders use different numbers of decoding steps to capture the temporal characteristics at multiple resolutions. To allow efficient information sharing among the decoders, a multiresolution fusion strategy is used to fuse the decoder outputs in a coarse-to-fine manner. Moroever, the decoded output is encouraged to be similar to the input time series by using a differentiable shape-forcing loss based on the dynamic time warping paths at different resolution levels. Details for each of these components will be described in the following sections.

Time Series Encoding

The recurrent neural network (RNN) has been commonly used to represent time series data. At time t, the update in a standard RNN is of the form:

$$\mathbf{h}_t^{(E)} = f^{(E)}([\mathbf{x}_t; \mathbf{h}_{t-1}^{(E)}]), \qquad (1)$$

where $\mathbf{h}_{t-1}^{(E)}$ is the hidden state from time $t-1$, and $f^{(E)}$ is a nonlinear function. A popular choice for $f^{(E)}$ is the long-short term memory (LSTM) unit (Hochreiter and Schmidhuber 1997). Kieu et al. (2019) suggest adding sparse skip connections to the RNN cells so that additional hidden states in the past can be considered. In other words, $f^{(E)}$ uses not only the immediate previous state $\mathbf{h}_{t-1}^{(E)}$, but also $\mathbf{h}_{t-s}^{(E)}$ for some skip length $s > 1$:

$$\mathbf{h}_t^{(E)} = f^{(E)}\left(\left[\mathbf{x}_t; \frac{w_1 \mathbf{h}_{t-1}^{(E)} + w_2 \mathbf{h}_{t-s}^{(E)}}{|w_1| + |w_2|}\right]\right). \qquad (2)$$

Here, the skip length s is randomly sampled from [1, 10]. (w_1, w_2) are randomly sampled from $\{(1,0), (0,1), (1,1)\}$ at each time step. In the proposed model, we use $L^{(E)}$ such RNNs to construct an ensemble of encoders. Their outputs are then combined as:

$$\mathbf{h} = F_{\text{MLP}}\left(\text{concat}[\mathbf{h}_1^{(E)}; \dots; \mathbf{h}_T^{(E)}]\right), \qquad (3)$$

where F_{MLP} is a fully-connected layer. $\mathbf{h}_T^{(E)}$ denotes the hidden state driven by $\mathbf{x}_t]$ and $t = 1, 2, \dots, T$.

Multiresolution Ensemble Decoding

In time series autoencoders, it is popular to reconstruct the input time series in reverse time order (Kieu et al. 2019; Yoo, Kim, and Kim 2019). In other words, for input $[\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]$, the target reconstructed output is $[\mathbf{x}_T, \mathbf{x}_{T-1}, \dots, \mathbf{x}_1]$. However, recurrent autoencoders (and their ensemble variants) may easily suffer from error accumulation during the sequential decoding process.

To alleviate this problem, we propose to use an ensemble of decoders with different decoding lengths. With a short decoding length, the decoder has to focus on the macro temporal characteristics; whereas a decoder with long decoding length can capture more detailed local temporal patterns. Thus, these different decoders are expected to capture temporal behaviors of the time series at different resolutions.

Specifically, let $L^{(D)}$ be the number of decoders. The kth decoder $\mathcal{D}^{(k)}$ ($k = 1, 2, \dots, L^{(D)}$) reconstructs a length-$T^{(k)}$ time series, where $T^{(k)} = \alpha_k T$ and

$$\alpha_k = 1/\tau^{k-1} \in (0, 1] \tag{4}$$

for some $\tau > 1$ ($\tau = 2$ in Figure 1). Note that $\alpha_1 = 1$ and $T^{(1)} = T$. We require $T^{(D)} \geq 2$, so that the decoder at the top takes at least two decoding steps. Each decoder can be any recurrent network. Here, we use the LSTM. $\mathcal{D}^{(k)}$ outputs $\mathbf{y}_t^{(k)}$ (from $t = T^{(k)}, T^{(k)}-1, \dots, 2$ as we decode in reverse time order) as:

$$\begin{aligned} \mathbf{y}_t^{(k)} &= \mathbf{W}^{(k)} \mathbf{h}_t^{(k)} + \mathbf{b}^{(k)}, \\ \mathbf{h}_{t-1}^{(k)} &= \text{LSTM}^{(k)}([\mathbf{y}_t^{(k)} + \epsilon \boldsymbol{\delta}; \mathbf{h}_t^{(k)}]), \end{aligned} \tag{5}$$

where $\mathbf{W}^{(k)}$ and $\mathbf{b}^{(k)}$ are learnable parameters, and $\mathbf{h}_{T^{(k)}}^{(k)}$ is initialized to zero. To improve robustness, we add a small amount of noise $\epsilon \boldsymbol{\delta}$ to the LSTM's input as in the denoising autoencoder (Vincent et al. 2008), where ϵ is a small scalar (we use 10^{-4}), and $\boldsymbol{\delta}$ is random noise from the standard normal distribution $\mathcal{N}(0, 1)$.

Coarse-to-Fine Fusion Since the outputs from different decoders have different lengths $T^{(k)}$'s, they cannot be easily summarized to an ensemble output by using the average or median as in (Kieu et al. 2019). Consider two decoders $\mathcal{D}^{(k+1)}$ and $\mathcal{D}^{(k)}$. Note from (4) that $T^{(k)} = \tau T^{(k+1)} > T^{(k+1)}$, and so information extracted from $\mathcal{D}^{(k+1)}$ is coarser than that from $\mathcal{D}^{(k)}$. In the following, we propose a simple yet efficient multiresolution coarse-to-fine strategy to fuse the coarser-grained decoder information with the finer-grained decoders in forming the ensemble output.

We start with the decoder at the top ($k = L^{(D)}$). Its output $\{\mathbf{h}_t^{(L^{(D)})}\}$, which is the coarsest among all decoders, is obtained from (5). For decoder $\mathcal{D}^{(k)}$ ($k = L^{(D)} - 1, \dots, 1$), instead of simply using (5), it first combines its previous hidden state $\mathbf{h}_{t+1}^{(k)}$ with the corresponding slighthly-coarser information $\mathbf{h}_{\lceil t/\tau \rceil}^{(k+1)}$ from sibling decoder $\mathcal{D}^{(k+1)}$ as:

$$\hat{\mathbf{h}}_t^{(k)} = \beta \mathbf{h}_{t+1}^{(k)} + (1-\beta) F'_{\text{MLP}} \left(\text{concat}[\mathbf{h}_{t+1}^{(k)}; \mathbf{h}_{\lceil t/\tau \rceil}^{(k+1)}] \right), \tag{6}$$

where F'_{MLP} is a two-layer fully-connected network with the PReLU (Parametric Rectified Linear Unit) (He et al. 2015) activation, and $\beta \mathbf{h}_{t-1}^{(k)}$ (with $\beta > 0$) plays a similar role as the residual connection (He et al. 2016). Analogous to (5), this $\hat{\mathbf{h}}_t^{(k)}$ is then fed into the LSTM cell to generate

$$\mathbf{h}_t^{(k)} = \text{LSTM}^{(k)}([\mathbf{y}_{t+1}^{(k)} + \epsilon \odot \boldsymbol{\delta}; \hat{\mathbf{h}}_t^{(k)}]), \;\; t = T^{(k)}-1, \dots, 1.$$

Finally, after reversing to the original time order, the ensemble's reconstructed output can be obtained from the bottommost decoder as $\overleftarrow{\mathbf{Y}}_{\text{recon}} = [\mathbf{y}_1^{(1)}, \mathbf{y}_2^{(1)}, \dots, \mathbf{y}_T^{(1)}]$.

Multiresolution Shape-forcing Loss To encourage $\overleftarrow{\mathbf{Y}}_{\text{recon}}$ to be close to the input $[\mathbf{x}_1, \dots, \mathbf{x}_T]$, we use the square loss to measure the reconstruction error:

$$\mathcal{L}_{\text{MSE}} = \sum_{t=1}^{T} \|\mathbf{y}_t^{(1)} - \mathbf{x}_t\|_2^2. \tag{7}$$

To further encourage the decoders to learn cossistent temporal patterns at different resolutions, we introduce a smoothed DTW loss to force the decoders' outputs to have similar shape as the original input. Let the output (in original time order) from the kth recurrent network $\mathcal{D}^{(k)}$ be $\overleftarrow{\mathbf{Y}}^{(k)} = [\mathbf{y}_1^{(k)}, \mathbf{y}_2^{(k)}, \dots, \mathbf{y}_{T^{(k)}}^{(k)}]$. Since $T^{(k)} \neq T$ for $k = 2, \dots, L^{(D)}$, we define a similarity between time series $[\mathbf{x}_1, \dots, \mathbf{x}_T]$ and each $\overleftarrow{\mathbf{Y}}^{(k)}$ by dynamic time warping (DTW) (Sakoe and Chiba 1978). The DTW similarity is based on distances along the (sub-)optimal DTW alignment path. Let the alignment be represented by a matrix $\mathbf{A} \in \{0, 1\}^{T \times T^{(k)}}$, in which $\mathbf{A}_{i,j} = 1$ when $\mathbf{x}_i$ is aligned to $\mathbf{y}_j^{(k)}$; and zero otherwise, and with boundary conditions that $\mathbf{A}_{1,1} = 1$ and $\mathbf{A}_{T,T^{(k)}} = 1$. All valid alignment paths run from the upper-left entry (1,1) to the lower-right entry $(T, T^{(k)})$ using moves $\downarrow$, $\rightarrow$ or $\searrow$. The costs on the alignments is stored in a matrix $\mathbf{C}$. For simplicity, we use $\mathbf{C}_{i,j} = \|\mathbf{x}_i - \mathbf{y}_j^{(k)}\|$, the Euclidean distance. The DTW distance between $\mathbf{X}$ and $\overleftarrow{\mathbf{Y}}^{(k)}$ is then given by:

$$\text{DTW}(\mathbf{X}, \overleftarrow{\mathbf{Y}}^{(k)}) = \min_{\mathbf{A} \in \mathcal{A}} \langle \mathbf{A}, \mathbf{C} \rangle. \tag{8}$$

where $\mathcal{A}$ is the set of $T \times T^{(k)}$ binary alignment matrices, and $\langle \cdot, \cdot \rangle$ is the matrix inner product. However, the DTW distance is non-differentiable due to the min operator. To integrate DTW into end-to-end training, (8) can be replaced by the smoothed DTW (sDTW) distance (Cuturi and Blondel 2017):

$$\text{sDTW}(\mathbf{X}, \overleftarrow{\mathbf{Y}}_{\text{multires}}^{(k)}) = -\gamma \log \left(\sum_{\mathbf{A} \in \mathcal{A}} e^{-\langle \mathbf{A}, \mathbf{C} \rangle / \gamma} \right), \tag{9}$$

where $\gamma > 0$. This is based on the smoothed min operator $\min^{\gamma}\{a_1, \dots, a_n\} = -\gamma \log \sum_{i=1}^{n} e^{-a_i/\gamma}$, which reduces to the min operator when γ approaches zero.

With the sDTW distance, we encourage decoders at different resolutions to output time series with similar temporal characteristics as the input. This leads to the following multiresolution shape-forcing loss:

$$\mathcal{L}_{\text{shape}} = \frac{1}{L^{(D)} - 1} \sum_{k=2}^{L^{(D)}} \text{sDTW}(\mathbf{X}, \overleftarrow{\mathbf{Y}}^{(k)}). \tag{10}$$

Combining with (7), the final loss is:

$$\mathcal{L} = \mathcal{L}_{\text{MSE}} + \lambda \mathcal{L}_{\text{shape}}, \tag{11}$$

where λ is a trade-off hyperparameter. For training, we use stochastic gradient descent on $\mathcal{L}$ with the Adam (Kingma and Ba 2015) optimizer. The training procedure is shown in Algorithm 1.

Anomaly Score and Detection

Given an unseen time series $\mathbf{X}$ with length T and the element from $\mathbf{X}$ at time t is denoted by $\mathbf{x}_t \in \mathbb{R}^d$. Let $\mathbf{y}_t$ be the corresponding reconstruction produced by the model, and $\mathbf{e}(t) = \mathbf{y}_t - \mathbf{x}_t$. Using the validation set, we fit a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ onto the set of $\mathbf{e}(t)$'s.

Algorithm 1 Training the proposed RAMED model.

```
Input: a set of time series $\{\mathbf{x}_i\}$; batchsize $B$; number of decoders $L^{(D)}$; $\tau$.
for each decoder, its decoding length is $T^{(k)} = \alpha_k T$;
repeat
    sample a batch of time series;
    for sample $i = 1, \dots, B$ do
        feed $\{\mathbf{X}_i\}$ to encoders and obtain hidden states;
        obtain joint representations $\{\mathbf{h}_i\}$ via (3);
        for $k = L^{(D)}, L^{(D)} - 1 \dots, 2$ do
            run the decoder $\mathcal{D}^{(k)}$;
            if $(k \neq L^{(D)})$ then
                perform coarse-to-fine fusion;
            end if
            obtain updated hidden states $\{\mathbf{h}_t^{(k)}\}$ and outputs $\{\mathbf{y}_t^{(k)}\}$ for all $t = T^{(k)}, T^{(k)} - 1, \dots, 1$.;
        end for
    end for
    minimize (11) by stochastic gradient descent;
until convergence.
```

On inference, the probability that $\mathbf{x}_{\text{test}}$ is anomalous can be defined as:

$$1 - \frac{1}{\sqrt{(2\pi)^d |\mathbf{\Sigma}|}} \exp\left(- \frac{(\mathbf{e}(t) - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{e}(t) - \boldsymbol{\mu})}{2} \right).$$

where d denotes the dimension of input $\mathbf{x}_t$. Thus, we can take

$$s(t) = (\mathbf{e}(t) - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{e}(t) - \boldsymbol{\mu}). \tag{12}$$

as the anomaly score. When this is greater than a predefined threshold, $\mathbf{x}_t$ is classified as an anomaly.

Experiments

In this section, experiments are performed on a number of real-world time series datasets, and the proposed RAMED model is compared with recent recurrent autoencoder models. Ablation studies are also provided.

Setup

Datasets. We use nine commonly-used real-world time series benchmarks[1] (Table 1):

- *ECG*: These are often used for detection of anomalous beats from electrocardiograms readings.
- *2D-gesture*: This contains time series of X-Y coordinates of an actor's right hand. The data is extracted from an video in which the actor grabs a gun from his hip-mounted holster, moves it to the target, and returns it to the holster. The anomalous region is in the area where the actor fails to return his gun to the holster.
- *Power-demand*: This contains one year of power consumption records measured by a Dutch research facility in 1997.
- *Yahoo's S5 Webscope*: This contains records from real production traffic of Yahoo's website. Anomalies are manually labeled by human experts.

For *ECG*, *2D-gesture* and *Power-demand*, the training set contains only normal data, and 30% of them is used for validation. The model with the best reconstruction loss on the the validation set is selected for evaluation. For *Yahoo's S5*, we split the whole time series the raw time series data is partitioned into three parts (40% for training, 30% for validation, and the remaining 30% for testing). The training part contains unknown anomalies and we use the model with the highest F1-value on the validation set for evaluation. All time series from nine datasets are partitioned into fixed-length sequences by using a sliding window (with stride 16 for all ECG datasets, 256 for *Power-demand* and 32 for others). Detailed information including the window size and the number of sequences in each partition (training/validation/testing) is summarized in Table 1.

Table 1: Statistics of the time series data sets. *Power-demand* and *Yahoo's S5* are univariate time series, while *ECG* and *2D-gesture* are bivariate.

Datasets	length	# trn	# val	# tst	% anomaly
ECG					
(A) chfdb_chf01_275	64	78	9	29	14.61
(B) chfdb_chf13_45590	64	207	12	21	12.35
(C) chfdbchf15	64	473	51	53	4.45
(D) ltstdb_20221_43	64	112	13	18	11.51
(E) ltstdb_20321_240	64	85	10	23	9.61
(F) mitdb_100_180	64	126	14	36	8.38
2D-gesture	64	180	39	47	24.63
Power-demand	512	49	11	29	11.44
Yahoo's S5	128	1,314	199	198	3.20

Baselines The proposed RAMED is compared with four recent anomaly detection baselines: (i) Recurrent autoencoder (RAE) (Malhotra et al. 2016); (ii) Recurrent reconstructive network (RRN) (Yoo, Kim, and Kim 2019), which combines attention, skip transition and a state-forcing regularizer; (iii) Recurrent Autoencoder Ensemble (RAE-ensemble) (Kieu et al. 2019), which uses an ensemble of RNNs with sparse skip connections as encoders and decoders; (iv) BeatGAN (Zhou et al. 2019), which is a recent CNN autoencoder-based generative adversarial network (GAN) for time series anomaly detection.

Evaluation Metrics. The classification as anomalies depends on the threshold setting for the anomaly score. Hence, instead of using precision and recall for performance evaluation, we use the (i) area under the ROC curve (AUROC), (ii) area under the precision-recall curve (AUPRC), and (iii)

[1] *ECG*, *2D-gesture* and *Power-demand* are from http://www.cs.ucr.edu/~eamonn/discords/, while *Yahoo's S5* is from https://webscope.sandbox.yahoo.com/.

Table 2: Anomaly detection results. Best results are highlighted. The larger the better. Average rank (the smaller the better) is recorded in the last column.

Metrics	*Methods*	*ECG*						*2D-gesture*	*Power-demand*	*Yahoo's S5*	*Avg Rank.*
		A	B	C	D	E	F				
AUROC	BeatGAN	0.6651	0.7314	0.5869	0.5933	0.8298	0.4419	0.7242	0.6180	0.8542	4.2
	RAE	0.6495	0.7524	0.6827	0.6071	0.7792	0.4468	0.7541	0.6365	0.8725	3.3
	RRN	0.6950	0.7207	0.6849	0.4705	0.7881	0.4787	0.7511	0.6203	0.8340	3.8
	RAE-ensemble	0.6826	0.7763	0.7055	0.6464	0.8314	0.3966	**0.7953**	**0.6445**	0.8693	2.3
	RAMED	**0.7358**	**0.7882**	**0.7879**	**0.6944**	**0.8336**	**0.5564**	0.7804	0.6212	**0.8874**	**1.3**
AUPRC	BeatGAN	0.5250	0.4494	0.1901	0.1484	0.3446	0.0766	0.4925	0.1249	0.4515	3.9
	RAE	0.5184	0.4032	0.3123	0.1554	0.2417	0.0776	0.5045	0.1334	0.4504	3.8
	RRN	0.5490	0.4313	0.3349	0.1163	0.3768	0.0793	0.4893	0.1258	0.4382	3.8
	RAE-ensemble	0.5623	0.5421	**0.4990**	**0.1847**	0.3848	0.0725	0.5320	**0.1352**	0.4507	2.1
	RAMED	**0.5714**	**0.5423**	0.3463	0.1778	**0.4578**	**0.1059**	**0.5746**	0.1269	**0.4599**	**1.4**
F1	BeatGAN	0.5193	0.4518	0.2799	0.2367	0.4702	0.1668	0.4923	0.2617	0.4230	3.9
	RAE	0.5251	0.4903	0.3279	0.2543	0.3363	0.1547	0.5339	**0.2760**	0.4219	3.4
	RRN	0.5608	0.4348	0.3830	0.2064	0.4437	0.1547	0.5309	0.2625	0.4233	3.4
	RAE-ensemble	**0.5642**	**0.5240**	**0.5868**	0.2775	0.4498	0.1547	**0.5625**	0.2699	0.4222	2.0
	RAMED	0.5427	0.5103	0.3408	**0.3087**	**0.5223**	**0.2063**	0.5344	0.2650	**0.4242**	**1.9**

best F1-score, which is the highest F1-score the model can achieve (Li et al. 2020; Su et al. 2019). This is done by enumerating 1000 thresholds uniformly distributed from 0 to the maximum value of the anomaly score for all time steps in the test set (Yoo, Kim, and Kim 2019). These metrics have been widely used in anomaly detection (Wang et al. 2019; Ren et al. 2019; Li et al. 2020; Su et al. 2019).

Implementation Details. Following (Kieu et al. 2019), multiple recurrent encoders with different fixed skip connections are used in our encoding stage (Figure 1). In our experiments, 3 decoders and 3 encoders are used. Each decoder is a single-layer LSTM with 64 units. We perform grid search on the hyperparameter β in (6) in the range $[0.1, 0.9]$ at an interval of 0.1, τ in (4) from $\{2, 3, 4\}$, and γ in (9) is set to 0.01. For fairness, all baselines use the same window length T. Our model is implemented in PyTorch. The Adam optimizer is used with an initial learning rate of 10^{-3}. Experiments are run on a machine with Intel Core i7-7700K, 4.20-GHz CPU, 32-GB RAM, and a GeForce GTX 1080-Ti 11G GPU.

Results

Results are shown in Table 2. As can be seen, RAE-ensemble has better performance than BeatGAN, RAE and RRN (with average ranks 2.3/2.1/2.0 for AUROC/AUPRC/F1, respectively). This agrees with the general view that ensemble learning is beneficial. Compared to RAE-ensemble, the proposed RAMED depends on multiresolution ensemble decoding. And RAMED (with average ranks 1.3/1.4/1.9 for AUROC/AUPRC/F1, respectively) outperforms RAE-ensemble, demonstrating that using multiresolution information in an ensemble can help time series reconstruction.

Ablation Study

In this section, we examine the contributions of the coarse-to-fine fusion strategy and multiresolution shape-forcing loss in the proposed RAMED model. Experiments are performed on the *ECG(A)* and *Yahoo's S5* data sets.

Effect of coarse-to-fine fusion Results are shown in Table 3. As can be seen, when only the bottom decoder (with the highest resolution) is used to reconstruct the input time series, performance in terms of all metrics decrease. The drops in AUROC / AUPRC / F1 are 8.79% / 6.79% / 4.97%, respectively on *ECG(A)*, and 0.98% / 0.42% / 0.21%, respectively, on *Yahoo's S5*. This verifies usefulness of the multiresolution information.

Table 3: Effectiveness of coarse-to-fine fusion in the proposed RAMED model.

ECG(A)	AUROC	AUPRC	F1
w/o coarse-to-fine fusion	0.6479	0.5035	0.4965
full model	**0.7358**	**0.5714**	**0.5427**

Yahoo's S5	AUROC	AUPRC	F1
w/o coarse-to-fine fusion	0.8780	0.4557	0.4221
full model	**0.8878**	**0.4599**	**0.4242**

Effect of multiresolution shape-forcing loss Multiresolution shape-forcing loss encourages decoders to learn temporal information from various resolutions. Results in Table 4 show that the shape-forcing loss also plays an important role in multiresolution decoding.

Table 4: Effectiveness of multiresolution shape-forcing loss $\mathcal{L}_{\text{shape}}$ in the proposed RAMED model.

ECG(A)	AUROC	AUPRC	F1
w/o $\mathcal{L}_{\text{shape}}$	0.6982	0.5430	0.5227
full model	**0.7358**	**0.5714**	**0.5427**

Yahoo's S5	AUROC	AUPRC	F1
w/o $\mathcal{L}_{\text{shape}}$	0.8749	0.4550	0.4225
full model	**0.8878**	**0.4599**	**0.4242**

Sensitivity to Hyperparameters In the proposed model, there are three important hyperparameters: (i) coarse-to-fine fusion weight β in Equation (6), (ii) tradeoff parameter λ on the multiresolution shape-forcing loss in Equation (11), and (iii) the factor τ used in Equation (4) which determines each decoder's decoding length. Tables 5, 6 and 7 show sensitivity analysis on them.

As can be seen in Table 5, β is relatively stable. When β is set to be 0.1, our model can achieve a good performance in terms of all three metrics. Empirically, to use more coarse-grained information to help temporal modeling at a high-resolution level, β can be set to be a smaller value such as 0.1. A larger β will ignore coarse-grained information and may degrade performance.

Table 5: Sensitivity analysis for β on the *ECG(A)* data set where β is a weight for coarse-to-fine fusion.

β	0.1	0.3	0.5	0.7	0.9
AUROC	0.6853	0.6779	0.6931	0.7052	0.6191
AUPRC	0.5528	0.5325	0.5419	0.5502	0.5293
F1	0.5385	0.5203	0.5244	0.5282	0.5474

In Table 5 and Table 7, we can find that both the hyperparameters λ and τ are relatively stable.

Discussions

There are two recurrent structures related to our RAMED: hierarchical RNN (Hermans and Schrauwen 2013) and pyramid RNN (Qianli et al. 2020).

RAMED vs Hierarchical RNN The hierarchical RNN is a stacked RNN with multiple recurrent layers, with each layer receiving hidden states from the previous layer as input. It presents a natural strategy to model multiresolution temporal information in time series. However, this is different from RAMED. First, RAMED is a recurrent network ensemble with multiple decoders. We build connections across decoders in a coarse-to-fine manner. In this way, a decoder learned at a lower-resolution can help provide global information for decoders at higher resolutions. Each decoder in RAMED can be used as any existing hierarchical RNNs.

RAMED vs Pyramid RNN The recent pyramid RNN (Qianli et al. 2020) is also similar to RAMED, as it considers aggregating multiresolution information from each recurrent layer. However, (i) RAMED organizes multiple decoders in a top-down manner, while pyramid RNN is a hierarchical RNN and is built bottom-up; 2) The pyramid RNN cannot be adapted to time series reconstruction/generation problems.

Table 6: Sensitivity analysis for λ on the *ECG(A)* data set where λ is a weight for multiresolution shape-forcing loss.

λ	0.001	0.01	0.1	1	10
AUROC	0.7162	0.6975	0.6741	0.7075	0.7031
AUPRC	0.5543	0.5356	0.5455	0.5439	0.5437
F1	0.5353	0.5204	0.5347	0.5266	0.5318

Table 7: Sensitivity analysis for τ on the *ECG(A)* data set where τ is related to each decoder's decoding length.

τ	1	2	3	4	5
AUROC	0.6320	0.7494	0.6795	0.7358	0.6947
AUPRC	0.5145	0.5519	0.5476	0.5714	0.5368
F1	0.5220	0.5266	0.5385	0.5427	0.5301

Conclusion

In this paper, we introduce a recurrent ensemble network called Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED) for time series anomaly detection. RAMED is based on a multiresolution shape-forcing loss and a new coarse-to-fine fusion mechanism. By introducing the multiresolution shape-forcing loss, decoder individuals can be encouraged to capture temporal dynamics at different resolution levels. With the coarse-to-fine fusion mechanism, all of the decoders can be well integrated into our ensemble framework. Then, the decoder at the highest resolution level can reconstruct the raw input time series with a better accuracy. Finally, comparisons on various time series benchmarks demonstrate that the proposed model achieves better performance than competitive baselines.

Acknowledgments

We thank the anonymous reviewers for their helpful feedbacks. The work described in this paper was partially funded by the Foreign Science and Technology Cooperation Program of Huangpu District of Guangzhou (No. 2018GH09, 2019-2020), the National Natural Science Foundation of China (Grant Nos. 61502174, 61872148), the Natural Science Foundation of Guangdong Province (Grant Nos. 2017A030313355, 2019A1515010768), the Guangzhou Science and Technology Planning Project (Grant Nos. 201704030051, 201902010020), the Key R&D Program of Guangdong Province (No. 2018B010107002), and the Fundamental Research Funds for the Central Universities.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Chen, P.; Liu, S.; Shi, C.; Hooi, B.; and Cheng, X. 2018. NeuCast: seasonal neural forecast of power grid time series. In *IJCAI*.

Chia, C.-C.; and Syed, Z. 2014. Scalable noise mining in long-term electrocardiographic time-series to predict death following heart attacks. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Chung, J.; Ahn, S.; and Bengio, Y. 2017. Hierarchical multiscale recurrent neural networks. In *ICLRs*.

Cook, A. A.; Misirli, G.; and Fan, Z. 2020. Anomaly detection for IoT time-series data: a survey. *IEEE Internet of Things Journal* 7(7): 6481–6494.

Cuturi, M.; and Blondel, M. 2017. Soft-DTW: a differentiable loss function for time-weries. In *ICML*.

Dietterich, T. G. 2000. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*.

Ding, Z.; Yang, B.; Chi, Y.; and Guo, L. 2016. Enabling smart transportation systems: a parallel spatio-temporal database approach. *IEEE Transactions on Computers* 65(5): 1377–1391.

Filonov, P.; Lavrentyev, A.; and Vorontsov, A. 2016. Multivariate industrial time series with cyber-attack simulation: fault detection using an lstm-based predictive data model. *Preprint arXiv* .

Gupta, M.; Gao, J.; Aggarwal, C. C.; and Han, J. 2014. Outlier detection for temporal data: a survey. *IEEE Transactions on Knowledge and Data Engineering* 26(9): 2250–2267.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In *ICCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hermans, M.; and Schrauwen, B. 2013. Training and analysing deep recurrent neural networks. In *Advances in Neural Information Processing Systems*.

Hihi, S. E.; and Bengio, Y. 1996. Hierarchical recurrent neural networks for long-term dependencies. In *Advances in Neural Information Processing Systems*.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8): 1735–1780.

Kieu, T.; Yang, B.; Guo, C.; and Jensen, C. S. 2019. Outlier detection for time series with recurrent autoencoder ensembles. In *IJCAI*.

Kieu, T.; Yang, B.; and Jensen, C. S. 2018. Outlier detection for multidimensional time series using deep neural networks. In *IEEE International Conference on Mobile Data Management (MDM)*.

Kingma, D. P.; and Ba, J. L. 2015. Adam: a method for stochastic optimization. In *ICLR*.

Li, L.; Yan, J.; Wang, H.; and Jin, Y. 2020. Anomaly detection of time series with smoothness-inducing sequential variational auto-encoder. *IEEE Transactions on Neural Networks* 1–15.

Liu, Y.; Yu, R.; Zheng, S.; Zhan, E.; and Yue, Y. 2019. NAOMI: non-autoregressive multiresolution sequence imputation. In *Advances in Neural Information Processing Systems*.

Malhotra, P.; Ramakrishnan, A.; Anand, G.; Vig, L.; Agarwal, P.; and Shroff, G. M. 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. *Preprint arXiv* .

Qianli, M.; Zhenxi, L.; Enhuan, C.; and Garrison, C. 2020. Temporal pyramid recurrent neural network. In *AAAI*.

Ren, H.; Xu, B.; Wang, Y.; Yi, C.; Huang, C.; Kou, X.; Xing, T.; Yang, M.; Tong, J.; and Zhang, Q. 2019. Time-series anomaly detection service at microsoft. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Ruff, L.; Vandermeulen, R.; Goernitz, N.; Deecke, L.; Siddiqui, S. A.; Binder, A.; Müller, E.; and Kloft, M. 2018. Deep one-class classification. In *ICML*.

Sakoe, H.; and Chiba, S. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26(1): 159–165.

Su, Y.; Zhao, Y.; Niu, C.; Liu, R.; Sun, W.; and Pei, D. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P.-A. 2008. Extracting and composing robust features with denoising autoencoders. In *ICML*.

Wang, S.; Zeng, Y.; Liu, X.; Zhu, E.; Yin, J.; Xu, C.; and Kloft, M. 2019. Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. In *Advances in Neural Information Processing Systems*.

Wold, H. 1938. *A study in the analysis of stationary time series*. Ph.D. thesis, Almqvist & Wiksell.

Yoo, Y.; Kim, U.; and Kim, J. 2019. Recurrent reconstructive network for sequential anomaly detection. *IEEE Transactions on Cybernetics* 1–12.

Yu, Q.; Jibin, L.; and Jiang, L. 2016. An improved ARIMA-based traffic anomaly detection algorithm for wireless sensor networks. *International Journal of Distributed Sensor Networks* 2016: 1–9.

Zhang, C.; Song, D.; Chen, Y.; Feng, X.; Lumezanu, C.; Cheng, W.; Ni, J.; Zong, B.; Chen, H.; and Chawla, N. 2019.

A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *AAAI*.

Zhou, B.; Liu, S.; Hooi, B.; Cheng, X.; and Ye, J. 2019. BeatGAN: anomalous rhythm detection using adversarially generated time series. In *IJCAI*.