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 is a popular model for time series anomaly detection, in which outliers or abnormal segments are identified by their high reconstruction errors. However, existing recurrent autoencoders can easily suffer from overfitting and error accumulation due to sequential decoding. In this paper, we propose a simple yet efficient recurrent network ensemble called Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED). By using decoders with different decoding lengths and a new coarse-to-fine fusion mechanism, lower-resolution information can help long-range decoding for decoders with higher-resolution outputs. A multiresolution shape-forcing loss is further introduced to encourage decoders' outputs at multiple resolutions to match the input's global temporal shape. Finally, the output from the decoder with the highest resolution is used to obtain an anomaly score at each time step. Extensive empirical studies on real-world benchmark data sets demonstrate that the proposed RAMED model outperforms recent strong baselines on time series anomaly detection.

Introduction

Anomaly detection aims to identify anomalous patterns from data. In particular, time series anomaly detection has received a lot of 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 an 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 roughly categorized as either predictive or reconstruction-based. Classical predictive models include the autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) models (Wold 1938). They are linear regressors and use the prediction error as anomaly score. More recently, recurrent neural networks (RNNs) (Zhang et al. 2019) and other deep predictors (Filonov, Lavrentyev, and Vorontsov 2016) are also used. However, these methods depend largely on the models' extrapolation capacities (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 use this to reconstruct the time series. Points or segments that cannot be well reconstructed are considered as outliers. Reconstruction methods are popular in applications such as anomalous rhythm detection (Zhou et al. 2019) and network traffic monitoring (Kieu, Yang, and Jensen 2018). In this paper, we focus on reconstruction-based methods.

In deep learning, the recurrent auto-encoder (RAE) (Malhotra et al. 2016) has demonstrated good performance in time series anomaly detection. Following the well-known sequence-to-sequence framework (Sutskever, Vinyals, and Le 2014), the RAE consists of an encoder and a decoder. The reconstruction error at each time step is used as an anomaly score. Very recently, other autoencoder variants have also been proposed. For example, Yoo, Kim, and Kim (2019) developed the recurrent reconstructive network (RRN), which uses self-attention and feedback transition to help capture the temporal dynamics. Kieu et al. (2019) proposed the recurrent autoencoder ensemble based on ensemble learning (Dietterich 2000). Both the encoders and decoders consist of several RNNs with sparse skip connections. On inference, the median reconstruction error from all decoders is used as the anomaly score. However, the RAE and its variants can have difficulties in decoding long time series due to error accumulation from previous time steps.

In this paper, we propose the Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED). Inspired by (Kieu et al. 2019), RAMED also has an ensemble of decoders. However, the difference is that the proposed decoders capture the 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 macro temporal characteristics such as trend patterns and seasonality; whereas a decoder with long decoding length can capture more detailed local temporal patterns. Furthermore, instead of simply averaging the decoder outputs as in (Kieu et al. 2019), the lower-resolution temporal information is used to guide decoding at a higher resolution. Specifically, we introduce a multiresolution shape-forcing loss to encourage the decoders to match the input's global temporal shape at multiple resolutions. This avoids overfitting the nonlinear local patterns at a higher resolution, and alleviates error accumulation during decoding. Finally, the output from the highest resolution (whose decoding length equals the length of the whole time series) is used as the ensemble output.

Our main contributions can be summarized as follows:

- We present the novel recurrent autoencoder RAMED, with multiple decoders of different decoding lengths. By introducing a shape-forcing reconstruction loss, decoders can capture temporal characteristics of the 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

Autoencoders for Time Series Anomaly Detection

The sequence-to-sequence model (Sutskever, Vinyals, and Le 2014) is a popular auto-encoding approach for sequential data. There are two steps in its learning procedure: (i) encoding, which compresses the sequential data into a fixed-length representation; and (ii) decoding, which reconstructs the original input from the learned compressed representation. The sequence-to-sequence model has been widely used in natural language processing (Bahdanau, Cho, and Bengio 2015). Recently, it is also used in time series applications such as prediction (Le Guen and Thome 2019), clustering (Ma et al. 2019) and anomaly detection (Malhotra et al. 2016; Yoo, Kim, and Kim 2019; Kieu et al. 2019). Two recent representative sequence-to-sequence models for time series anomaly detection are the recurrent auto-encoder (RAE) (Malhotra et al. 2016) and recurrent reconstructive network (RRN) (Yoo, Kim, and Kim 2019).

Time Series Encoding The recurrent neural network (RNN) is often used to encode time series data. Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$, where $\mathbf{x}_t \in \mathbb{R}^d$, be a time series of length T. At time t, the encoder's hidden state $\mathbf{h}_t^{(E)}$ is updated as:

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

where $f^{(E)}$ is a nonlinear function. The $\mathbf{h}_T^{(E)}$ at the last time step T is then used as $\mathbf{X}$'s compressed representation.

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

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

where coefficients w_1, w_2 are randomly sampled from $\{(1,0),(0,1),(1,1)\}$ at each time step. In (Kieu et al. 2019), the skip length s is randomly sampled from [1, 10] and fixed before training.

Decoding for Anomaly Detection The encoder's compressed representation $\mathbf{h}_T^{(E)}$ can be decoded by using a LSTM. In time series anomaly detection, decoding is usually easier when performed in time-reverse order (Kieu et al. 2019; Yoo, Kim, and Kim 2019), i.e., the target reconstructed output for input $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$ is $[\mathbf{y}_T, \mathbf{y}_{T-1}, \ldots, \mathbf{y}_1]$, where $\mathbf{y}_t$ is the LSTM's output at time t (for $t = T, T-1, \ldots, 1$). After initializing $\mathbf{h}_T$ by $\mathbf{h}_T^{(E)}$, $\{\mathbf{h}_{T-1}, \ldots, \mathbf{h}_1\}$ are obtained as:

$$\begin{aligned} \mathbf{y}_t &= \mathbf{W}\mathbf{h}_t + \mathbf{b}, \\ \mathbf{h}_{t-1} &= \mathrm{LSTM}([\mathbf{y}_t; \mathbf{h}_t]), \end{aligned} \tag{3}$$

where $\mathbf{W}, \mathbf{b}$ are learnable parameters. An anomaly score is computed for each $\mathbf{x}_t$ based on the error $\mathbf{e}(t) = \mathbf{y}_t - \mathbf{x}_t$. As can be seen from (3), this error can accumulate during the sequential decoding process.

Encoder-Decoder Ensemble In the recurrent autoencoder ensemble (RAE-ensemble) (Kieu et al. 2019), multiple recurrent encoders and decoders are used. Let $L^{(E)}$ be the number of encoders, and $\mathbf{h}_T^{(E_i)}$ be the representation from the ith encoder. The integrated compressed representation $\mathbf{h}^{(E)}$ is obtained by

$$\mathbf{h}^{(E)} = F_{\mathrm{MLP}}\Big(\mathrm{concat}[\mathbf{h}_T^{(E_1)}; \ldots; \mathbf{h}_T^{(E_i)}; \ldots; \mathbf{h}_T^{(E_{L^{(E)}})}]\Big), \tag{4}$$

where F_{MLP} is a fully-connected layer, and $\mathbf{h}^{(E)}$ shares the same dimension as each $\mathbf{h}_T^{(E_i)}$. During decoding, $L^{(D)}$ decoders are used, with each of which following the same recurrent decoding process. After initializing $\mathbf{h}_T^{(k)}$ to $\mathbf{h}^{(E)}$, the kth decoder $\mathcal{D}^{(k)}$ outputs $\{\mathbf{y}_T^{(k)}, \ldots, \mathbf{y}_1^{(k)}\}$ and $\{\mathbf{h}_{T-1}, \ldots, \mathbf{h}_1\}$ as:

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

where $\mathbf{W}^{(k)}$, and $\mathbf{b}^{(k)}$ are learnable parameters. Note that this also suffers from error accumulation as in (3). During inference, outputs from all the decoders are pooled together.

Multiresolution Temporal Modeling

To capture multiresolution temporal information, Hihi and Bengio (1996) developed a hierarchical RNN that integrates multiple delays and time scales in different recurrent neurons. Similarly, to model multiscale structures in text. Hermans and Schrauwen (2013); Chung, Ahn, and Bengio (2017) introduced the hierarchical multiscale RNN. This

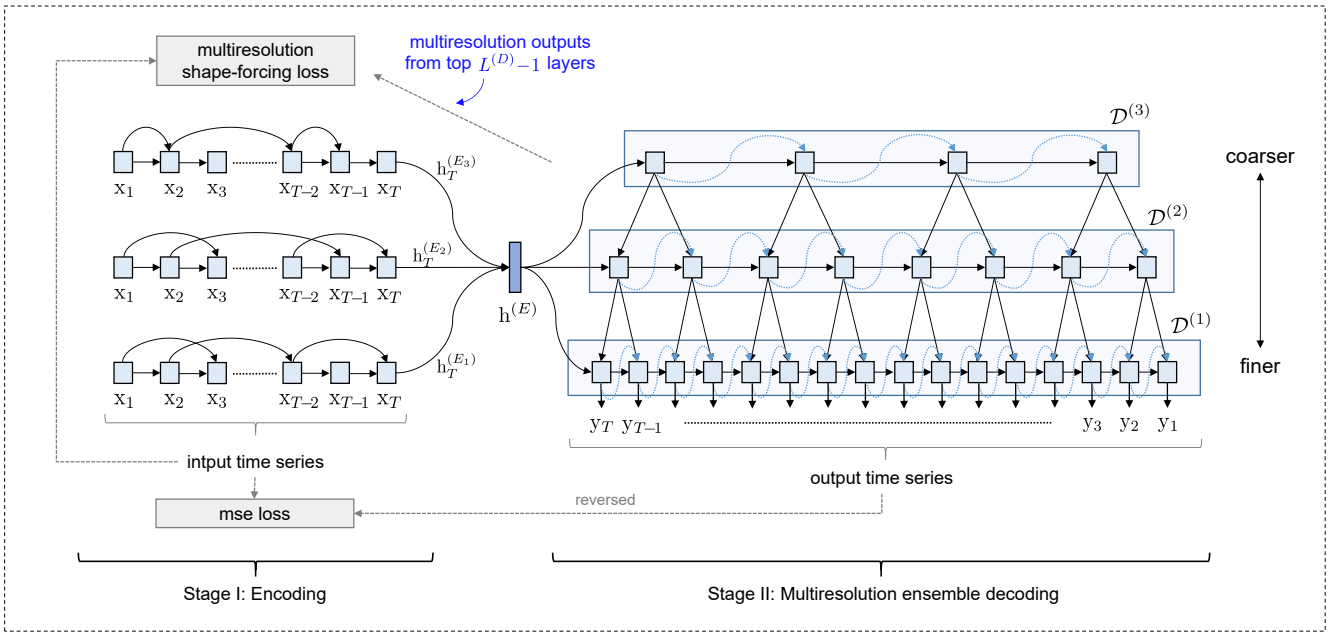


Figure 1: The proposed Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED).

stacks multiple recurrent layers, with each layer receiving hidden states from the previous layer as input. Instead of simply stacking multiple recurrent layers, Liu et al. (2019) proposed a coarse-to-fine procedure for time series imputation. Very recently, the pyramid RNN (Ma et al. 2020) aggregates the multiresolution information from each recurrent layer in a bottom-up manner.

Proposed Architecture

In this paper, we utilize multiresolution temporal information by integrating with a coarse-to-fine decoding process. Figure 1 shows the proposed model (with $L^{(E)} = L^{(D)} = 3$), which will be called Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED).

Multiresolution Ensemble Decoding

As in (Kieu et al. 2019), RAMED uses an ensemble of $L^{(E)}$ RNN encoders with the encoding process in (4). At the lowest resolution layer, macro temporal characteristics of the time series are captured. This is then passed to the next layer (with a higher decoding resolution), and so on.

Subsequently, multiple reconstructions are obtained by running $L^{(D)}$ recurrent decoders on the compressed representation $\mathbf{h}^{(E)}$. To encourage different decoders to capture temporal behaviors of the time series at different resolutions, we use different numbers of decoding steps for the decoders. A decoder with short decoding length has to focus on the macro temporal characteristics; whereas a decoder with long decoding length can capture more detailed local temporal patterns. A multiresolution fusion strategy is used to efficiently fuse the decoder outputs in a coarse-to-fine manner.

Moreover, the decoded output is encouraged to be similar to the input time series by using a differentiable shape-forcing loss based on dynamic time warping (DTW) (Sakoe and Chiba 1978).

The following sections describe these various components in more detail.

Decoder Lengths The kth decoder $\mathcal{D}^{(k)}$ reconstructs a time series of length $T^{(k)}$, where $T^{(k)} = \alpha_k T$ and

$$\alpha_k = 1/\tau^{k-1} \in (0, 1] \tag{6}$$

for some $\tau > 1$ ($\tau = 2$ in Figure 1). Note that $\alpha_1 = 1$ and $T^{(1)} = T$. We require $T^{(L^{(D)})} \geq 2$, so that the decoder at the top takes at least two decoding steps.

To improve robustness, as in the denoising autoencoder (Vincent et al. 2008), we add a small amount of noise $\epsilon\boldsymbol{\delta}$ to the LSTM's input, where ϵ is a small scalar (10^{-4} in the experiments), 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, they cannot be summarized to an ensemble output by simply using the average or median as in (Kieu et al. 2019). In the following, we propose a simple yet efficient multiresolution coarse-to-fine strategy to fuse the coarser-grained decoder with the finer-grained decoders.

Consider two decoders $\mathcal{D}^{(k+1)}$ and $\mathcal{D}^{(k)}$. Note from (6) 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 other words, the decoder at the top ($k = L^{(D)}$), with output $\{\mathbf{h}_1^{(L^{(D)})}, \ldots, \mathbf{h}_{T^{(D)}}^{(L^{(D)})}\}$ obtained via (5), is the coarsest among all decoders.

For the other decoders $\mathcal{D}^{(k)}$s ($k = L^{(D)} - 1, \ldots, 1$), instead of using (5), it first combines its previous hidden

state $\mathbf{h}_{t+1}^{(k)}$ with the corresponding slightly-coarser information $\mathbf{h}_{\lceil t/\tau \rceil}^{(k+1)}$ from the sibling decoder $\mathcal{D}^{(k+1)}$ as:

$$\hat{\mathbf{h}}_t^{(k)} = \beta \mathbf{h}_{t+1}^{(k)} + (1-\beta) F'_{\mathrm{MLP}}\left(\mathrm{concat}[\mathbf{h}_{t+1}^{(k)}; \mathbf{h}_{\lceil t/\tau \rceil}^{(k+1)}]\right), \quad (7)$$

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

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

for $t = T^{(k)} - 1, \ldots, 1$.

Finally, the ensemble's reconstructed output can be obtained from the bottom-most decoder as $\mathbf{Y}_{\mathrm{recon}} = [\mathbf{y}_1^{(1)}, \mathbf{y}_2^{(1)}, \ldots, \mathbf{y}_T^{(1)}]$ (after reversing to the original time order). To encourage $\mathbf{Y}_{\mathrm{recon}}$ to be close to input $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T]$, we use the square loss as the reconstruction error:

$$\mathcal{L}_{\mathrm{MSE}}(\mathbf{X}) = \sum_{t=1}^{T} \|\mathbf{y}_t^{(1)} - \mathbf{x}_t\|^2. \quad (8)$$

Multiresolution Shape-Forcing Loss To further encourage decoders to learn consistent temporal patterns at different resolutions, we force the decoders' outputs to have similar shapes as the original input by introducing a loss based on dynamic time warping (DTW) (Sakoe and Chiba 1978).

Let the output from decoder $\mathcal{D}^{(k)}$ be $\mathbf{Y}^{(k)} = [\mathbf{y}_1^{(k)}, \mathbf{y}_2^{(k)}, \ldots, \mathbf{y}_{T^{(k)}}^{(k)}]$. Since $T^{(k)} \neq T$ for $k = 2, \ldots, L^{(D)}$, we define a similarity between time series $\mathbf{X}$ and each $\mathbf{Y}^{(k)}$ by DTW. 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 $\mathbf{A}_{1,1} = 1$ 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 alignment costs are 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 $\mathbf{Y}^{(k)}$ is then:

$$\mathrm{DTW}(\mathbf{X}, \mathbf{Y}^{(k)}) = \min_{\mathbf{A} \in \mathcal{A}} \langle \mathbf{A}, \mathbf{C} \rangle. \quad (9)$$

where $\mathcal{A}$ is the set of $T \times T^{(k)}$ binary alignment matrices, and $\langle \cdot, \cdot \rangle$ is the matrix inner product. The DTW distance is non-differentiable due to the min operator. To integrate DTW into end-to-end training, we replace (9) by the smoothed DTW (sDTW) distance (Cuturi and Blondel 2017):

$$\mathrm{sDTW}(\mathbf{X}, \mathbf{Y}^{(k)}) = -\gamma \log \left(\sum_{\mathbf{A} \in \mathcal{A}} e^{-\langle \mathbf{A}, \mathbf{C} \rangle / \gamma} \right), \quad (10)$$

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

Algorithm 1 Recurrent Autoencoder with Multiresolution Ensemble Decoding (RAMED).

Input: a set of time series $\{\mathbf{X}_b\}$; batch size B; number of encoders $L^{(E)}$; number of decoders $L^{(D)}$; τ.

```
for each decoder, its decoding length is T^(k) = α_k T;
repeat
    sample a batch of B time series;
    for b = 1, ..., B do
        feed time series X_b to encoders and obtain the last hidden states {h_{b,T}^(E)};
        obtain joint representations {h^(E)} via (4);
        for k = L^(D), L^(D)−1 ..., 1 do
            run the decoder D^(k);
            if (k ≠ L^(D)) then
                perform coarse-to-fine fusion;
            end if
            obtain updated hidden states {h_{b,t}^(k)} and outputs {y_{b,t}^(k)};
        end for
        minimize (12) by SGD or its variants;
    end for
until convergence.
```

With the sDTW distance, we encourage decoders at different resolutions to output time series with similar temporal characteristics as the input. Here, decoders whose decoding length is less than the length of the whole time series are considered. This leads to the following multiresolution shape-forcing loss:

$$\mathcal{L}_{\mathrm{shape}}(\mathbf{X}) = \frac{1}{L^{(D)} - 1} \sum_{k=2}^{L^{(D)}} \mathrm{sDTW}(\mathbf{X}, \mathbf{Y}^{(k)}). \quad (11)$$

Given a batch of samples $\{\mathbf{X}_b\}_{b=1,2,\ldots,B}$ (where B is the batch size), the total loss is:

$$\mathcal{L} = \frac{1}{B} \sum_{b=1}^{B} \left(\mathcal{L}_{\mathrm{MSE}}(\mathbf{X}_b) + \lambda \mathcal{L}_{\mathrm{shape}}(\mathbf{X}_b) \right), \quad (12)$$

where λ is a trade-off parameter. This can be minimized by stochastic gradient descent or its variants (such as Adam (Kingma and Ba 2015)). The training procedure is shown in Algorithm 1.

Anomaly Score and Detection

Given a time series $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$ and its reconstruction $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T]$, the reconstruction error at time t is $\mathbf{e}(t) = \mathbf{y}_t - \mathbf{x}_t$. We then fit a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ using the set of $\{\mathbf{e}(t)\}$ from all time steps and all time series in the validation set.

On inference, the probability that $\mathbf{x}_t$ from an unseen time series in the test set is anomalous is defined as:

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

Dataset	T	# Training	# Validation	# Testing	% Anomaly
ECG					
(A) chfdb_chf01_275	64	40	17	59	14.61
(B) chfdb_chf13_45590	64	53	22	40	12.35
(C) chfdbchf15	64	237	101	104	4.45
(D) ltstdb_20221_43	64	57	24	35	11.51
(E) ltstdb_20321_240	64	43	18	45	9.61
(F) mitdb_100_180	64	64	27	70	8.38
2D-gesture	64	91	39	47	24.63
Power-demand	512	25	11	29	11.44
Yahoo's S5	128	659	398	394	3.20

Table 1: Statistics of the time series data sets.

Thus, we can take

$$s(t) = (\mathbf{e}(t) - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{e}(t) - \boldsymbol{\mu}) \tag{13}$$

as $\mathbf{x}_t$'s 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 the following nine commonly-used real-world time series benchmarks[1] (Table 1):

1. *ECG*: This is a collection of 6 data sets on the detection of anomalous beats from electrocardiograms readings.
2. *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.
3. *Power-demand*: This contains one year of power consumption records measured by a Dutch research facility in 1997.
4. *Yahoo's S5 Webscope*: This contains records from real production traffic of the Yahoo website. Anomalies are manually labeled by human experts.

ECG and *2D-gesture* are bivariate time series ($d = 2$), while *Power-demand* and *Yahoo's S5* are univariate ($d = 1$). For each of *ECG*, *2D-gesture* and *Power-demand*, the public data set includes a training set (containing only normal data) and a test set. We use 30% of the training set for validation, and the rest for actual training. The model with the lowest reconstruction loss on the validation set is selected for evaluation. For *Yahoo's S5*, the available data set is split into three parts: with 40% of the samples for training, another 30% for validation, and the remaining 30% for testing. The training set contains unknown anomalies, and we use the model with the highest AUROC value on the validation set for evaluation.

[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/.

The time series are partitioned into length-T sequences by using a sliding window. The sliding window has a stride of 32 on the *ECG* data sets, 512 on *Power-demand*, and 64 on *2D-gesture* and *Yahoo's S5*. Table 1 shows the sequence length T, number of sequences in the training/validation/testing set, and percentage of anomalous samples in the test set.

Baselines The proposed RAMED model is compared with four recent anomaly detection baselines:[2] (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) (Goodfellow et al. 2014) for time series anomaly detection.

Evaluation Metrics Performance measures such as precision and recall depend on thresholding the anomaly score. To avoid setting this threshold, we use the following metrics which have been widely used in anomaly detection (Wang et al. 2019; Ren et al. 2019; Li et al. 2020; Su et al. 2019): (i) area under the ROC curve (AUROC), (ii) area under the precision-recall curve (AUPRC), and (iii) the highest F1-score (denoted $\mathrm{F1}_{best}$) (Li et al. 2020; Su et al. 2019), which is selected from using 1000 thresholds uniformly distributed from 0 to the maximum anomaly score over all time steps in the test set (Yoo, Kim, and Kim 2019).

Implementation Details We use 3 encoders and 3 decoders. Each encoder and decoder is a single-layer LSTM with 64 units. We perform grid search on the hyperparameter β in (7) from $\{0.1, 0.2, \ldots, 0.9\}$, λ in (12) from $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$, τ in (6) is set to 3 and γ in (10) is set to 0.1. The Adam optimizer (Kingma and Ba 2015) is used with an initial learning rate of 10^{-3}.

[2] RAE and RRN are downloaded from https://github.com/YongHoYoo/AnomalyDetection, BeatGAN is from https://github.com/Vniex/BeatGAN, and RAE-ensemble is from https://github.com/tungk/OED

metric	*method*	*ECG*						*2D-gesture*	*Power-demand*	*Yahoo's S5*	*avg rank*
		A	B	C	D	E	F				
AUROC	BeatGAN	0.6566	0.7056	0.7329	0.6173	0.8160	0.4223	0.7256	0.5796	0.8728	4.33
	RAE	0.6728	0.7502	0.8289	0.5452	0.7970	0.4715	0.7601	0.6122	0.8823	3.78
	RRN	0.6393	0.7623	0.7405	0.6318	0.8101	0.4531	0.7530	0.6607	0.8869	3.44
	RAE-ensemble	0.6884	0.7788	0.8570	0.6400	0.8035	0.5213	0.7808	0.6587	0.8850	2.44
	RAMED	**0.7127**	**0.8551**	**0.8736**	**0.6473**	**0.8828**	**0.6399**	**0.7839**	**0.6787**	**0.8942**	**1.00**
AUPRC	BeatGAN	0.5197	0.4101	0.2254	0.1613	0.3342	0.0778	0.4952	0.1228	0.4702	4.44
	RAE	0.5501	0.4249	0.4996	0.1435	0.2126	0.0894	0.4979	0.1350	0.4782	3.78
	RRN	0.5260	0.5653	0.4139	0.1652	0.3206	0.0833	0.4866	0.1446	0.4794	3.22
	RAE-ensemble	0.5549	0.4769	0.5256	0.2026	0.2798	0.0948	0.5287	0.1400	0.4783	2.56
	RAMED	**0.5803**	**0.7008**	**0.5486**	**0.2203**	**0.3784**	**0.1253**	**0.5331**	**0.1627**	**0.4809**	**1.00**
$\mathrm{F1}_{best}$	BeatGAN	0.5102	0.4204	0.2931	0.2502	0.4776	0.1562	0.4941	0.2266	0.4484	4.44
	RAE	0.5478	0.4736	0.5046	0.2193	0.3886	0.1581	0.5300	0.2798	0.4473	3.78
	RRN	0.5440	0.5502	0.4537	0.2621	0.4548	0.1562	0.5240	0.2926	**0.4502**	3.00
	RAE-ensemble	0.5479	0.5016	0.5333	0.2735	0.3910	0.1602	0.5511	0.2678	0.4497	2.67
	RAMED	**0.5762**	**0.6871**	**0.5541**	**0.3466**	**0.4855**	**0.2090**	**0.5633**	**0.2934**	**0.4502**	**1.00**

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

method	*ECG(A)*			*ECG(B)*			*ECG(C)*		
	AUROC	AUPRC	$\mathrm{F1}_{best}$	AUROC	AUPRC	$\mathrm{F1}_{best}$	AUROC	AUPRC	$\mathrm{F1}_{best}$
w/o coarse-to-fine fusion	0.6781	0.5305	0.5275	0.8196	0.6144	0.5539	0.7537	0.4700	0.5263
w/o $\mathcal{L}_{\mathrm{shape}}$	0.6916	0.5720	0.5556	0.8542	0.6362	0.6266	0.8450	0.4828	0.5365
full model	**0.7127**	**0.5803**	**0.5762**	**0.8551**	**0.7008**	**0.6871**	**0.8736**	**0.5486**	**0.5541**
method	*ECG(D)*			*ECG(E)*			*ECG(F)*		
	AUROC	AUPRC	$\mathrm{F1}_{best}$	AUROC	AUPRC	$\mathrm{F1}_{best}$	AUROC	AUPRC	$\mathrm{F1}_{best}$
w/o coarse-to-fine fusion	0.5125	0.1326	0.2092	0.8212	0.3058	0.3886	0.5083	0.0875	0.1593
w/o $\mathcal{L}_{\mathrm{shape}}$	0.5609	0.1742	0.2537	0.8455	0.3228	0.4235	0.5598	0.0993	0.1704
full model	**0.6473**	**0.2203**	**0.3466**	**0.8828**	**0.3784**	**0.4855**	**0.6399**	**0.1253**	**0.2090**
method	*2D-gesture*			*Power-demand*			*Yahoo's S5*		
	AUROC	AUPRC	$\mathrm{F1}_{best}$	AUROC	AUPRC	$\mathrm{F1}_{best}$	AUROC	AUPRC	$\mathrm{F1}_{best}$
w/o coarse-to-fine fusion	0.7656	0.5292	0.5448	0.6357	0.1385	0.2642	0.8846	0.4501	0.4335
w/o $\mathcal{L}_{\mathrm{shape}}$	0.7716	0.5328	0.5525	0.6632	0.1500	0.2805	0.8895	0.4783	0.4497
full model	**0.7839**	**0.5331**	**0.5633**	**0.6787**	**0.1627**	**0.2934**	**0.8942**	**0.4809**	**0.4502**

Table 3: Effectiveness of coarse-to-fine fusion and multiresolution shape-forcing loss in RAMED.

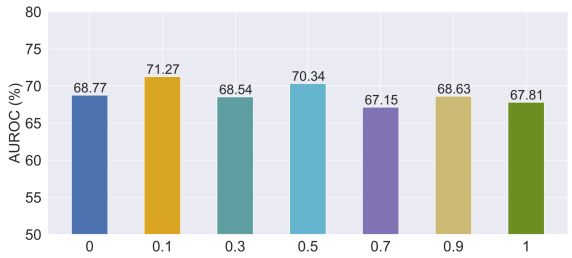
Performance Comparison

Results are shown in Table 2. In terms of the average ranking, RAE-ensemble has better performance among the 4 baselines. This agrees with the general view that ensemble learning is beneficial. The proposed RAMED consistently outperforms RAE-ensemble on all three metrics, and the improvement on the AUROC is particularly large. This demonstrates that using multiresolution information in an ensemble further helps time series reconstruction, such that lower false positive rates and higher true negative rates can be achieved.

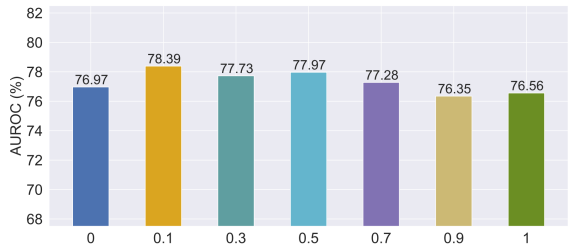
Ablation Study

In this section, we examine the contributions of the coarse-to-fine fusion strategy and multiresolution shape-forcing loss in the proposed RAMED model. Sensitivity analysis on the hyperparameters is also performed.

Effect of Coarse-to-Fine Fusion In this experiment, we still use 3 encoders as in the full model, but only use one decoder (with decoding length equal to the input length). As can be seen from Table 3, when only one decoder is left to

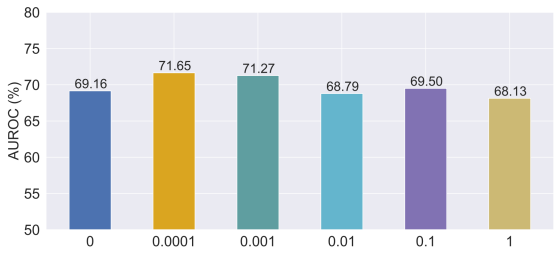


(a) *ECG(A)*.

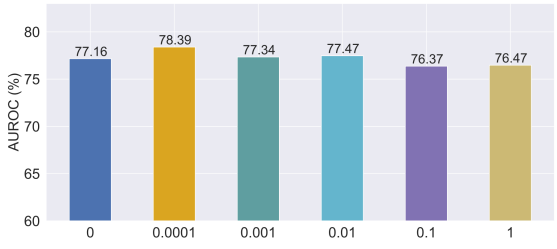


(b) *Gesture*.

Figure 2: Effect of varying β.



(a) *ECG(A)*.



(b) *Gesture*.

Figure 3: Effect of varying λ.

ECG(A)	$T^{(k)}$'s	AUROC	AUPRC	F1_{best}
$L^{(D)}=1$	64	0.6781	0.5305	0.5275
$L^{(D)}=2$	21, 64	0.6976	0.5828	0.5708
$L^{(D)}=3$	7, 21, 64	0.7172	0.5803	0.5762
$L^{(D)}=4$	2, 7, 21, 64	0.6743	0.5132	0.5196

Gesture	$T^{(k)}$'s	AUROC	AUPRC	F1_{best}
$L^{(D)}=1$	64	0.7656	0.5292	0.5448
$L^{(D)}=2$	21, 64	0.7693	0.5080	0.5462
$L^{(D)}=3$	7, 21, 64	0.7839	0.5331	0.5633
$L^{(D)}=4$	2, 7, 21, 64	0.7779	0.5153	0.5545

Table 4: Effect of varying $L^{(D)}$ on *ECG(A)* and *Gesture*. $T^{(k)}$'s are the decoding lengths of the various decoders.

reconstruct the input time series, performance on all metrics decrease. This verifies usefulness of the multiresolution fusion strategy.

Effect of Multiresolution Shape-Forcing Loss In this experiment, we remove the multiresolution shape-forcing loss from (12), and only minimize the reconstruction error. As can be seen from Table 3, the shape-forcing loss also plays an important role in multiresolution decoding.

Sensitivity to Hyperparameters We study the following hyperparameters in the proposed model: (i) coarse-to-fine fusion weight β in (7), (ii) tradeoff parameter λ on the multiresolution shape-forcing loss in (12), and (iii) $L^{(D)}$, the number of decoders. The default hyperparameter settings are $\beta = 0.1, \lambda = 10^{-4}$ and $L^{(D)} = 3$. Experiments are performed on the *ECG(A)* and *Gesture* data sets.

Figure 2 shows the AUROC's at different β's. As can be seen, the performance w.r.t. β is relatively stable. When β is set to 0.1, the proposed model achieves the best performance. This is because when β is small, more coarse-grained information can be used to help temporal modeling at a higher-resolution levels; whereas a larger β may ignore the coarse-grained information and degrades performance.

Figure 3 shows the AUROC's at different λ's. As can be seen, when λ is small (10^{-4}), but nonzero, better performance is achieved.

Table 4 shows the AUROC's with different numbers of decoders. As can be seen, increasing $L^{(D)}$ (from 1 to 3) can improve performance as more abundant multiresolution temporal patterns are involved. However, when $L^{(D)}$ increases to 4, performance is degraded. This is because when $L^{(D)} = 4$, the coarsest decoder has a decoding length of only 2, and cannot provide useful global temporal information.

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 new coarse-to-fine fusion mechanism, which integrates all the decoders into an ensemble, and a multiresolution shape-forcing loss, which encourages decoders' outputs to match the input's global temporal shape at multiple resolutions. This avoids overfitting the nonlinear local patterns at a higher resolution, and alleviates error accumulation during decoding. Experiments on various time series benchmark data sets demonstrate that the proposed model achieves better anomaly detection performance than competitive baselines.

Acknowledgments

This work 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 *International Conference on Learning Representations*.

Chia, C.-C.; and Syed, Z. 2014. Scalable noise mining in long-term electrocardiographic time-series to predict death following heart attacks. In *International Conference on Knowledge Discovery and Data Mining*.

Chung, J.; Ahn, S.; and Bengio, Y. 2017. Hierarchical multiscale recurrent neural networks. In *International Conference on Learning Representations*.

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-series. In *International Conference on Machine Learning*.

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:1612.06676.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*.

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 *International Conference on Computer Vision*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

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 *International Joint Conferences on Artificial Intelligence*.

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*.

Kingma, D. P.; and Ba, J. L. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Le Guen, V.; and Thome, N. 2019. Shape and time distortion loss for training deep time series forecasting models. In *Advances in Neural Information Processing Systems*.

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*.

Ma, Q.; Lin, Z.; Chen, E.; and Garrison, C. 2020. Temporal pyramid recurrent neural network. In *AAAI Conference on Artificial Intelligence*.

Ma, Q.; Zheng, J.; Li, S.; and Cottrell, G. W. 2019. Learning representations for time series clustering. 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 multisensor anomaly detection. Preprint arXiv:1607.00148.

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 *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 *International Conference on Machine Learning*.

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 *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 *International Conference on Machine Learning*.

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 analysis of stationary time series. *Journal of the Royal Statistical Society* 102(2): 295–298.

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

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 Conference on Artificial Intelligence*.

Zhou, B.; Liu, S.; Hooi, B.; Cheng, X.; and Ye, J. 2019. BeatGAN: Anomalous rhythm detection using adversarially generated time series. In *International Joint Conferences on Artificial Intelligence*.