Supplementary Materials

1. Pseudo-code of WETAS Framework

Algorithm 1 shows the pseudo-code of training the framework. In practice, the model parameters are updated by using minibatch SGD with the Adam optimizer [5].

Algorithm 1: Optimizing the WETAS framework

Input: A training set containing N instances of temporal data $\{(\mathbf{X}^{(1)}, y^{(1)}), \ldots, (\mathbf{X}^{(N)}, y^{(N)})\}$
Output: Updated DiCNN model $f(\cdot; \Theta)$ and the anomaly weight vector $\mathbf{w}$

while *convergence* **do**
  for $i = 1, \ldots, N$ **do**
    ▷ Compute the local anomaly scores
    $\mathbf{h}_1^{(i)}, \ldots, \mathbf{h}_T^{(i)} = f(\mathbf{x}_1^{(i)}, \ldots, \mathbf{x}_T^{(i)}; \Theta)$
    $s_1^{(i)}, \ldots, s_T^{(i)} = \sigma(\mathbf{w}^\top \mathbf{h}_1^{(i)}), \ldots, \sigma(\mathbf{w}^\top \mathbf{h}_T^{(i)})$
    ▷ Compute the global anomaly score
    $\mathbf{h}_*^{(i)} = \text{global-pooling}(\mathbf{h}_1^{(i)}, \ldots, \mathbf{h}_T^{(i)})$
    $s_*^{(i)} = \sigma(\mathbf{w}^\top \mathbf{h}_*^{(i)})$
    ▷ Obtain the sequential pseudo-label
    $m_1^{(i)}, \ldots, m_T^{(i)} =$ min-max-norm$(\mathbf{w}^\top \mathbf{h}_1^{(i)}, \ldots, \mathbf{w}^\top \mathbf{h}_T^{(i)})$
    $\widetilde{\mathbf{z}}_+^{(i)} = y^{(i)} \cdot \phi_L(m_1^{(i)}, \ldots, m_T^{(i)})$
    $\widetilde{\mathbf{z}}_-^{(i)} = (1 - y^{(i)}) \cdot \phi_L(m_1^{(i)}, \ldots, m_T^{(i)})$
    ▷ Calculate the two losses
    $\mathcal{L}_c = -y^{(i)} \log s_*^{(i)} - (1 - y^{(i)}) \log(1 - s_*^{(i)})$
    $\mathcal{L}_a = \max(0, (1/T) \cdot \text{DTW}_\gamma(\widetilde{\mathbf{z}}_+^{(i)}, \mathbf{X}^{(i)}) - (1/T) \cdot \text{DTW}_\gamma(\widetilde{\mathbf{z}}_-^{(i)}, \mathbf{X}^{(i)}) + \beta)$
    $\mathcal{L} = \mathcal{L}_c + \mathcal{L}_a$
    ▷ Update all the model parameters
    $\Theta = \Theta - \eta \cdot \partial \mathcal{L} / \partial \Theta$
    $\mathbf{w} = \mathbf{w} - \eta \cdot \partial \mathcal{L} / \partial \mathbf{w}$

2. Computation of Dynamic Time Warping

The dynamic time warping (DTW) discrepancy (and the optimal alignment matrix) between two time-series of lengths M and N is usually computed by solving a dynamic program based on Bellman recursion, which takes a quadratic $O(MN)$ cost. The continuous relaxation of DTW (i.e., soft-DTW), which enables to calculate the gradient of DTW with respect to its input, also can be computed in a similar way to the original DTW [1]. Algorithm 2 presents the detailed algorithms for computing $\text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$ and $\nabla_{\Delta(\widetilde{\mathbf{z}}, \mathbf{X})} \text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$ based on Bellman recursion.

Algorithm 2: Forward recursion and backward recursion to compute $\text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$ and $\nabla_{\Delta(\widetilde{\mathbf{z}}, \mathbf{X})} \text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$

1 **Function** *forward* $(\widetilde{\mathbf{z}}, \mathbf{X})$**:**
    ▷ Fill the alignment cost matrix $R \in \mathbb{R}^{L \times T}$
2 $R_{0,0} = 0$
3 $R_{:,0} = R_{0,:} = \infty$
4 **for** $l = 1, \ldots, L$ **do**
5 **for** $t = 1, \ldots, T$ **do**
6 $R_{l,t} = \delta(\widetilde{z}_l, \mathbf{x}_t) + \min_\gamma \{R_{l-1,t-1}, R_{l,t-1}\}$
7 **return** $\text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X}) = R_{L,T}$
8 **Function** *backward* $(\widetilde{\mathbf{z}}, \mathbf{X})$**:**
    ▷ Fill the soft alignment matrix $E \in \mathbb{R}^{L \times T}$
9 $E_{L+1,T+1} = 1$ ▷ $E_{l,t} := \partial R_{L,T} / \partial R_{l,t}$
10 $E_{:,T+1} = E_{L+1,:} = 0$
11 $R_{:,T+1} = R_{L+1,:} = -\infty$
12 **for** $l = L, \ldots, 1$ **do**
13 **for** $t = T, \ldots, 1$ **do**
14 $a = \exp \frac{1}{\gamma}(R_{l,t+1} - R_{l,t} - \delta(\widetilde{z}_l, \mathbf{x}_{t+1}))$
15 $b = \exp \frac{1}{\gamma}(R_{l+1,t+1} - R_{l,t} - \delta(\widetilde{z}_{l+1}, \mathbf{x}_{t+1}))$
16 $E_{l,t} = a \cdot E_{l,t+1} + b \cdot E_{l+1,t+1}$
17 **return** $\nabla_{\Delta(\widetilde{\mathbf{z}}, \mathbf{X})} \text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X}) = E$

In order to obtain the eligible segmentation result, we need to enforce the constraint that a single time point should not be aligned with multiple consecutive labels. To this end, our forward recursion which computes $\text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$ does not consider the $\downarrow$ relation in its recurrence; i.e., $R_{l,t}$ depends on only $R_{l-1,t-1}$ and $R_{l,t-1}$, excluding $R_{l-1,t}$ (Line 6 in Algorithm 2). Accordingly, the backward recursion which computes $\nabla_{\Delta(\widetilde{\mathbf{z}}, \mathbf{X})} \text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$ also does not allow the $\uparrow$ relation; i.e., $E_{l,t}$ is obtained from $E_{l,t+1}$ and $E_{l+1,t+1}$, excluding $E_{l+1,t}$ (Line 16 in Algorithm 2).

Note that the gradient of soft-DTW with respect to the cost matrix, $\nabla_{\Delta(\widetilde{\mathbf{z}}, \mathbf{X})} \text{DTW}_\gamma(\widetilde{\mathbf{z}}, \mathbf{X})$, can effectively update the anomaly weight vector $\mathbf{w}$ as well as the model parameters of DiCNN Θ by the help of the gradient back-propagation. This is possible because each entry of the cost matrix $[\Delta(\widetilde{\mathbf{z}}, \mathbf{X})]_{lt} = \delta(\widetilde{z}_l, \mathbf{x}_t)$ is defined by the binary cross entropy between a pseudo-label $\widetilde{z}_l$ and a local

anomaly score $s_t = \sigma(\mathbf{w}^\top \mathbf{h}_t)$,

$$\delta(\widetilde{z}_l, \mathbf{x}_t) = -\left\{\widetilde{z}_l \cdot \log s_t + (1 - \widetilde{z}_l) \cdot \log(1 - s_t)\right\}.$$

In the end, the gradient optimizes each local anomaly score to be closer to the pseudo-labels that are softly-aligned with the time point.

Complexity Analysis. In terms of efficiency, we analyze that our framework additionally takes the computational cost of $O(LT)$ for DTW alignment between $\widetilde{\mathbf{z}}$ and $\mathbf{X}$ (per CNN inference and its gradient back-propagation), as described in Algorithm 2. Furthermore, because of 1) the small L value ($L \ll T$), 2) batch-wise computations in the PyTorch framework, and 3) GPU parallel computations for DTW recursions based on the numba library, it does not raise any severe efficiency issue.

3. Reproducibility

For reproducibility, our implementation is publicly available[1], and Table 1 presents the optimization details of WETAS. We empirically found that the performances are hardly affected by these hyperparameters for the optimization.

Table 1: Details for the optimization of WETAS.

Batch size	32 (for EMG, GHL, SMD) 8 (for Subway)
Optimizer	Adam optimizer
Initial learning rate	0.0001
Max # epochs	200
Stopping criterion	Instance-level F1 (on validation)

4. Hyperparaemter Search

For our WETAS framework, we search the optimal values of the following hyperparameters: the length of a sequential pseudo-label $L \in \{4, 8, 12, 16\}$, the anomaly threshold for pseudo-labeling $\tau \in \{0.1, 0.3, 0.5, 0.7\}$, and the margin size for the alignment loss $\beta \in \{0.1, 0.5, 1.0, 2.0\}$. The optimal values are selected based on instance-level F1-scores on the validation set. The selected hyperparameter values for reporting the final performance are listed in Table 2. In case of the smoothing parameter γ used for soft-DTW, we fix its value to 0.1 without further tuning.

5. Baseline Methods

We describe the details of the anomaly detection methods which are used as the baseline in our experiments. All of them employ their own deep neural networks to effectively model the temporal dependency among time points, and compute the anomaly score for each point or segment.

[1]https://github.com/donalee/wetas

Table 2: Selected hyperparameter values for WETAS.

Datasets	EMG	GHL	SMD	Subway
global-pooling	avg	max	max	avg
L	4	16	12	12
τ	0.3	0.1	0.5	0.5
β	0.1	1.0	0.5	0.5

- **Donut** [11]: A simple VAE model optimized by the modified evidence lower bound (M-ELBO). It also uses a sampling-based imputation technique for missing points, in order to effectively deal with anomalous points during the detection.
- **LSTM-VAE** [8]: A VAE model that employs long short term memory (LSTM) for its encoder and decoder. It is trained to reconstruct the non-anomalous training data well, and defines the anomaly score by the reconstruction error.
- **LSTM-NDT** [2]: A LSTM network that is trained to predict the next input (i.e., sequence modeling). It additionally adopts the non-parametric dynamic thresholding (NDT) technique to automatically determine the optimal anomaly threshold.
- **OmniAnomaly** [9]: The state-of-the-art point-level anomaly detection model that uses gated recurrent units (GRU) as the encoder and decoder of VAE. It incorporates advanced techniques into VAE, including normalizing flows and a linear Gaussian space model, to consider stochasticity and temporal dependency among the time points.
- **DeepMIL** [10]: The multiple-instance learning method[2] that learns from weakly labeled temporal data. It produces the anomaly prediction for each fixed-length segment, thus the result can be used for temporal anomaly segmentation. We consider different numbers of the segments in a single instance, denoted by DeepMIL-4, 8, 16.

Note that Donut, LSTM-VAE, LSTM-NDT, and OmniAnomaly fall into the category of unsupervised learning as they do not utilize any anomaly labels for training, and their variants that use only normal instances are categorized as semi-supervised learning. DeepMIL is the only existing method that is based on weakly supervised learning.

6. Additional Experiments

Comparison with frame-level video anomaly detectors. In case of the video dataset (i.e., Subway), we additionally report the AUROC scores of unsupervised video anomaly

[2]The term "multiple-instance" used in MIL refers to same-length temporal segments that compose a single bag. Thus, "instance" and "bag" in MIL respectively correspond to "segment" and "instance" in our approach.

detection methods [3, 4, 6, 7] as a benchmark. They aim to train the networks that take spatio-temporal (or spatial) inputs to compute the anomaly score of each video frame. Note that all these methods produce the frame-level (or point-level) anomaly predictions, without utilizing the instance-level anomaly labels for training. In Table 3, the weakly supervised methods[3] (i.e., DeepMIL and WETAS) show better performance than the unsupervised methods. Even though DeepMIL and WETAS simply use the pre-computed visual features of each frame (extracted by the pre-trained ResNet), it outperforms the other domain-specific baseline methods (optimized in an end-to-end manner) by the help of the instance-level anomaly labels. This indicates that leveraging the weak supervision can be more effective to discriminate normal and anomalous video frames compared to fine-tuning the networks for visual feature extraction.

Table 3: Comparison with several recent frame-level video anomaly detection methods, Dataset: Subway.

Methods	AUROC	Methods	AUROC
Ionescu et al. [3]	85.7%	DeepMIL-4 [10]	95.7%
Ionescu et al. [4]	95.1%	DeepMIL-8 [10]	96.9%
Liu et al. [6]	93.1%	DeepMIL-16 [10]	96.4%
Pang et al. [7]	92.7%	WETAS (ours)	97.8%

Learning curves. We plot the learning curves of WETAS by using the EMG, GHL, SMD, and Subway datasets. In Figure 1, as the number of epochs increases, the classification loss and alignment loss consistently decrease for both the training and validation sets. This implies that WETAS can infer more accurate sequential pseudo-labels as the training progresses, and the alignment loss better guides its model to output anomaly scores that are well aligned with the pseudo-label. As illustrated in Figure 4 (in our paper), the pseudo-labeling and the DTW-based segmentation collaboratively improve with each other. Consequently, the instance-level and point-level F1-scores for the test set increase as well. The results empirically show that the instance-level F1-score (or the total loss) on the validation set can be a good termination criterion for the optimization of WETAS, thus we adopt it in our experiments.

Sensitivity Analyses. We finally examine the performance changes of WETAS with respect to the following hyperparameters: the length of a sequential pseudo-label L, the anomaly threshold for pseudo-labeling τ, and the margin size for the alignment loss β. In Figure 2, we observe that the final performance of WETAS is not sensitive to β,

[3]To compute AUROC for the weakly supervised methods, we regard the anomaly score of each segment as the score for all points in the segment (for DeepMIL), and use the local (or point-level) anomaly scores without the DTW-based segmentation (for WETAS).

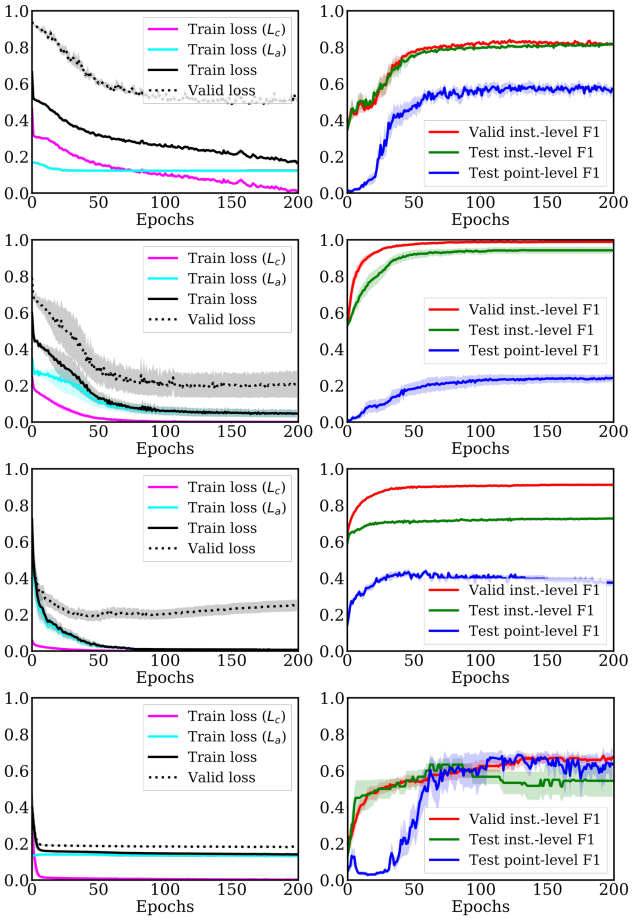

Figure 1: Learning curves of WETAS in terms of training/validation loss and test/validation F1-score, Datasets: EMG, GHL, SMD, and Subway (from the first row).

and it shows higher F1-score when using a smaller τ and a larger L. Specifically,

- β does not much affect the final performance of WETAS, because it simply controls the margin size in the alignment loss.
- A smaller τ encourages to find out more anomalous segments, which leads to a high recall for anomaly detection, by making the sequential pseudo-label have more number of 1s.
- A larger L allows a finer-grained segmentation by aligning the time points with more number of anomaly pseudo-labels.

Nevertheless, unlike the DeepMIL, the granularity of pseudo-labeling is not a critical factor for WETAS because the DTW-based segmentation is capable of dynamically aligning an input instance with its pseudo-label. In conclusion, the best performing hyperparameter values successfully optimize WETAS under the weak supervision so that it can identify variable-length anomalous segments.