Appendix

The machine used in the experiments has one Intel Core i7-12700K CPU with 128GB memory and one NVIDIA GeForce RTX 3060 GPU with 12GB memory.

Experiment Settings for Anomalous Sequence Detection

Sequences in each time series are preprocessed with z-score normalization. All final scores output by detectors are normalized in $[0, 1]$. For detectors that rely on randomization, we report the average result of 10 trials on each time series.

[**Datasets**] Synthetic datasets noisy_sine and ARMA are originated from a previous work [22]. Real-world datasets include MIT-BIH Supraventricular Arrhythmia Database (MBA) [17,31] and other datasets from various domains have been studied in earlier works [24, 47, 22].

Some datasets have two versions, e.g., ann_gun and stdb_308; and each version uses one of the two variables. When either version produces similar AUC for most detectors, we have chosen to use one only. Some datasets are trivial, e.g., chfdb_chf0175 and qtdbsel102; and all detectors have the perfect result (AUC=1), so we do not show them in Table 7.

We labelled anomalous periods for each time series following the previous work [24, 47, 22]. Details are given in Table 17. Positions of anomalies in MBA datasets can be seen in folder "MBA_Annotation".

The period of some datasets varies slightly at different time steps in the series; but it has no effect on the detection accuracy of all algorithms. Our algorithm works well when the sequence length is set to be roughly the length of the period.

Brief descriptions of some datasets are given as follows.

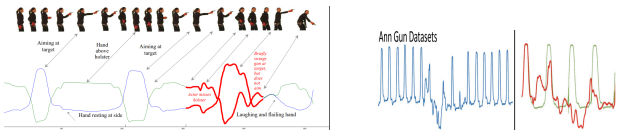
dutch_pwrdemand: There are a total of 6 anomalous weeks. Some papers [22, 24, 4] use this dataset with fewer anomalies because they treat continuous anomalous weeks as one anomaly.

ann_gun: It has only one anomalous period when it was first used in Keogh's work [24], as shown in Figure 11a. Other anomalous periods in this dataset were later identified [4], and they are shown in Figure 11b.

Patient_respiration: Like the previous work [22], we use the subset that begins at 15500 and ends at 22000 from the nprs44 dataset [24]. There are one apparent anomaly and one subtle anomaly in this dataset as shown in Figure 12a.

TEK: Following the previous work [22], we also concatenate dataset TEK14, TEK16 and TEK17 as TEK of length 15000. In Keogh's work [24], a total of 4 anomalies are marked. But TEK14 has 2 anomalous snippets belonging to the same period as shown in Figure 12b. Since we regard each anomaly as an anomalous sequence of one complete period, it is treated as one anomalous periodic sequence of length 1000. So there are a total of 3 anomalous sequences in our annotations of this dataset.

MBA803, MBA805, MBA806, MBA820 and MBA14046: These datasets are subsets of the full MBA dataset, as used in the previous work [4].



(a) (b)

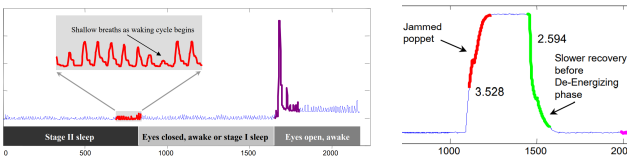
Fig. 11: (a) One anomaly period and (b) additional anomalous periods as identified by [4] on the ann_gun dataset. The diagrams are extracted from [24] and [4], respectively.

[**Algorithms**] The STOMP [65] implementation of MP is used; NormA is from `http://helios.mi.parisdescartes.fr/~themisp/norma/`;

$\mathcal{K}_I$-based detectors are our implementations based on [58]; and WFD is from `https://github.com/GAMES`

Table 17: Locations of anomalous periodic sequences in each dataset in terms of index i in X_i. Other details are in the MBA_Annotation folder at `https://github.com/IsolationKernel/TS/tree/main/AnomalousSubsequenceDetection`.

Dataset	period length	anomalous period index i
noisy_sine	300	5,10,20,30
ARMA	500	100,101,160
GPS_trajectory	2200	2,5
Patient_respiration	150	6,33
TEK	1000	1,9,12
dutch_pwrdemand	672	0,12,17,18,19,51
ann_gun	150	2,14,15,16,18
mitdb_100_180	250	7
mitdbx_108	370	11,27,28,29,30
stdb_308	400	6
lstdb_20221_43	170	4
lstdb_20321_40	200	4
qtdbsele0606	140	8
MBA803	105	
MBA805	100	see details
MBA806	75	in folder:
MBA820	100	MBA_Annotation
MBA14046	90	



(a) (b)

Fig. 12: Anomalies in (a) Patient_respiration; (b) a period of TEK14. The diagrams are extracted from [24].

`-UChile/Wasserstein-Fourier`. Others are from scikit-learn.org. All are in Python.

The parameter search ranges of all algorithms used are given in Table 18.

As for the 1Line method, we use one of the following five types of basic vectorized primitive functions in Matlab as an anomaly score for each sliding window of size ω:

I. $\pm$diff(X): the difference between the current point and the previous point. Here $\omega = 1$.
II. $\pm$movmax(X, $\boldsymbol{\omega}$)
III. $\pm$movmin(X, $\boldsymbol{\omega}$)
IV. $\pm$movmean(X, $\boldsymbol{\omega}$)
V. $\pm$movstd(X, $\boldsymbol{\omega}$)

where X is the time series; and the maximum, minimum, mean or standard deviation is computed for each window of $\boldsymbol{\omega}$ points.

We run these 5 one-liners on each dataset and report the median AUC (out of the five values) in Table 7. Low median values indicate that the datasets are hard to detect using the 1Line method; otherwise, the datasets have anomalies that can be easily detected.

Table 18: Parameter search ranges. All distribution-based detectors have $m = V$, where V is the period length of each time series, and have two parameters that need to be tuned. NormA & STOMP have one parameter; iForest has two.

Algorithm	Parameter search ranges
IDK^2,IDK*IK,k-IDK	$\psi, \psi_2 \in \{2^q \mid q = 1, 2, \ldots, 7\}$; $t = 100$
OCSVM,OCSMM	$\sigma \in \{10^i \mid i = -4, -3, \ldots, 0, 1\}$
Wasserstein	#bin $= \{10,20,50,100,200\}$; $p = 2$
NormA,STOMP,iForest,1Line	$\omega \in$\{V, V$\pm$50, V$\pm$100\}
iForest	$\psi \in \{2^q \mid q = 1, 2, \ldots, 11, 12\}$; $t = 100$
LOF,kNN, k-IDK	$k \in \{1, 3, 5, 7, 11, 21, 51, 101, 201, 501, 1001, 2001\}$

[**Measures**] The detection accuracy of an anomaly detector is measured in terms of AUC (Area under ROC curve). As all the anomaly detectors are unsupervised learners, all models are trained with the given datasets with no labels. Only after the trained models have made predictions, the ground truth labels are used to compute the AUC for each dataset.

Given a periodic time series X of length n and period length m, a sequence X_i of X is a subset of contiguous values of length m, for $i = 1, \ldots, s$, where $s = \lfloor length(X)/m \rfloor$. A distribution-based (non-sliding-window) algorithm outputs a score of each periodic sequence X_i. Then AUC can be calculated based on scores α_i for X_i $\forall i = 1, \ldots, s$.

An anomaly detector using the sliding window size ω produces a total of $length(X) - \omega + 1$ sequences from X. When calculating AUC, scores of the sliding sequences are transformed into periodic sequence scores as follows: Let S_h be the anomaly score of sequence X_h, where $1 \leq h \leq (length(X) - \omega + 1)$. The final score corresponds to a periodic sequence X_i is the maximum score of S_h $\forall h$ such that at least half of X_h is included in X_i.

Experiment Settings for Anomalous Time Series Detection

[**Datasets**] Out of the initial 109 time series datasets (used in [10]) in UCR Time Series Classification Archive [9], We remove the datasets having the length of time series less than 200 and choose the top 20 datasets with the largest number of time series for our evaluation. (The datasets MixedShapesSmallTrain and FreezerSmallTrain are not included because they are the subsets of the two datasets chosen, namely, MixedShapesRegularTrain and FreezerRegularTrain, respectively.)

New labeled datasets (normal vs. anomalous) are created as follows.

Each dataset in the archive contains a training set and a testing set. For each of the combined training-and-testing time series dataset having k classes, we take $\lfloor k/2 \rfloor$ largest classes and treat them to be the normality. Then, we sample 2% of time series from the other classes with an initial seed number 10 and treat them as anomalous time series.

The characteristics of the generated datasets are shown in Table 19. Note that the previous work [1,2] chose only 1 class as the normality which makes the task trivial. The way we produce datasets allowing several classes as the normality make this task more interesting and challenging.

[**Algorithms**] $\mathcal{K}_I$-based detectors are our implementations based on [58]. Mini-Rocket is from `https://github.com/angus924/minirocket`. TS2Vec is from `https://github.com/yuezhihan/ts2vec`. WTK is from `https://github.com/BorgwardtLab/WTK`. NeuTraL is from `https://github.com/boschresearch/NeuTraL-AD`. All the above are in Python. DOTS is a Scala implementation from `https://github.com/B-S`

Table 19: Characteristics of the 20 datasets used in the experiment of anomalous time series detection.

Dataset	#Series	Length	#Anomalies
InsWinbeatSou	1024	256	34
UWavGesLibY	2284	315	44
ChlorineCon	2347	166	40
Yoga	1800	426	30
FordA	2574	500	47
Phoneme	1761	1024	7
UWavGesLibX	2284	315	44
UWavGesLibZ	2284	315	44
FordB	2304	500	43
Wafer	6417	152	15
UWavGesLibAll	2284	945	44
HandOutlines	884	2709	9
NonInvECGT2	1991	750	36
MixShaRegTrain	1464	1024	29
CinCECGTorso	724	1639	14
FiftyWords	711	270	3
NonInvECGTh1	1991	750	36
Mallat	1224	1024	24
StarLightCurves	5405	1024	78
FreRegTrain	1530	301	30

`eif/anomaly-detection-time-series`; Hyndman is a R implementation from `https://github.com/robjhyndman/anomalous-acm`. DTW and SBD are from the Python package tslearn [53].

The warping window of DTW is set to 5% of the length of the time series of each dataset, because the detection accuracy becomes worse when we attempt other window sizes or no windows.

For TS2Vec, we use its experimental settings of classification task because we require the instance-level representations.

For Mini-Rocket and NeuTraL, we use their default parameter configurations suggested in their papers.

The parameter search ranges of all the algorithms that need parameter tuning are given in Table 20.

Table 20: Parameter search ranges. l is the length of time series.

heightAlgorithm	Parameter search ranges
$\mathcal{K}_I$	$\psi \in \{2^q \mid q = 1, 2, \ldots, 11\}$; $t = 100$
LOF	$k \in \{1, 3, 5, 7, 11, 21, 51, 101, 201, 501, 1001, 2001\}$
DOTS	$\lambda \in \{0.05, 0.07, 0.1, 0.3, 0.5, 0.7, 1, 3, 5, 7, 10, 30, 50\}$
Hyndman	width, window $\in \{0.05l, 0.1l, 0.2l, 0.3l, 0.4l, 0.5l\}$
TS2Vec	batch size $\in \{4, 8, 16\}$
WTK	$\omega \in \{0.1l, 0.3l, 0.5l\}$
SINK	$\gamma \in \{1, 2, \ldots, 20\}$

As for the 1Line method, we use one of the following five types of basic vectorized primitive functions in Matlab as an anomaly score for each time series $T_j, j = 1, \ldots, n$, where n is the number of time series in the dataset.

I. $\pm\max(T_j)$: the maximum value of time series
II. $\pm\min(T_j)$: the minimum value of time series
III. $\pm\text{mean}(T_j)$: the mean value of time series
IV. $\pm\text{std}(T_j)$: the standard deviation of time series
V. $\pm\max(\text{diff}(T_j))$: the maximum of the difference between the current point and the previous point

We run these five 1-Line methods on each dataset and report the maximum AUC in Table 10. Low maximum values indicate that the datasets are hard to detect using any of the 1Line methods; otherwise, the datasets have anomalies which can be easily detected.

Additional Experimental Results for Anomalous Time Series Detection

Table 21: Actual runtime (in CPU seconds) when the number of time series increases from 2^{10} to the total number of 50000 on the InsectSound datatset.

#Series	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	50000
DTW	1073	6890	57659	–	–	–	–
SBD	30	122	479	1891	7521	29993	69748
WTK	10264	40927	177250	–	–	–	–
Rocket	5	11	22	44	88	195	323
$\mathcal{K}_I$	9	18	35	70	139	278	424
TS2Vec	28	29	30	32	38	46	56

Table 22: Actual runtime (in CPU seconds) when the length of time series increases from 2^{14} to the full length (236784) on the DucksAndGeese datatset.

Series length	2^{14}	2^{15}	2^{16}	2^{17}	236784
DTW	5848	23251	102546	–	–
SBD	4	21	48	100	221
WTK	71548	–	–	–	–
Rocket	3	5	11	30	70
$\mathcal{K}_I$	23	46	92	183	342
TS2Vec	162	163	166	172	179