Change Point Detection in Multi-channel Time Series via a Time-invariant Representation

Zhenxiang Cao, Nick Seeuws, Maarten De Vos, and Alexander Bertrand

***Abstract*—Change Point Detection (CPD) refers to the task of identifying abrupt changes in the characteristics or statistics of time series data. Recent advancements have led to a shift away from traditional model-based CPD approaches, which rely on predefined statistical distributions, toward neural network-based and distribution-free methods using autoencoders. However, many state-of-the-art methods in this category often neglect to explicitly leverage spatial information across multiple channels, making them less effective at detecting changes in cross-channel statistics. In this paper, we introduce an unsupervised, distribution-free CPD method that explicitly incorporates both temporal and spatial (cross-channel) information in multi-channel time series data based on the so-called Time-Invariant Representation (TIRE) autoencoder. Our evaluation, conducted on both simulated and real-life datasets, illustrates the significant advantages of our proposed multi-channel TIRE (MC-TIRE) method, which consistently delivers more accurate CPD results.**

***Index Terms*—Autoencoder, change point detection, multi-channel time series, unsupervised learning**

I. Introduction

CHANGE Point Detection (CPD) is a fundamental task in time series analysis, serving as a crucial preprocessing step in various application domains, including biomedical informatics [1], [2], speech signal processing [3], [4], financial analysis [5], [6], human activity recognition [7], [8], and climatology [9], [10]. Its primary objective is to identify abrupt changes in the underlying statistics of time series data, effectively marking the points at which the time series loses its stationarity.

Closely related to CPD is the concept of time series segmentation (TSS), which is a 'top-down' oriented method that aims to divide sequential data into smaller segments based on a finite number of states of the underlying system. For example, the AutoPlait model [11] splits input recordings by assigning the subsequences to a fixed number of regimes. TSS is often associated with classification tasks and can be addressed in supervised settings. In contrast, CPD methods take a more 'bottom-op' approach, where the aim is to detect sudden statistical changes in observed variables, even when the segments between consecutive change points lack a clear physical interpretation. CPD is inherently an unsupervised problem.

This research received funding from the Flemish Government (AI Research Program) and from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No.802895) and from the European Union's Horizon 2020 research and innovation programme under grant agreement NO.766456 (project AMPHORA). All authors are affiliated to Leuven.AI - KU Leuven institute for AI, B-3000, Leuven, Belgium.

In this study, we focus on designing a general CPD algorithm suitable for multi-channel sequential data. We aim to detect various types of change points in the statistics of multi-channel time series. This includes not only the common temporal changes in individual channels, such as changes in mean, variance, and distributions, but also spatial changes in the cross-channel correlation structure. It is important to note that our aim is not to detect changes in the shapes of subsequences when these changes do not significantly impact the statistics of data, as such cases fall outside the scope of the CPD problem [12]. Assigning meaning or interpretation to the segments between detected change points is a task typically addressed in other applications like TSS.

Traditional CPD methods like Cumulative Sum (CUSUM) [13] and Generalized Likelihood Ratio (GLR) [14], [15] compare distribution models in adjacent intervals around change point candidates. A drawback of these methods is their reliance on pre-designed parameter models, limiting their performance on real-world datasets. To address this issue, some studies [16], [17] attempted to estimate density ratios instead of original distributions. However, the accuracy of these approaches relies heavily on selecting suitable hyperparameters, posing a challenge for users.

In the era of big data, data-driven learning methods, particularly neural network-based frameworks, have made substantial progress compared to traditional statistical CPD techniques. For example, Munir et al. [18] and Perslev et al. [19] employed convolutional neural networks (CNNs) to identify change points and anomalies in time series data. Additionally, due to the sequential nature of time series data, dynamic deep learning models like Long Short-Term Memory (LSTM) [20], [21] and Gated Recurrent Units (GRU) [22] have been extensively explored for CPD applications [23], [24]. Among these neural network-based approaches, autoencoder-based approaches have gained popularity. For instance, the Autoencoder-based Breakpoints Detection (ABD) model [25] employs an autoencoder structure to map sequential data into latent space features. It identifies change points when a significant dissimilarity is detected among these latent features. De Ryck et al. [26] introduced the Time-Invariant Loss to train autoencoders with a time-invariant representation (TIRE) model, leading to significant improvements in CPD performance on various simulated and real-world benchmark datasets. However, the TIRE model does not explicitly account for the multi-channel (spatial) structure in time series data.

In this paper, we enhance TIRE for multi-channel time series by addressing two types of change points: those affecting the multi-channel correlation structure and those im-

pacting only one or a few specific channels. We propose an unsupervised end-to-end learning framework with two parallel branches to model these change points separately.

Our work's contributions can be summarized as follows: • We introduce an effective unsupervised multi-channel CPD method, explicitly considering the spatial or multi-channel structure in time series representation, in contrast to the original TIRE model.

• We conduct a comprehensive evaluation and ablation study on various simulated and real-world datasets, showcasing the superior performance of our multi-channel model compared to state-of-the-art methods.

• We offer an open-source implementation of our proposed multi-channel TIRE model.

The paper's structure is as follows: Section II briefly reviews the original TIRE method. Section III addresses the multi-channel CPD problem and introduces our multi-channel TIRE (MC-TIRE) model. Section IV presents the experiments on simulated and real-life datasets, with results in Section V. In Section VI, we conduct an ablation study to gain deeper insights into our method's workings and draw conclusions in Section VII.

II. Review of TIRE-based CPD

The TIRE model, as described in [26], comprises a simple autoencoder (AE) with a single fully-connected layer in both the encoder and decoder. It takes a window of N time samples as input. By using a limited number of features (neurons in the hidden layer), the AE is designed to capture general data trends. To adapt these extracted representations for the CPD problem, the authors of [26] introduced the concepts of time-invariant (TI) and time-variant (TV) features.

The idea is that TI features ($\mathbf{s}$), extracted from consecutive windows, should be close to each other in feature space when no change point is present. Time-variant features ($\mathbf{u}$), on the other hand, contain additional time or window-specific information necessary for decoding the current window. In essence, a window with N time samples is reconstructed using a combination of features from $\mathbf{s}$ (which remain consistent across multiple windows) and $\mathbf{u}$ (which can vary for each window). For example, to encode a linear trend in the time domain, $\mathbf{s}$ can encode the slope of the trend (consistent across windows), while $\mathbf{u}$ can encode the intercept at the window edge (time or window-dependent). Subsequently, only the dissimilarity between the TI features is considered in a post-processing stage to pinpoint change points.

To ensure that a feature in $\mathbf{s}$ possesses the TI property, TIRE employs a dedicated loss function alongside the reconstruction loss: the TI loss ($\mathcal{L}^{TI}$). Its objective is to minimize the distance between TI features learned from consecutive windows, defined as:

$$\mathcal{L}^{TI} = \sum_{t} ||\mathbf{s}[t] - \mathbf{s}[t-1]||_2. \tag{1}$$

III. Multi-channel CPD

In this section, we delve into the multi-channel CPD problem, introduced in subsection III-A. Subsections III-B cover the pre-processing and input structure of our proposed model. Subsection III-C provides insights into the specifics of our proposed multi-channel TIRE (MC-TIRE) model and its training process. In subsections III-D and III-E, we elucidate how the trained MC-TIRE model can be employed for CPD through a dedicated post-processing procedure. The implementation of the MC-TIRE model is accessible at https://github.com/caozhenxiang/MC-TIRE.

A. Definitions of Multi-channel Change Points

In multi-channel time series CPD, we focus on two types of change points:

Coherence changes: Transitions in inter-channel correlation or coherence, often affecting multiple channels simultaneously due to external factors like signal source shifts.

Residual changes: Changes that impact only one or a few specific channels, often due to alterations in individual sensor characteristics.

To capture these two changes, we approximate a stationary C-channel time series segment $\mathbf{X}_{seg} \in \mathbb{R}^{C\times K}$ with a length of K time samples using the model:

$$\mathbf{X}_{seg} = \mathbf{A}^T\mathbf{S} + \mathbf{B}, \tag{2}$$

where the matrix $\mathbf{S} \in \mathbb{R}^{R\times K}$ models R latent uncorrelated random source signals, which are mixed into the observations via a mixing matrix $\mathbf{A} \in \mathbb{R}^{R\times C}$, and where $\mathbf{B} \in \mathbb{R}^{C\times K}$ contains the per-channel signal components that can not be captured by the first term. The underlying idea is that the term $\mathbf{A}^T\mathbf{S}$ in (2) aims to capture the across-channel information as much as possible, while the residual information is represented by the $\mathbf{B}$ term. As a result, if the entries of $\mathbf{A}^T\mathbf{S}$ (or $\mathbf{B}$) show a significant discrepancy in two consecutive segments, a candidate coherence (or residual) change should be reported.

Remark: We introduce two change point types to clarify the MC-TIRE concept, without the intention of explicit classification. We use "residual change" instead of "non-coherent change" to emphasize a flexible boundary, aligning with the model in (2). It is essential to note that $\mathbf{B}$ in (2) may not be entirely incoherent, influenced by factors like underestimated R or non-linear channel relationships. This terminology matches the structure of the MC-TIRE model (see subsection III-C), with one branch capturing inter-channel coherence and another modeling residual signal components.

B. Pre-processing

Given a time series recording $\mathbf{X} \in \mathbb{R}^{C\times T}$, we first normalize each channel $\mathbf{x}^c \in \mathbb{R}^T$ into the range [-1,1], split it into overlapping windows of size N;

$$\mathbf{x}^c[t] = [x^c[t], x^c[t+1], \ldots, x^c[t+N-1]]^T \in \mathbb{R}^N, \tag{3}$$

and then concatenate them into a 2-dimensional matrix $\mathbf{Y}[t]$ with a different column for each time step t:

$$\mathbf{Y}[t] = [\mathbf{x}^1[t], \mathbf{x}^2[t], \ldots, \mathbf{x}^C[t]]^T \in \mathbb{R}^{C\times N}. \tag{4}$$

Note that consecutive windows only shift with a single time sample.

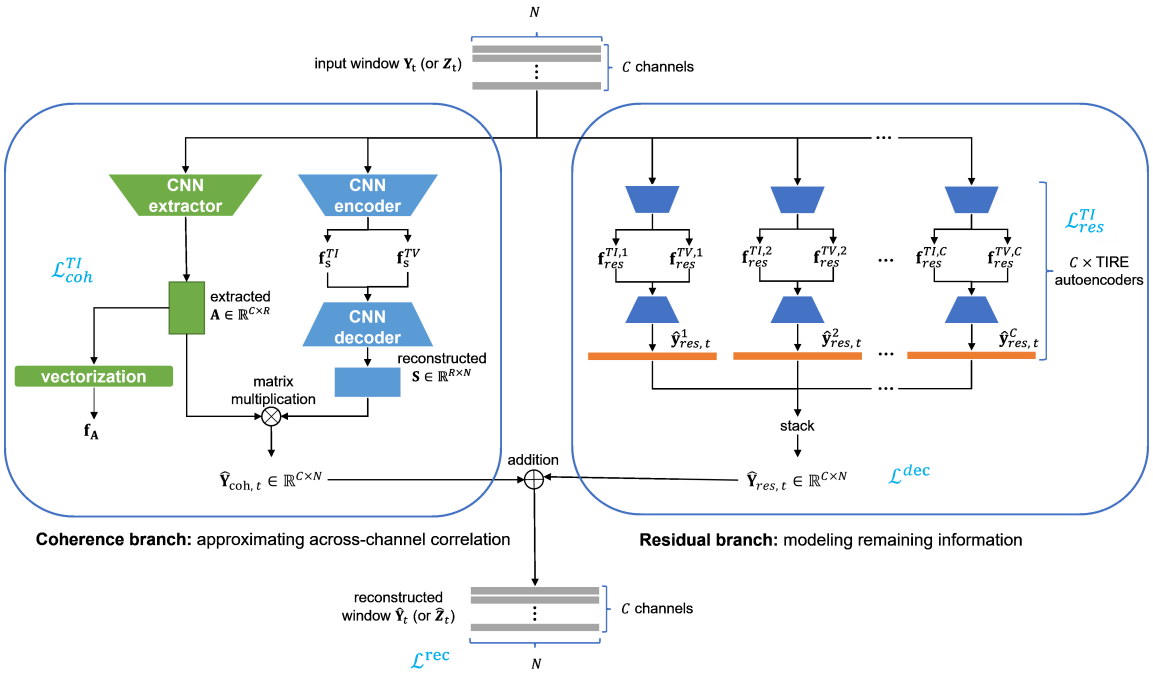


Fig. 1. The architecture of the multi-channel TIRE (MC-TIRE) model.

Similar to [26], we also apply the discrete Fourier transform (DFT) $\mathcal{F}$ with a pre-defined length of M to the data windows to obtain the spectral information. Accordingly, we get the data windows in the frequency domain:

$$\mathbf{Z}[t] = [|\mathcal{F}(\mathbf{x}^1[t])|, |\mathcal{F}(\mathbf{x}^2[t])|, \ldots, |\mathcal{F}(\mathbf{x}^C[t])|]^T \in \mathbb{R}^{C\times M}, \tag{5}$$

where $|\cdot|$ represents that we only take the magnitudes from the DFT outputs and discard the phase information. As proposed in [26], we give the freedom to the user to either use time domain (TD) information ($\mathbf{Y}[t]$), frequency domain (FD) information ($\mathbf{Z}[t]$) or both, depending on which domain is more relevant to detect change points. In the case where both TD and FD features are used as input, two separate MC-TIRE models are trained for each of them. The choice of data representation heavily depends on the application context. In case no such information is available, the default is to include both domains as a "safe" choice [26].

C. Multi-channel TIRE Model (MC-TIRE)

Beginning with the data model in (2), we introduce a novel unsupervised CPD framework, depicted in Fig. 1, that is end-to-end trainable. Given that our model also incorporates the concept of the TI loss, this work can be considered an extension of the TIRE model to multi-channel time series, denoted as a multi-channel TIRE model (MC-TIRE). Throughout the remainder of this section, we will use the TD variable $\mathbf{Y}[t]$ to describe the model. However, it is essential to note that everything presented is equally applicable to the FD input $\mathbf{Z}[t]$.

As depicted in Fig. 1, the MC-TIRE model comprises two parallel branches. The coherence branch focuses on capturing inter-channel coherence characteristics as comprehensively as possible, while the residual branch is dedicated to approximating the residual information that cannot be perfectly modeled by the coherence branch. The input window $\mathbf{Y}[t]$ is reconstructed as $\widehat{\mathbf{Y}}[t] = \widehat{\mathbf{Y}}_{coh}[t] + \widehat{\mathbf{Y}}_{res}[t]$, where $\widehat{\mathbf{Y}}_{coh}[t]$ and $\widehat{\mathbf{Y}}_{res}[t]$ represent the outputs of the coherence and residual branches, respectively.

1) Coherence branch: Within the coherence branch of the MC-TIRE model, two threads serve distinct functions. The first thread employs a CNN-based extractor to model the matrix $\mathbf{A}$ in (2) for the current time window $\mathbf{Y}[t]$. The second thread features a CNN-based autoencoder, extracting a feature vector ($\mathbf{f}_{\mathbf{S}}$) that encodes the information needed to reconstruct the source matrix $\mathbf{S}$ in (2). This feature vector consists of both a TI part ($\mathbf{f}_{\mathbf{S}}^{TI}$) and a TV part ($\mathbf{f}_{\mathbf{S}}^{TV}$).

The design philosophy behind this approach is based on the assumption that source signals exhibit stable statistics that can be captured using a limited number of parameters. These parameters should remain relatively constant across different windows, unless there is a change in the statistics of one of the source signals, indicating a coherence change point. The matrix sizes of $\mathbf{A}$ and $\mathbf{S}$ are defined by the hyperparameter R, which corresponds to the rank of the model (2), allowing for model complexity tuning.

To merge the feature vectors from both threads, we vectorize the modeled matrix $\mathbf{A}$ into a feature vector $\mathbf{f}_{\mathbf{A}}$. We consider $\mathbf{f}_{\mathbf{A}}$ as a TI representation of the coherence structure across different channels, assuming it remains approximately constant over time between change points. The TI feature vector in the coherence branch is defined by concatenating $\mathbf{f}_{\mathbf{A}}$ and $\mathbf{f}_{\mathbf{S}}^{TI}$:

$$\mathbf{f}_{coh}^{TI} = [\mathbf{f}_{\mathbf{A}}^T, (\mathbf{f}_{\mathbf{S}}^{TI})^T]^T. \tag{6}$$

Following the data model in (2), a matrix multiplication operation is applied between the matrices $\mathbf{A}$ and $\mathbf{S}$ in the

coherence branch to obtain a matrix that holds the inter-channel correlation information, denoted as $\widehat{\mathbf{Y}}_{coh}[t] \in \mathbb{R}^{C\times N}$. Both threads in the coherence branch are constructed using CNN layers. This choice is made because CNN layers enable the input window to maintain a multi-channel format during processing and explicitly consider the correlation or coherence structure across channels.

2) Residual branch: The residual branch involves C individual instances of the single-channel TIRE structure from [26]. Each TIRE model takes the C-channel windows $\mathbf{Y}[t]$ as input and reconstructs the residual component in one specific channel. It employs fully-connected layers, following the approach from [26]. By jointly training the coherence and residual branches end-to-end, the residual branch learns to model the component that the coherence branch cannot capture.

The residual component in each row in the c-th channel of $\mathbf{Y}[t]$ is reconstructed using the c-th TIRE model from a feature vector $\mathbf{f}_{res}^{c}$. This feature vector is divided into a TI part, denoted as $\mathbf{f}_{res}^{TI,c}$, and a window-dependent part, designated as $\mathbf{f}_{res}^{TV,c}$. The decoder for the c-th channel reconstructs the residual components within that channel as a vector $\widehat{\mathbf{y}}_{res}^{c}[t] \in \mathbb{R}^{N}$ based on $\mathbf{f}_{res}^{c}$. Combining the outputs of all TIRE models along the channel dimension generates a matrix $\widehat{\mathbf{Y}}_{res}[t] \in \mathbb{R}^{C\times N}$ representing all residual activities. Additionally, the residual TI feature vector is defined as the concatenation of all per-channel TI feature vectors:

$$\mathbf{f}_{res}^{TI} = [(\mathbf{f}_{res}^{TI,1})^T, (\mathbf{f}_{res}^{TI,2})^T, \ldots, (\mathbf{f}_{res}^{TI,C})^T]^T. \tag{7}$$

Training both branches in Fig. 1 jointly in an end-to-end fashion is crucial. This ensures that the residual branch is aware of what information it should encode and what information it can discard because the coherence branch captures it. This interaction and complementarity between the two branches are further enhanced during training through a dedicated loss function (see (11)).

3) Losses: To guide the fitting process of our proposed MC-TIRE model, we incorporate multiple loss functions to ensure that the model extracts pertinent information for CPD.

• **reconstruction loss ($\mathcal{L}^{rec}$):** Similar to [25] and [26], we define the end-to-end reconstruction loss to minimize the distance between the reconstructed time window $\widehat{\mathbf{Y}}_t$ and the original input window $\mathbf{Y}_t$:

$$\mathcal{L}^{rec} = \sum_t ||\widehat{\mathbf{Y}}[t] - \mathbf{Y}[t]||_2^2. \tag{8}$$

• **TI loss ($\mathcal{L}^{TI}$):** The core concept behind the TI loss is that if neighboring windows contain observations with a similar underlying distribution, their projections should be very close in the TI feature space. Therefore, TI features should exhibit similarity across neighboring windows [26]. In our context, we incorporate two instances of the TI loss for the coherence and residual branches[1]:

$$\mathcal{L}_{coh}^{TI} = \sum_t ||\mathbf{f}_{coh}^{TI}[t] - \mathbf{f}_{coh}^{TI}[t-1]||_2 \tag{9}$$

and

$$\mathcal{L}_{res}^{TI} = \sum_t ||\mathbf{f}_{res}^{TI}[t] - \mathbf{f}_{res}^{TI}[t-1]||_2. \tag{10}$$

• **decorrelation loss ($\mathcal{L}^{dec}$):** The decorrelation loss is designed to reduce the correlation between the various rows of $\widehat{\mathbf{Y}}_{res}$ in the residual branch. This encourages the coherence branch to capture as much of the across-channel correlated information as possible. Consequently, this loss helps differentiate between the coherence and residual branches. The decorrelation loss is defined as:

$$\mathcal{L}^{dec} = \sum_t ||\mathbf{C}[t] - \mathbf{I}||_2, \tag{11}$$

where $\mathbf{I}$ represents the identity matrix, and $\mathbf{C}[\mathbf{t}]$ is a $C \times C$ sample correlation matrix across channels for the entire batch. Each entry in the i-th row and j-th column in $\mathbf{C}[\mathbf{t}]$ corresponds to the Pearson correlation coefficient between the i-th and j-th rows of $\widehat{\mathbf{Y}}_{res}[t]$.

All these loss functions are combined into a single loss:

$$\mathcal{L}_{total} = \mathcal{L}^{rec} + \lambda_1\mathcal{L}_{coh}^{TI} + \lambda_2\mathcal{L}_{res}^{TI} + \lambda_3\mathcal{L}^{dec}, \tag{12}$$

where the scalars λ_1, λ_2, and λ_3 represent hyperparameters used to balance the losses. To minimize the need for hyperparameter tuning and illustrate the robustness of the MC-TIRE framework, we will employ the same default values for our experiments across all 8 benchmark datasets (as discussed in Section IV). Additionally, we will show the model's resilience to changes in loss weights in our sensitivity analysis (see Section VI-D).

D. Per-branch Detection of Candidate Change Points

So far, we have introduced the MC-TIRE framework and outlined its training process in an unsupervised manner, meaning the training procedure does not have prior knowledge of change points in the dataset. The next step is to apply the trained MC-TIRE model to the time series data and identify potential change points by comparing the dissimilarity between consecutive overlapping time windows in the TI feature space. This approach aligns with other autoencoder-based unsupervised CPD methods [25], [26]. Given the parallel structure in the MC-TIRE model, we will first explain how to identify candidate change points based on the coherence and residual TI features, a process heavily influenced by the post-processing methods in [26]. The strategy for combining the candidate change points from both branches to obtain the final alarms will be discussed in the following subsection.

In each branch of the MC-TIRE model, when both TD and FD features are accessible, they are merged into a unified feature vector, following the approach in [26]:

$$\mathbf{f}_{(\cdot)}^{TI,both}[t] = [\alpha \cdot (\mathbf{f}_{(\cdot)}^{TI,TD}[t])^T, \beta \cdot (\mathbf{f}_{(\cdot)}^{TI,FD}[t])^T]^T. \tag{13}$$

The parameters $\alpha \geq 0$ and $\beta \geq 0$ control the influence of features in the TD and FD. These values can be set as hyperparameters, either through domain-specific insights or through data-driven methods (refer to equation (13)). For simplicity, we will drop the "both" superscript from the TI features and related terms in the following discussions.

[1]The number of change points is assumed to be much smaller than the number of windows, which makes this loss a reasonable choice for the majority of terms in the summations in (9)-(10).

The assumption behind TI features is that they primarily encapsulate short-term signal statistics. However, due to imperfections and noise, the dissimilarity measure can exhibit numerous local maxima. To address this, we apply a zero-delay weighted moving average filter as in [26]. This filter helps eliminate the impact of minor fluctuations in the TI features:

$$\widetilde{\mathbf{f}}^{TI}_{(\cdot)}[t] = \sum_{k=-N+1}^{N-1} \mathbf{v}[N-k] \cdot \mathbf{f}^{TI}_{(\cdot)}[t+k], \tag{14}$$

where $\mathbf{v}$ represents a triangular-shaped weighting window defined as $\mathbf{v}[k] = \mathbf{v}[2N-k] \triangleq k/N^2$ for $1 \leq k \leq N$. Here, N is a predetermined window size, as introduced in subsection III-B. The suffix $(\cdot)$ can be either coh (for the coherence branch) or res (for the residual branch). The dissimilarity measure $\mathcal{D}_{(\cdot)}$ is defined by calculating the distances between the smoothed TI features:

$$\mathcal{D}_{(\cdot)}[t] = ||\widetilde{\mathbf{f}}^{TI}_{(\cdot)}[t] - \widetilde{\mathbf{f}}^{TI}_{(\cdot)}[t+N]||_2. \tag{15}$$

When domain-specific knowledge is unavailable, the default values of α and β in (16) are set to treat information from both domains equally, based on the dissimilarity measure:

$$\alpha = Q(\mathcal{D}^{FD}, 95\%) \text{ and } \beta = Q(\mathcal{D}^{TD}, 95\%), \tag{16}$$

where $Q(\cdot, x\%)$ denotes the $x\%$ percentile with the goal of removing outliers. $\mathcal{D}^{TD}$ and $\mathcal{D}^{FD}$ represent the dissimilarity measures associated with $\mathbf{f}^{TI,TD}_{(\cdot)}[t]$ and $\mathbf{f}^{TI,FD}_{(\cdot)}[t]$, respectively.

We subsequently utilize the weighting window $\mathbf{v}$ to further filter $\mathcal{D}_{(\cdot)}$ and reduce the number of false-positive samples:

$$\widetilde{\mathcal{D}}_{(\cdot)}[t] = \sum_{k=-N+1}^{N+1} v[N-k] \cdot \mathcal{D}_{(\cdot)}[t+k] \tag{17}$$

Finally, the peaks in the smoothed dissimilarity measure $\widetilde{\mathcal{D}}_{(\cdot)}$ are detected as candidate change points $\mathbf{P}_{(\cdot)} = \{p^1_{(\cdot)}, p^2_{(\cdot)}, \ldots, p^n_{(\cdot)}\}$. For ease of expression, we will use $p^i_{(\cdot)}$ to represent both the i-th detected candidate change point and its corresponding temporal location in this paper.

E. Combining Candidate Change Points across Branches

We have generated two sets of candidate change points ($\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$) by processing the TI features separately in both branches, namely $\mathbf{f}^{TI}_{coh}$ and $\mathbf{f}^{TI}_{res}$. However, as we will show in subsection VI-C, taking union of these two sets results in a high number of false alarms. To address this issue, we introduce additional post-processing to merge the candidate change points from both sets. We consider two scenarios and switch between them in a data-driven manner:

- **Scenario 1**: Most or all change points in the entire time series belong to a single type (either coherence or residual changes).
- **Scenario 2**: No dominant type of change points exists in the full input signal, meaning an equal distribution between coherence and residual change points.

To switch between these two scenarios, we compare the relative contributions of the two branches in the MC-TIRE model. In **Scenario 1**, where one branch clearly dominates, we discard change points detected by the less dominant branch to reduce false alarms. In **Scenario 2**, where both branches contribute significantly, we propose a pipeline to benefit from both. The implementation is as follows:

To quantify the relative contributions of both branches, we calculate the variance in the energy of the reconstructed time series ($\widehat{\mathbf{Y}}_{coh}$ and $\widehat{\mathbf{Y}}_{res}$). This variance serves as a proxy for the amount of change in the signals modeled by each branch. The underlying concept is that a substantial energy variance implies that the branch is predominantly modeling change-relevant information. Moreover, a higher energy variance is expected when the branch models higher-energy signal components, which is a desired effect. We anticipate that change point candidates caused by high-energy signal components are more relevant, while changes in the statistics of low-energy signal components, which are barely noticeable in the total signal, should be disregarded since they do not significantly alter the total signal's statistics and might be indistinguishable from noise, leading to a surge in false alarms.

We begin by calculating energy values within each reconstructed window at each time sample t for the c-th channel using the following formula:

$$E^c_{(\cdot)}[t] = ||\widehat{\mathbf{y}}^c_{(\cdot)}[t]||_2^2, \tag{18}$$

where $\widehat{\mathbf{y}}^c_{(\cdot)}[t]$ denotes the c-th row of $\widehat{\mathbf{Y}}_{(\cdot)}[t]$, and the suffix $(\cdot)$ can be selected between $_{coh}$ or $_{res}$ depending on the branch. Then, the sample variance of these energies across time t for the entire recording is calculated with the formula:

$$EV^c_{(\cdot)} = \frac{\sum_{t=1}^{T} ||E^c_{(\cdot)}[t] - \frac{1}{T}\sum_{t=1}^{T} E^c_{(\cdot)}[t]||_2^2}{T-1}. \tag{19}$$

Subsequently, we compute the channel-wise percentage of EV^c_{res} in the sum $EV^c_{coh} + EV^c_{res}$ to determine how to manage the detected candidates in both branches. These percentages for each channel are stored in a ratio vector $\mathbf{r}$:

$$\mathbf{r} = [\frac{EV^1_{res}}{EV^1_{coh}+EV^1_{res}}, \frac{EV^2_{res}}{EV^2_{coh}+EV^2_{res}}, \ldots, \frac{EV^C_{res}}{EV^C_{coh}+EV^C_{res}}]^T. \tag{20}$$

Note that $\mathbf{r}$ will only contain values between 0 and 1.

In **Scenario 1**, two possible situations arise:

i) If the majority of change points in the entire recording belong to the coherence type, it is expected that $EV_{coh} >> EV_{res}$. Consequently, all elements in $\mathbf{r}$ will be smaller than a threshold γ_1. As a result, we can safely discard all candidate change points detected by the residual branch, considering only the candidates from the coherence branch as the output alarms.

ii) Conversely, if the residual change points dominate the entire signal, we select the elements in $\mathbf{r}$ that exceed another threshold γ_2. These elements correspond to the channels where the residual change points are present. Simultaneously, other elements in $\mathbf{r}$ should remain at a nominal value smaller than γ_3, i.e., there should be no values between γ_3 and γ_2, otherwise we conclude to be in Scenario 2 (see below). We only retain the candidate change points from the residual

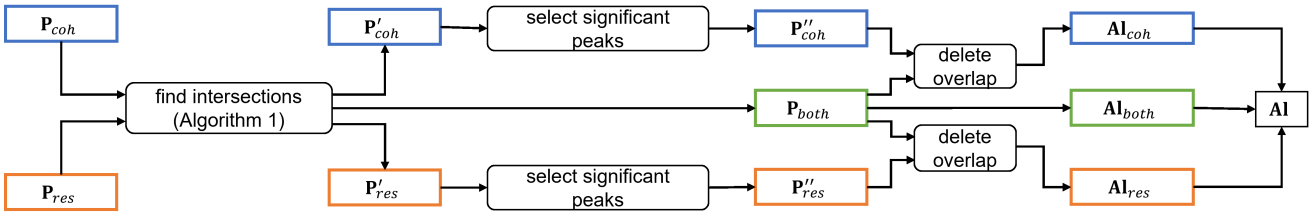


Fig. 2. The combination pipeline for balanced number of the coherence and residual changes in Scenario 2.

branch in the channels for which $\mathbf{r} > \gamma_2$ as the detected alarms.

Through experiments conducted on 8 distinct benchmark datasets, we have empirically determined that $\gamma_1 = 0.25$, $\gamma_2 = 0.65$, and $\gamma_3 = 0.25$ serve as effective default values that yield strong performance across all 8 benchmark datasets described in Section IV.

In **Scenario 2**, where the elements in the $\mathbf{r}$ vector do not meet the conditions discussed in the previous scenario, we propose a merging pipeline to harness the advantages of both branches. This pipeline, illustrated in Fig. 2, takes as input two sets of candidate change points, $\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$, and generates three distinct sets of detected alarms: those exclusively detected by only one of the branches ($\mathbf{Al}_{coh}$ and $\mathbf{Al}_{res}$), as well as those detected by both branches ($\mathbf{Al}_{both}$).

After obtaining the two sets of candidate change points, $\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$, from both branches, we seek their intersection. Given that candidate change points may not be precisely aligned with the ground truth change points and require a tolerance distance τ (see subsection IV-A1), we merge a candidate change point (p_{coh}) in the set $\mathbf{P}_{coh}$ with another candidate change point (p_{res}) from the set $\mathbf{P}_{res}$ if they are located within τ time samples of each other. In this case, we remove p_{coh} and p_{res} from their respective sets and add the merged change point (denoted as p_{both}) to the set $\mathbf{P}_{both}$, indicating that both branches have detected this candidate. The temporal location of p_{both} is estimated as a weighted mean of the time indices of p_{coh} and p_{res}, with weights computed using the smoothed dissimilarities of the TI features, $\widetilde{\mathcal{D}}_{coh}$ and $\widetilde{\mathcal{D}}_{res}$ at the corresponding time indices. To ensure that these values are comparable, they are normalized with $\widetilde{\mathcal{D}}^{avg}_{coh}$ and $\widetilde{\mathcal{D}}^{avg}_{res}$, which denote the average value over the 10% largest elements in $\widetilde{\mathcal{D}}_{coh}[\mathbf{P}_{coh}]$ and $\widetilde{\mathcal{D}}_{res}[\mathbf{P}_{res}]$. This normalization is based on a range of values to avoid using a single (possibly outlier) point for the normalization. A value of 10% was empirically found to be a reasonable range for this normalization. The process for constructing $\mathbf{P}_{both}$ is detailed in Algorithm 1. The algorithm's output comprises three sets: $\mathbf{P}_{both}$, $\mathbf{P}'_{coh}$, and $\mathbf{P}'_{res}$, with the latter two corresponding to the original sets $\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$, from which the common change points (now included in $\mathbf{P}_{both}$) have been removed.

All candidate change points in $\mathbf{P}_{both}$ are jointly detected by both branches, ensuring their reliability. For the remaining candidate change points in sets $\mathbf{P}'_{coh}$ and $\mathbf{P}'_{res}$, we further refine the selection by preserving only significant candidates and filtering out potential false positives. Specifically, we compare each element in $\widetilde{\mathcal{D}}_{(\cdot)}[\mathbf{P}'_{(\cdot)}]$ with $\widetilde{\mathcal{D}}^{avg}_{(\cdot)}$ in the respective branch, retaining candidates $p'_{(\cdot)}$ that satisfy the condition: $\widetilde{\mathcal{D}}_{(\cdot)}[p'_{(\cdot)}] > 0.25 \cdot \widetilde{\mathcal{D}}^{avg}_{(\cdot)}$. The remaining candidates are stored in set $\mathbf{P}''_{(\cdot)}$.

Because we define the temporal location of all elements in $\mathbf{P}_{both}$ using the weighted mean metric as discussed in Algorithm 1, some potential change points in $\mathbf{P}_{both}$ might still be too close to the remaining elements in $\mathbf{P}''_{coh}$ and $\mathbf{P}''_{res}$. Therefore, we again check for overlap between $\mathbf{P}''_{(\cdot)}$ and $\mathbf{P}_{both}$ using the tolerance τ. This time, we do not employ the weighted mean for merging these candidate change points. Instead, we directly remove the matched candidates from $\mathbf{P}''_{(\cdot)}$

Algorithm 1: Find intersections between candidate change points in both branches

Input:
$\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$: Sets of peaks found in coherence branch and residual branch;
$\widetilde{\mathcal{D}}^{avg}_{coh}$ and $\widetilde{\mathcal{D}}^{avg}_{res}$: Average value over 10% largest elements in $\widetilde{\mathcal{D}}_{coh}[\mathbf{P}_{coh}]$ and $\widetilde{\mathcal{D}}_{res}[\mathbf{P}_{res}]$;
τ: Pre-defined tolerance.

Output:
$\mathbf{P}'_{coh}$: Set of candidate change points that are exclusive to the coherence branch;
$\mathbf{P}'_{res}$: Set of candidate change points that are exclusive to the residual branch;
$\mathbf{P}_{both}$: Set of candidate change points that are detectable in both branches.

for p_{coh} *in* $\mathbf{P}_{coh}$ **do**
  Select p_{res} in $\mathbf{P}_{res}$ for which the temporal distance $d(p_{coh}, p_{res})$ is minimal.
  if $d(p_{coh}, p_{res}) < \tau$ **then**
    ⋆ Compute weights $a = \widetilde{\mathcal{D}}_{coh}[p_{coh}]/\widetilde{\mathcal{D}}^{avg}_{coh}$ and $b = \widetilde{\mathcal{D}}_{res}[p_{res}]/\widetilde{\mathcal{D}}^{avg}_{res}$;
    ⋆ Add change point $p_{both} = \lfloor (a \cdot p_{coh} + b \cdot p_{res})/(a+b) \rfloor$ to $\mathbf{P}_{both}$, where $\lfloor \cdot \rfloor$ denotes the floor function;
    ⋆ Remove p_{coh} and p_{res} from $\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$, respectively.
  end
end
Copy all remaining elements in $\mathbf{P}_{coh}$ and $\mathbf{P}_{res}$ to $\mathbf{P}'_{coh}$ and $\mathbf{P}'_{res}$, respectively.
return $\mathbf{P}_{both}$, $\mathbf{P}'_{coh}$ *and* $\mathbf{P}'_{res}$

since the candidates in $\mathbf{P}_{both}$ possess a higher confidence level.

The final outcome is three separate sets of detected alarms: $\mathbf{Al}_{coh}$ (containing all remaining candidate change points from $\mathbf{P}''_{coh}$), $\mathbf{Al}_{res}$ (containing all remaining candidate change points from $\mathbf{P}''_{res}$), and $\mathbf{Al}_{both}$ (containing all candidate change points in $\mathbf{P}_{both}$). Finally, we form the union ($\mathbf{Al}$) from these three alarm sets to produce our ultimate detection result.

IV. Experiments

In this section, we describe experiments to evaluate the MC-TIRE model. We provide evaluation metrics, baseline models for comparison, benchmark datasets, and details in the proposed model. All hyperparameters in MC-TIRE model were empirically chosen, often drawing from [26], with a general aim of using consistent default values across all datasets, unless otherwise specified.

A. Evaluation Metrics and Baseline methods

1) Evaluation metrics: Following [25] and [26], we clarify the criteria for determining if a detection alarm a corresponds to a ground truth change point b. To deem a detection alarm a as correct, it must meet all three of the following conditions: i) b is the closest ground truth change point to a; ii) Given a pre-defined tolerance τ, the distance between a and b fulfill the relationship: $|a - b| \leq \tau$; iii) Any ground truth change point b can only be linked to a single detected alarm a.

Based on the above definition, we employ the f1-score, which is a popular CPD evaluation metric [27], [28], [29], [30], to evaluate the performance of our proposed MC-TIRE model and other baseline algorithms.

2) Baseline methods: The performance of our proposed MC-TIRE model is compared against six baseline CPD algorithms, including GLR [14], [15], relative unconstrained least-squares importance fitting (RuLSIF) [17], Fast low-cost online semantic segmentation (Floss) [12] [2], ABD [25], kernel learning CPD (KL-CPD) [31], and the original TIRE algorithm [26].

We use the same baseline GLR algorithm as in [26], employing an AR(2) model on consecutive window pairs and their union, defining dissimilarity with a generalized log-likelihood ratio. RuLSIF utilizes kernel-specific parameters for change point detection based on density ratios between adjacent windows. In our comparison, we adhere to the parameters suggested in its original paper [17], as fine-tuning did not significantly enhance our benchmark results. The Floss algorithm employs a Matrix Profile [32] to detect changes in temporal shape patterns. As suggested in the original paper [12], we compute the corrected arc crossings (CAC) individually on each channel and segment the recording based on the averaged results across all channels. Regarding the KLCPD, ABD, and TIRE models, like our MC-TIRE model, they measure dissimilarity between features learned by neural networks (RNN for KLCPD and autoencoder for ABD and TIRE). We use default parameter settings from [31] for KL-CPD. For the TIRE model, which can be seen as a special case of our MC-TIRE, ignoring the correlation information from our coherence branch, we maintain the same architecture as in the residual branch of the MC-TIRE framework. This provides a fair comparison regarding the added benefit of the coherence branch. As for ABD, we utilize the same settings as in TIRE to avoid the influence of the specific choice of neural network architecture. All neural network-based methods, including KLCPD, ABD, TIRE, and MC-TIRE, are trained for 200 epochs. Moreover, we incorporate the same post-processing discussed in subsection III-D to the GLR, KLCPD, ABD, TIRE, and MC-TIRE algorithms to ensure that the results are not solely influenced by the introduction of our post-processing step. For RuLSIF, we adopt the post-processing outlined in the original paper [17], as this algorithm has a tailored post-processing step.

B. Benchmark Datasets

We conduct evaluations of the MC-TIRE model and all baseline methods across five simulated and three real-life datasets. The simulated changes encompass variations in the across-channel correlation structure, as well as changes in mean and variance. Additionally, we consider scenarios where changes in mean and variance are introduced by the $\mathbf{B}$ or $\mathbf{S}$ matrices, leading to impacts in a specific channel or across all channels simultaneously. Being a multi-channel extension of the TIRE model, the MC-TIRE model is theoretically capable of handling various other types of changes in individual channels, as detailed in [26].

1) Simulated datasets: All simulated datasets are created using the multi-channel data model in (2). We introduce five different types of change points, resulting in five distinct simulated datasets (as outlined below). Each dataset includes eight different realizations, with the number of channels C varying from 2 to 9. We set R to be equal to C, which means the matrix $\mathbf{A}$ in (2) is a square matrix[3]. All realizations consist of "T=5000" time samples. Along the temporal axis, we insert change points at t_n after every 100 time samples, i.e., $K = 100$ in (2).

Following [17], [26], we generate each row in $\mathbf{S}$ using a 1-dimensional auto-regressive model:

$$s(t) = a_1 s(t-1) + a_2 s(t-2) + \epsilon_t, \tag{21}$$

with the initial condition $s(1) = s(2) = 0$. The error term is Gaussian noise $\epsilon_t \sim \mathcal{N}(\mu_t, \sigma_t^2)$. As a result, $\mathbf{S}$ is guaranteed to be a full-rank matrix. Besides, we set $a_1 = 0.6$, $a_2 = -0.5$, $\mu_t = 0$, and $\sigma_t = 1.5$ unless specified otherwise. We generate the matrix $\mathbf{A}$ as a $C \times C$ matrix of full rank, in which the entries are drawn from a Gaussian distribution $\mathcal{N}(0, 1)$.

The simulated datasets can be categorized based on the type of changes:

[2]The floss is actually a TSS algorithm that focuses on signal shape patterns, and so it is not truly a CPD algorithm per se. Nevertheless, it can still perform decently on CPD problems where the change points denote changes in the signal patterns as it is the case for some of our benchmark datasets

[3]We intentionally introduce a significant difference between the value of R in the coherence branch of the MC-TIRE model and the value of R in the generated data. This is done to illustrate the robustness of MC-TIRE against an underestimation of this parameter. In general, we have found that the default value of $R = 1$ is sufficient to capture coherence changes, even when the actual rank of the matrix $\mathbf{A}$ is higher.

TABLE I
OVERVIEW OF BENCHMARK DATASETS

	Nr. of series	Length	Nr. of channels	Nr. of change points	Type of change points
change-A	8	5000	$2 \sim 9$	49	coherence change points caused by resetting $\mathbf{A}$ at time t_n
change-S (JM)	8	5000	$2 \sim 9$	49	coherence change points caused by varying mean values **in all channels** of $\mathbf{S}$ at time t_n
change-S (SV)	8	5000	$2 \sim 9$	49	coherence change points caused by varying variance **in all channels** of $\mathbf{S}$ at time t_n
change-B (JM)	8	5000	$2 \sim 9$	49	residual change points caused by varying mean values **in one channel** of $\mathbf{B}$ at time t_n
change-B (SV)	8	5000	$2 \sim 9$	49	residual change points caused by varying variance **in one channel** of $\mathbf{B}$ at time t_n
Honeybee Dance	6	827±202	3	20±4	honeybee waggle changes
HASC-2011	1	39397	3	39	human activity changes
UCI-test	1	2947	50	119	human activity changes

(i) Datasets with coherence changes:

We create three different datasets with changes in the across-channel coherence structure. These changes can be induced by altering the entries in the matrix $\mathbf{A}$ or by modifying the temporal statistics in the matrix $\mathbf{S}$ (which then affect all channels through the matrix $\mathbf{A}$). In addition, each element in $\mathbf{B}$ is sampled from a uniform distribution:

$$\mathbf{B}_{ij} \sim \mathcal{U}(\frac{1}{10}min(\mathbf{A}^T\mathbf{S}), \frac{1}{10}max(\mathbf{A}^T\mathbf{S})), \tag{22}$$

where $min(\cdot)$ and $max(\cdot)$ denote the minimal and maximal element in the matrix, respectively. We design these simulated datasets with coherence changes to demonstrate the advantage of explicitly taking the across-channel correlation into consideration in the MC-TIRE model.

• Change-A: In this dataset, we resample the matrix $\mathbf{A}$ from the Gaussian distribution $\mathcal{N}(0, 1)$ at each t_n.

• Change-S (Jumping Mean): In this dataset, we add a bias (mean) to $s(t)$ in (21). This bias value changes at each change point t_n. At time t_n, we simultaneously adjust the mean values of all rows in $\mathbf{S}$. The mean value of the c-th row in $\mathbf{S}$ is changed to

$$\mu_t^c = \begin{cases} 0, & 1 \le t \le t_1 \\ \mu^c_{t_{n-1}} + \dfrac{n}{\eta^c}, & t_{n-1} \le t \le t_n, \end{cases} \tag{23}$$

with $\eta_c \sim \mathcal{U}(-10, 10)$, which results in an increasing piece-wise constant mean over time.

• Change-S (Scaling Variance): In this dataset, the differences between adjacent segments in the c-th channel of $\mathbf{S}$ are achieved by altering the standard deviation σ_t^c:

$$\sigma_t^c = \begin{cases} 1, & t_{n-1} \le t \le t_n \text{ and } n \text{ is odd} \\ ln(e + \dfrac{n}{\xi^c}), & t_{n-1} \le t \le t_n \text{ and } n \text{ is even}, \end{cases} \tag{24}$$

where $\xi_c \sim \mathcal{U}(2, 10)$. Similar to the previous dataset, we adjust the standard deviations of all rows of $\mathbf{S}$ simultaneously.

(ii) Datasets with residual changes:

We create two additional datasets in which each change point at t_n affects only one of the channels in the matrix $\mathbf{B}$ of (2). These datasets are designed to prove that the MC-TIRE model can also identify change points that only affect one specific channel.

• Change-B (Jumping Mean): In this dataset, we generate each channel (row) from $\mathbf{B}$ with the auto-regressive model in (21). The mean value in the c-th channel (μ^c) jumps between 0 and 2 alternatively at time $t_n^c = 100 \cdot c + 100 \cdot i \cdot C$, where C is the number of channels in the current dataset and i varies from 0 to $\lfloor T/(100 \cdot C) \rfloor - 1$:

$$\mu_t^c = \begin{cases} 0, & t_{n-1}^c \le t \le t_n^c \text{ and } n \text{ is odd} \\ 2, & t_{n-1}^c \le t \le t_n^c \text{ and } n \text{ is even}. \end{cases} \tag{25}$$

It is noted that, while we introduce change points in each channel, they appear at different positions in time.

• Change-B (Scaling Variance): The last type of change points are introduced by changing the standard deviation σ_t^c in the c-th channel of $\mathbf{B}$ at time t_n^c as:

$$\sigma_t^c = \begin{cases} 0, & t_{n-1}^c \le t \le t_n^c \text{ and } n \text{ is odd} \\ ln(e + \dfrac{n}{8}), & t_{n-1}^c \le t \le t_n^c \text{ and } n \text{ is even}. \end{cases} \tag{26}$$

2) Real-life datasets: We also evaluated our MC-TIRE framework on three real-life datasets:

Honeybee Dance [33]: The honeybee dance dataset is a widely used dataset for evaluating CPD approaches, as referenced in [28], [31], [34], [35], [36]. It comprises six sequences with varying lengths, ranging from 602 to 1124 time samples, tracking the movements of a bee during a "dance" used for communication among bees. In each sequence, three features are recorded, represented by "C=3" channels. These features represent the coordinates in a 2D plane and angle differences. The ground truth change points correspond to different stages of the waggle dance.

HASC-2011 [37]: Another commonly used evaluation dataset in the CPD context is the human activity recognition data [4] [7], [16], [17], [31], [34], [38]. For our second real-life dataset, we select a subset of the HASC Challenge 2011 dataset, featuring three-axis accelerometer data. The goal in change point detection is to localize transitions among six behaviors, which typically result in noticeable changes in the statistics of observed variables: staying, walking, jogging, skipping, walking downstairs, and climbing stairs. Similar to [17], [26], [39], we choose data from person 671. However, we use the original three-dimensional (multi-channel) data as input, avoiding the conversion to a 1D time series based on the L_2-norm.

UCI-test [40]: The UCI dataset, like HASC-2011, is a human activity recognition dataset that could be employed for evaluating CPD algorithms, as referenced in [25]. This dataset

[4]The human activity recognition problem is indeed tackled by TSS algorithms. However, the transitions between different human movements often lead to changes in the observed signal statistics. For this reason, datasets of this nature are often used for evaluating CPD approaches.

captures data from an embedded accelerometer and gyroscope in a smartphone worn on the waist. It collects three-axial linear acceleration and 3-axial velocity. The corresponding labels represent six activities performed by participants: walking, walking upstairs, walking downstairs, sitting, standing, and laying. Various statistical and other characteristics of the raw sensor signals are computed, resulting in a 561-feature vector with time and frequency domain variables. In our experiments, we use only the first 50 features of the test dataset for our evaluation. We opted for this reduction due to high feature redundancy and limited computation resources. According to our observation, these 50 features effectively capture human activity transitions without compromising baseline model performance.

We provide a summary of the details for all benchmark datasets used in our evaluation in Table I. Additionally, we visualize three snippets along with the dissimilarity measures collected from the coherence branch (green) and residual branch (orange) in Fig. 3. These snippets represent simulated coherence changes, simulated residual changes, and real-life changes, respectively. Further visualization examples can be found in the supplementary materials.

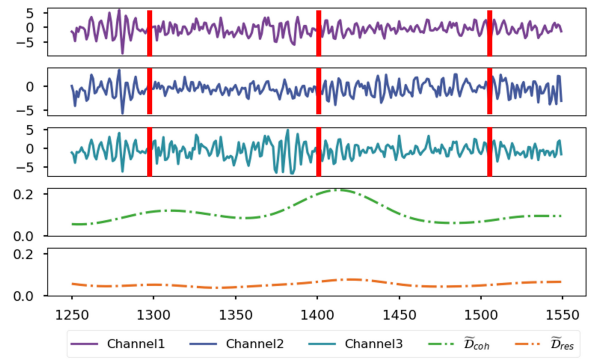
C. Experiment Details

As discussed in Section III, we utilize convolutional neural networks in the coherence branch and fully-connected autoencoders to build TIRE models for reconstructing each channel of the input time window in the residual branch. In the coherence branch, the user can specify the rank of the underlying data model, which corresponds to the number of rows in the matrices $\mathbf{A}$ and $\mathbf{S}$ in (2). In our experiments, we set $R = 1$ for all datasets to enforce a low-rank model that captures most of the correlation structure with minimal trainable parameters.

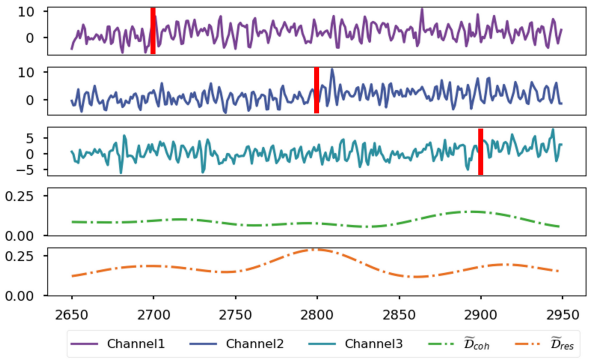
Following [26], we limit the dimension of TV features in the autoencoders to prevent over-encoding. The length of the feature vectors $\mathbf{f}_{\mathbf{S}}^{TI}$ and $\mathbf{f}_{\mathbf{S}}^{TV}$ is set to 3 and 1, respectively. In the residual branch, we maintain the sizes of $\mathbf{f}_{res}^{TI,c}$ and $\mathbf{f}_{res}^{TV,c}$ for the c-th channel to 2 and 1, respectively.

We employ the Adam optimizer with a learning rate of 0.001 to compute the gradient of our loss function $\mathcal{L}_{total}$ over each mini-batch of size 64. To mitigate the impact of randomness stemming from initialization and mini-batch shuffling, we conduct 10 runs for each dataset using all learning-based algorithms. In defining the loss function, the trade-off hyperparameters are set as $\lambda_1 = \lambda_2 = 0.01$, and $\lambda_3 = 0.1$ for all datasets.

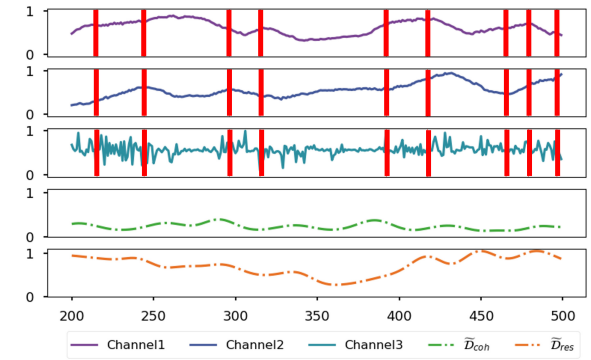
Regarding the window size N, it remains consistent across all baseline algorithms but is tailored to each dataset to ensure it is smaller than the expected temporal interval between change points, yet sufficient to capture segment characteristics. We select N values to maximize the median f1-score across all algorithms in the comparison. The resulting N values are $N = 40$ for all simulated datasets and $N = 16$, $N = 280$, $N = 8$ for the Honeybee Dance, the HASC-2011, and the UCI-test, respectively. To minimize adjustable hyperparameters, we set the tolerance τ to be equal to the window size N defined in each dataset.



(a) change-A



(b) change-B(JM)



(c) Honeybee Dance

Fig. 3. Snippets from a subset of evaluation datasets. Vertical red lines indicate the temporal and spatial positions of ground truth change points. Given that per-channel labels are unavailable for the real-life datasets, we mark the location of ground truth labels in all channels.

In line with [26], we introduce three different settings for the domain-relevant hyperparameters α and β: MC-TIRE_TD ($\alpha = 1$ and $\beta = 0$), MC-TIRE_FD ($\alpha = 0$ and $\beta = 1$), and MC-TIRE_both (α and β are set following (16)), corresponding to modeling the data in the TD, FD, or both domains, respectively.

V. Results

Our objective is to propose a generic algorithm that demonstrates strong performance across various benchmark datasets with different types of change points. TABLE II summarizes

TABLE II
THE F1-SCORES AND STANDARD DEVIATIONS ACROSS REALIZATIONS AND REPETITIONS OF MC-TIRE FRAMEWORK AND BASELINE METHODS ON SIMULATED AND REAL-LIFE DATASETS. WE HIGHLIGHT THE BEST-PERFORMING APPROACH (INDICATED BY THE HIGHEST MEAN VALUE IN F1-SCORE) FOR EACH DATASET IN BOLD.

	Simulated Datasets					Real-life Datasets		
	change-A	change-S(JM)	change-S(SV)	change-B (JM)	change-B (SV)	Honeybee Dance	HASC-2011	UCI-test
GLR [14], [15]	0.677±0.083	0.690±0.145	0.715±0.203	0.672±0.118	0.676±0.141	0.72±0.07	0.473	0.660
RuLSIF [17]	0.796±0.030	0.825±0.086	0.836±0.026	0.745±0.076	0.757±0.085	0.70±0.19	0.520	0.689
Floss [12]	0.662±0.052	0.696±0.041	0.674±0.046	0.665±0.066	0.644±0.087	0.55±0.12	0.545	0.505
ABD [25]	0.693±0.034	0.844±0.036	0.717±0.052	0.880±0.077	0.704±0.066	0.62±0.13	0.466±0.031	0.842±0.046
KLCPD [31]	0.646±0.031	0.522±0.078	0.724±0.062	0.654±0.072	0.702±0.032	0.63±0.15	0.376±0.082	0.691±0.036
TIRE_TD [26]	0.709±0.052	0.859±0.062	0.726±0.075	0.889±0.072	0.718±0.044	0.67±0.11	0.474±0.029	0.872±0.018
TIRE_FD [26]	0.710±0.050	0.842±0.052	0.835±0.070	0.790±0.079	0.805±0.052	0.74±0.11	0.486±0.028	0.624±0.240
TIRE_both [26]	0.702±0.044	0.856±0.058	0.830±0.079	0.877±0.071	0.817±0.046	0.69±0.13	0.480±0.027	0.736±0.207
MC-TIRE_TD	0.955±0.023	**0.916±0.037**	0.858±0.057	**0.900±0.044**	0.843±0.057	0.65±0.12	0.491±0.037	**0.913±0.012**
MC-TIRE_FD	0.935±0.039	0.889±0.044	**0.952±0.034**	0.742±0.059	**0.948±0.031**	**0.81±0.1**5	**0.569±0.014**	0.859±0.020
MC-TIRE_both	**0.956±0.034**	0.911±0.037	0.925±0.037	0.898±0.043	0.941±0.046	0.75±0.13	0.551±0.054	0.903±0.006

the evaluation results of the MC-TIRE and other baseline methods.

The evaluation results reveal some interesting trends. Firstly, the MC-TIRE model demonstrates a significant improvement on the simulated datasets containing only coherence change points (change-A, change-S (JM), and change-S (SV)) compared to all the baselines in our comparison. This trend is also evident when comparing MC-TIRE to the original TIRE model, highlighting the effectiveness of the MC-TIRE framework in capturing changes in the coherence across different channels. The effectiveness of the MC-TIRE model in addressing changes in coherence across channels is further supported by observations in the first subplot of Fig. 3. The coherence branch in MC-TIRE (green) shows clear local maxima in the neighborhood of the change points, as opposed to the individual TIRE models in the residual branch (orange). This indeed aligns with the design objectives of our model.

Furthermore, MC-TIRE also outperforms TIRE on the datasets change-B (JM) and change-B (SV). This could be attributed to the use of independent per-channel TIRE models for each channel in the residual branch, providing more insights into each channel compared to the original TIRE model, which takes concatenated windows across all channels as input.

Consistently, MC-TIRE yields superior detection results on the real-life datasets compared to the baseline methods. By combining the results from both simulated and real-life datasets, we can see that the combination of information in the TD and FD via (13) typically provides adequate detection performance. Although these results are sometimes not as strong as those obtained with domain-specific information (i.e., where the end-user selects either TD or FD as the input representation), the combination strategy employed in TIRE and MC-TIRE still offers a viable approach to merging information from both domains when no expert knowledge is available to determine whether the TD or FD better emphasizes the change points.

In addition, the baseline models face challenges in handling multi-channel time series, and none of them consistently produce satisfactory detection results on all benchmark datasets. This issue becomes particularly noticeable in the UCI-test datasets, which contain a larger number of channels compared to other datasets. Compared to the GLR model, the RuLSIF model tends to deliver better detection performance on the majority of datasets, as it removes the limitation imposed by the assumed parameter models. The ABD model can be seen as a special case of the TIRE structure with a zero weight for the TI loss. By comparing the performance of ABD and the TIRE model, we can observe the benefits of introducing TI regularization in the feature space and the incorporation of frequency domain information. As a TSS algorithm, the Floss model excels at segmenting sequential recordings based on shape and demonstrates impressive performance on the HASC-2011 dataset, as intended. However, on other benchmark datasets where changes are mostly expressed in the statistics and lack clear regimes in shape, the Floss model typically loses its effectiveness. This conclusion aligns with the authors' arguments in [12] and with results in the single-channel CPD context as presented in [29].

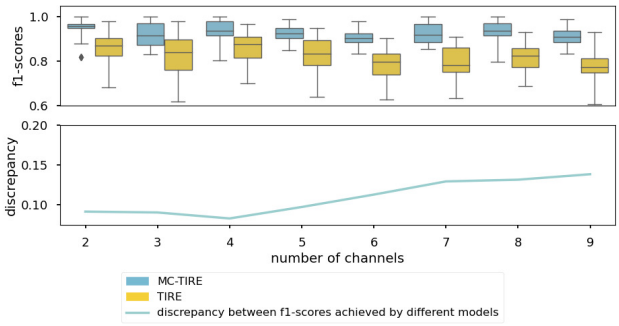
VI. ABLATION STUDY

In this section, we conduct a series of experiments to uncover the specific features of MC-TIRE that contribute to its superior change point detection (CPD) performance compared to the original TIRE model introduced in [26].

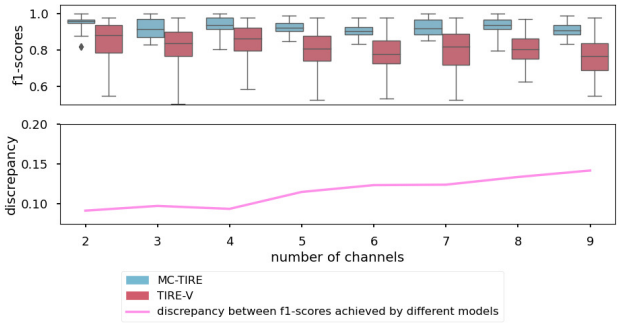
A. Leveraging Multi-channel Information

In this subsection, we compare the effectiveness of the MC-TIRE model with the original TIRE model, which lacks explicit multi-channel coherence modeling. To ensure a fair comparison, we also introduce another variant called TIRE-V. TIRE-V adjusts the number of latent TI features to match the number of channels in the input time series, aligning the dimensions of feature vectors for both MC-TIRE and TIRE-V.

We assess the detection performance with respect to the number of channels in the processed time series. Our simulated datasets consist of eight realizations, each with a varying number of channels from 2 to 9. Running MC-TIRE, TIRE, and TIRE-V on these realizations allows us to examine the influence of the number of channels.



(a) MC-TIRE vs. TIRE



(b) MC-TIRE vs. TIRE-V

Fig. 4. Comparison of the influence of channel numbers on the detection performance achieved by MC-TIRE and its competitors. The box plots show f1-scores achieved by MC-TIRE and its competitors across datasets and different runs. The lines below them visualize the differences between the mean values of f1-scores (positive values correspond to MC-TIRE outperforming its competitor)

Fig.4 displays the f1-scores, with the box plots illustrating the spread of f1-scores across five simulated datasets and 10 runs for each dataset. Additionally, we present the differences in mean f1-scores achieved by the different models. For more detailed visualizations specific to individual simulated datasets, please refer to the supplementary materials.

Fig.4 reveals an increasing trend in the improvement introduced by MC-TIRE over the original TIRE model as the number of channels increases. Furthermore, the MC-TIRE consistently outperforms the TIRE model on all simulated multi-channel time series. Despite the adjustment of the number of latent features in the TIRE-V model to match the input channels, it still lags behind the MC-TIRE model, which is inherently tailored for multi-channel time series. This underscores that the MC-TIRE model excels in leveraging multi-channel information, aligning with our design objectives.

B. Coherence versus Residual Change Points

To validate the distinct roles of the two parallel branches in MC-TIRE, designed for modeling coherence and residual features, we conduct a separate analysis of the detection results from each branch across all our simulated datasets.

Fig.5 demonstrates that the coherence branch consistently outperforms the residual branch in datasets with coherence change points, such as change-A, change-S (JM), and change-S (SV). Conversely, this relationship is reversed in the change-

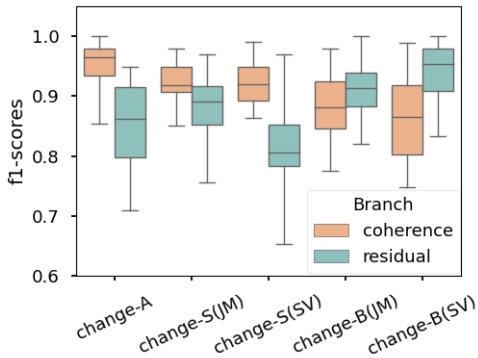


Fig. 5. Comparison between the coherence and residual branches on simulated datasets.

B (JM) and change-B (SV) datasets, which contain change points exclusive to single channels at each t_n. This observation affirms that each branch indeed targets different types of change points. However, it is important to note that the detection results in both branches are not mutually exclusive; coherence changes can sometimes be detectable in the residual branch and vice versa. This phenomenon suggests that both branches are to some extent influenced by change points of a different type than they were originally designed for. This is somewhat expected, given the choice of $R = 1$ in MC-TIRE, which is an underestimation of the true rank of the matrix $\mathbf{A}$ used to generate the data. Consequently, the coherence branch can only capture a portion of the coherent activity in the signal. Furthermore, changes occurring in one specific channel usually also affect the across-channels to some degree. While this is not problematic, it underscores the need for the mechanism to combine the decisions of both branches when they make conflicting decisions, as summarized in subsection III-E.

C. Influence of the Candidates Combination Pipeline

In Fig.2, we presented a pipeline designed to combine candidate change points detected by different branches within the MC-TIRE model. To assess the necessity of this pipeline, we conducted experiments with two alternative combination strategies and compared their performance. Specifically, we compared the proposed pipeline with the following strategies:

- "and" rule: the detection alarms are found by taking the intersection of candidate change point sets $\mathbf{P}_{coh}$ and $\mathbf{P}_{coh}$.
- "or" rule: the detection alarms are found by taking the union of candidate change point sets $\mathbf{P}_{coh}$ and $\mathbf{P}_{coh}$.

We evaluate these strategies with two recordings, one is from the change-B(JM) dataset and the other one is from the Honeybee Dance dataset. We collect results over 10 runs. The mean values and standard deviations of precision, recall, and f1-score are given in TABLE III.

As illustrated in Table III, the "and" rule strategy typically yields the highest precision among the three strategies. This suggests that maintaining consistency between both branches contributes to reducing the false-positive rate. However, this strategy comes at the expense of an increased false-negative rate. Conversely, the "or" rule strategy enhances recall

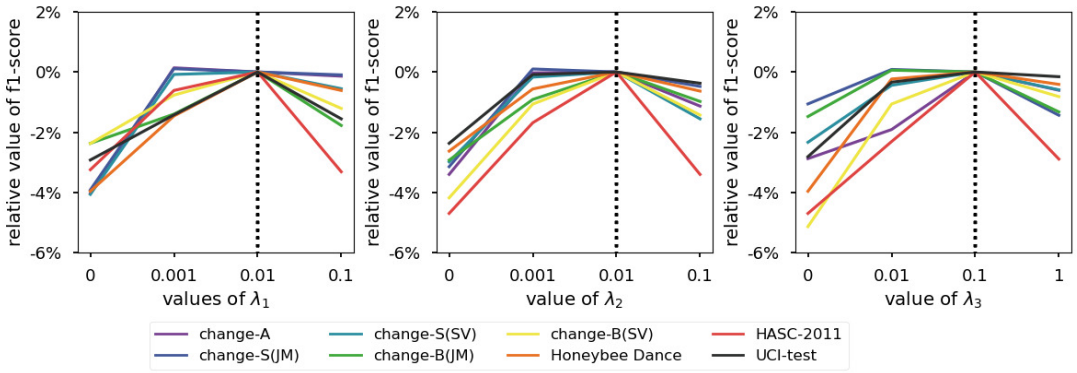


Fig. 6. Influence of the loss weights to the detection performance of MC-TIRE. The default values are marked with the vertical dotted lines.

TABLE III
COMPARISON OF CANDIDATE CHANGE POINTS MERGING STRATEGIES.

		precision	recall	f1-score
recording from change-B (JM)	"and" rule	**0.957±0.014**	0.714±0.014	0.817±0.008
	"or" rule	0.529±0.016	**0.977±0.011**	0.687±0.015
	proposed pipeline	0.872±0.033	0.955±0.017	**0.912±0.021**
recording from Honeybee Dance	"and" rule	**0.66±0.05**	0.59±0.06	0.62±0.04
	"or" rule	0.38±0.03	**0.86±0.06**	0.53±0.04
	proposed pipeline	0.60±0.02	0.83±0.04	**0.70±0.03**

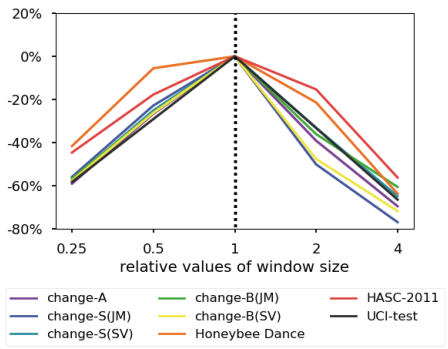


Fig. 7. Influence of the window size to the detection performance of MC-TIRE.

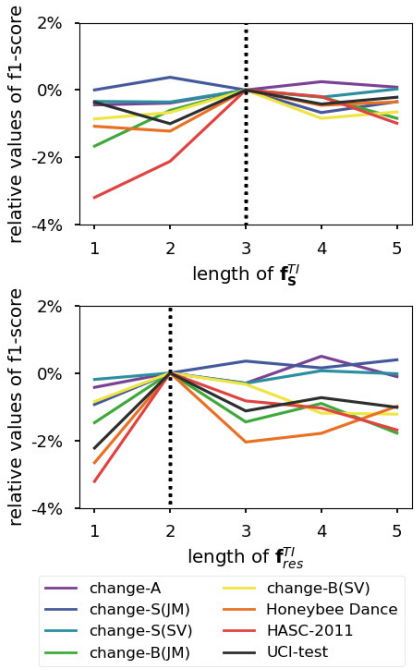


Fig. 8. Influence of the length of $\mathbf{f}_{\mathbf{S}}^{TI}$ and $\mathbf{f}_{res}^{TI}$ to the detection performance of MC-TIRE. The default values are marked with the vertical dotted lines.

while sacrificing precision. Our proposed combination pipeline strikes a balance between precision and recall, combining the best of both worlds. This approach results in a superior f1-score, showcasing its ability to achieve a harmonious trade-off between precision and recall.

D. Sensitivity Analysis

In this subsection, our objective is to assess the dependence of the proposed method's performance on the hyperparameters outlined in Subsection IV-C. Specifically, we investigate the influence of several key hyperparameters, including the window size (N), the weights assigned to each loss term, as well as the number of TI features in both the source matrix ($\mathbf{f}_{\mathbf{S}}^{TI}$) and the residual branch ($\mathbf{f}_{res}^{TI}$).

The first type of hyperparameters we are concerned with is the loss weights in equation (12). To assess the impact of these loss terms, we conducted a series of extensive experiments. In these experiments, we systematically varied the value of each loss weight while keeping the other two weights constant. Since the variations in f1-scores resulting from changes in weight values are relatively small compared to the differences among datasets, we re-scale the f1-scores. Specifically, we set the f1-scores achieved with the default weight values for each dataset as the baseline and report the relative f1-scores achieved with other weight settings. This relative f1-score is defined as the increased ratio between the f1-score obtained with the new weight setting and the f1-score achieved with the default weight value. A negative value suggests that the new weight setting leads to a reduction in the f1-score. To avoid clutter in the presentation of results across datasets, we display the averaged relative f1-scores without standard

deviations across all realizations and runs for the benchmark datasets in Fig.6. Fig.6 illustrates that the performance of the MC-TIRE model improves across all benchmark datasets when non-zero values are assigned to the loss weights. This validates the significance of incorporating each loss term as defined in equation (12). Additionally, we observe that the MC-TIRE model is relatively insensitive to variations in the values of these loss weights. Even when the loss weights are adjusted to be 10 times or one-tenth of their default values, the differences in relative f1-scores remain consistently below 5% across all benchmark datasets.

We continue with assessing the impact of the chosen window size (N). In the MC-TIRE model, N defines the length of each input window to the model, as well as the length of the moving average filter in post-processing, as inspired by [26]. To gauge the sensitivity of MC-TIRE to this hyperparameter, we present in Fig.7 the averaged relative f1-scores across all realizations and runs for the benchmark datasets. We scale the default N using coefficients 0.25, 0.5, 1, 2, and 4. Since the default value of N varies among datasets, we depict the x-axis in Fig.7 as the relative window size, denoting the scale coefficient. As observed, both larger and smaller values of N lead to a decrease in the detection performance of the MC-TIRE model. This phenomenon can be attributed to two factors. Firstly, an excessively small N might not adequately capture the statistics of the segments, while an overly large N could result in windows containing multiple adjacent change points. Secondly, a very short moving average filter may not effectively eliminate false alarms, while a very long one could render nearby peaks in the dissimilarity measure indistinguishable. Similar phenomena are also observable in the original TIRE model [26].

Furthermore, we examined the impact of varying the length of $\mathbf{f}_{\mathbf{S}}^{TI}$ and $\mathbf{f}_{res}^{TI}$. Similar to Fig.6, we present the averaged relative f1-scores for different settings in Fig.8. It is evident that having a small number of TI features in both $\mathbf{f}_{\mathbf{S}}^{TI}$ and $\mathbf{f}_{res}^{TI}$ can lead to a decrease in performance. This is primarily because a limited number of TI features increases the likelihood of encoding change-relevant information into the TV features. However, the performance of the MC-TIRE model remains stable across the majority of datasets as we increase the length of $\mathbf{f}_{\mathbf{S}}^{TI}$ and $\mathbf{f}_{res}^{TI}$.

VII. Conclusion

We introduced the Multi-Channel Time-Invariant Representation (MC-TIRE) model for change point detection in multi-channel time series. In contrast to other methods, MC-TIRE explicitly models and leverages the coherent structure across channels to enhance change point detection in multi-channel time series. The proposed model comprises two branches: a coherence branch that captures the majority of the coherent signal activity and a residual branch, which handles the remaining activity. Additionally, we presented a heuristic approach for merging potentially conflicting alarms from both branches without significantly increasing the false alarm rate. Across extensive evaluations on both simulated and real-life benchmark datasets, the MC-TIRE model consistently outperforms state-of-the-art CPD methods, including the original TIRE model. This demonstrates the effectiveness of MC-TIRE in achieving superior CPD performance in the setting of multi-channel time series.

Acknowledgement

We utilized the Chat Generative Pre-trained Transformer (ChatGPT), a large language model-based chatbot, for proof-reading our manuscript and condensing its content to meet the page limitations. No text was generated by ChatGPT from scratch.

References

[1] P. Yang, G. Dumont, and J.M. Ansermino. Adaptive change detection in heart rate trend monitoring in anesthetized children. *IEEE Transactions on Biomedical Engineering*, 53(11):2211–2219, 2006.

[2] Rakesh Malladi, Giridhar P Kalamangalam, and Behnaam Aazhang. Online bayesian change point detection algorithms for segmentation of epileptic activity. In *2013 Asilomar Conference on Signals, Systems and Computers*, pages 1833–1837. IEEE, 2013.

[3] Md Foezur Rahman Chowdhury, S-A Selouani, and D O'Shaughnessy. Bayesian on-line spectral change point detection: a soft computing approach for on-line asr. *International Journal of Speech Technology*, 15(1):5–23, 2012.

[4] David Rybach, Christian Gollan, Ralf Schluter, and Hermann Ney. Audio segmentation for speech recognition using segment features. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4197–4200. IEEE, 2009.

[5] Jushan Bai and Pierre Perron. Estimating and testing linear models with multiple structural changes. *Econometrica*, pages 47–78, 1998.

[6] Marc Lavielle and Gilles Teyssiere. Adaptive detection of multiple change-points in asset price volatility. In *Long memory in economics*, pages 129–156. Springer, 2007.

[7] Ian Cleland, Manhyung Han, Chris Nugent, Hosung Lee, Sally McClean, Shuai Zhang, and Sungyoung Lee. Evaluation of prompted annotation of activity data recorded from a smart phone. *Sensors*, 14(9):15861–15879, 2014.

[8] Haeran Cho and Piotr Fryzlewicz. Multiple-change-point detection for high dimensional time series via sparsified binary segmentation. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pages 475–507, 2015.

[9] Jaxk Reeves, Jien Chen, Xiaolan L Wang, Robert Lund, and Qi Qi Lu. A review and comparison of changepoint detection techniques for climate data. *Journal of applied meteorology and climatology*, 46(6):900–915, 2007.

[10] Naoki Itoh and Jürgen Kurths. Change-point detection of climate time series by nonparametric method. In *Proceedings of the world congress on engineering and computer science*, volume 1, pages 445–448. Citeseer, 2010.

[11] Yasuko Matsubara, Yasushi Sakurai, and Christos Faloutsos. Autoplait: Automatic mining of co-evolving time sequences. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 193–204, 2014.

[12] Shaghayegh Gharghabi, Chin-Chia Michael Yeh, Yifei Ding, Wei Ding, Paul Hibbing, Samuel LaMunion, Andrew Kaplan, Scott E Crouter, and Eamonn Keogh. Domain agnostic online semantic segmentation for multi-dimensional time series. *Data mining and knowledge discovery*, 33:96–130, 2019.

[13] Michele Basseville, Igor V Nikiforov, et al. *Detection of abrupt changes: theory and application*, volume 104. prentice Hall Englewood Cliffs, 1993.

[14] Ulrich Appel and Achim V Brandt. Adaptive sequential segmentation of piecewise stationary time series. *Information sciences*, 29(1):27–56, 1983.

[15] AV Brandt. Detecting and estimating parameter jumps using ladder algorithms and likelihood ratio tests. In *ICASSP'83. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 8, pages 1017–1020. IEEE, 1983.

[16] Yoshinobu Kawahara and Masashi Sugiyama. Sequential change-point detection based on direct density-ratio estimation. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(2):114–127, 2012.

[17] Song Liu, Makoto Yamada, Nigel Collier, and Masashi Sugiyama. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–83, 2013.

[18] Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access*, 7:1991–2005, 2018.

[19] Mathias Perslev, Michael Hejselbak Jensen, Sune Darkner, Poul Jørgen Jennum, and Christian Igel. U-time: A fully convolutional network for time series segmentation applied to sleep staging. *arXiv preprint arXiv:1910.11162*, 2019.

[20] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000.

[21] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[22] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[23] Zahra Ebrahimzadeh, Min Zheng, Selcuk Karakas, and Samantha Kleinberg. Pyramid recurrent neural networks for multi-scale change-point detection. 2018.

[24] Zunya Shi and Abdallah Chehade. A dual-lstm framework combining change point detection and remaining useful life prediction. *Reliability Engineering & System Safety*, 205:107257, 2021.

[25] Wei-Han Lee, Jorge Ortiz, Bongjun Ko, and Ruby Lee. Time series segmentation through automatic feature learning. *arXiv preprint arXiv:1801.05394*, 2018.

[26] Tim De Ryck, Maarten De Vos, and Alexander Bertrand. Change point detection in time series data using autoencoders with a time-invariant representation. *IEEE Transactions on Signal Processing*, 2021.

[27] Shohreh Deldari, Daniel V Smith, Hao Xue, and Flora D Salim. Time series change point detection with self-supervised contrastive predictive coding. In *Proceedings of the Web Conference 2021*, pages 3124–3135, 2021.

[28] Gerrit J. J. van den Burg and Christopher K. I. Williams. An evaluation of change point detection algorithms, 2020.

[29] Arne De Brabandere, Zhenxiang Cao, Maarten De Vos, Alexander Bertrand, and Jesse Davis. Semi-supervised change point detection using active learning. In *International Conference on Discovery Science*, pages 74–88. Springer, 2022.

[30] Zhenxiang Cao, Nick Seeuws, Maarten De Vos, and Alexander Bertrand. A semi-supervised interactive algorithm for change point detection. *Data Mining and Knowledge Discovery*, pages 1–29, 2023.

[31] Wei-Cheng Chang, Chun-Liang Li, Yiming Yang, and Barnabás Póczos. Kernel change-point detection with auxiliary deep generative models. *arXiv preprint arXiv:1901.06077*, 2019.

[32] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 1317–1322. Ieee, 2016.

[33] Sang Min Oh, James M Rehg, Tucker Balch, and Frank Dellaert. Learning and inferring motion patterns using parametric segmental switching linear dynamic systems. *International Journal of Computer Vision*, 77(1):103–124, 2008.

[34] Kevin C Cheng, Eric L Miller, Michael C Hughes, and Shuchin Aeron. On matched filtering for statistical change point detection. *IEEE Open Journal of Signal Processing*, 1:159–176, 2020.

[35] Ryan D. Turner. Gaussian processes for state space models and change point detection. 2012.

[36] Xiang Xuan and Kevin Murphy. Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 1055–1062, New York, NY, USA, 2007. Association for Computing Machinery.

[37] Nobuo Kawaguchi, Ying Yang, Tianhui Yang, Nobuhiro Ogawa, Yohei Iwasaki, Katsuhiko Kaji, Tsutomu Terada, Kazuya Murao, Sozo Inoue, Yoshihiro Kawahara, Yasuyuki Sumi, and Nobuhiko Nishio. Hasc2011corpus: Towards the common ground of human activity recognition. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, UbiComp '11, page 571–572, New York, NY, USA, 2011. Association for Computing Machinery.

[38] Kevin C Cheng, Shuchin Aeron, Michael C Hughes, Erika Hussey, and Eric L Miller. Optimal transport based change point detection and time series segment clustering. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6034–6038. IEEE, 2020.

[39] Kevin C. Cheng, Shuchin Aeron, Michael C. Hughes, Erika Hussey, and Eric L. Miller. Optimal transport based change point detection and time series segment clustering, 2020.

[40] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra Perez, and Jorge Luis Reyes Ortiz. A public domain dataset for human activity recognition using smartphones. In *Proceedings of the 21th International European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 437–442, 2013.

Zhenxiang Cao received the BSc degree from the Beijing University of Technology, China, in 2013, and the MSc degree from Technische Universität München (TUM), Germany, in 2017. Currently, he is working toward the PhD degree in the faculty of electrical engineering (ESAT) at KU Leuven under the supervision of Alexander Bertrand and Maarten De Vos. His research interests include machine learning, data mining, deep learning as well as their applications in time series analysis.

Nick Seeuws holds MSc degrees in mathematical engineering and artificial intelligence from KU Leuven, Leuven, Belgium. He is currently pursuing a PhD degree in electrical engineering at KU Leuven under the supervision of Alexander Bertrand and Maarten De Vos. His research interests are machine learning, signal processing, and biomedical engineering.

Maarten De Vos has a joint appointment as BOFZAP in the Departments of Engineering and Medicine at KU Leuven after being Associate Professor at the University of Oxford, United Kingdom, and Junior Professor at the University of Oldenburg, Germany. He obtained an MSc (2005) and PhD (2009) in Electrical Engineering from KU Leuven, Belgium. His academic work focuses on AI for healthcare monitoring and innovative biomedical monitoring for daily life applications. He is renown for the derivation of personalised biosignatures of patient health and the incorporation of smart analytics into wearable sensors.

His pioneering research in the field of mobile real-life brain-monitoring has won several innovation prices, among which the prestigious Mobile Brain Body monitoring prize in 2017, the Children's Prize for best childhood innovation in 2018, the Martin Black Prize for the best paper in Physiological Measurements in 2019 and in 2021 he received the IEEE EMBS Benelux award for best paper in the biomedical field.

He has been guest editor for International Journal of Clinical Neurophysiology, Sensors and Physiological Measurements and is currently on the editorial board of Journal of neural engineering, IEEE journal of biomedical health informatics and Nature Digital Health.

Alexander Bertrand (Senior member IEEE) received the Ph.D. degree in electrical engineering from KU Leuven, in 2011. He is currently Associate Professor at the Electrical Engineering Department (ESAT) of KU Leuven, Belgium. He was a Visiting Researcher at the University of California, Los Angeles (in 2010) and at the University of California, Berkeley (in 2013). His research involves multi-channel and distributed signal processing algorithm design with a focus on biomedical sensor arrays and sensor networks, with applications in, a.o., brain-computer interfaces and hearing prostheses. Dr. Bertrand is recipient of an ERC starting grant (2018), the 2012 FWO/IBM Award, the 2013 KU Leuven Research Council Award (Science & Technology), and was awarded several research fellowships and grants. He is currently Associate Editor for the IEEE Transactions on Signal Processing (2nd term), and a recurrent Associate Editor for the yearly international conference of the IEEE Engineering in Medicine and Biology Society (IEEE EMBC 2016–2022). He is currently a member of the Sensor Array & Multi-channel (SAM) technical committee and a board member of the Benelux chapter of the IEEE Signal Processing Society.