Towards Uncertainty Quantification for Time Series Segmentation

Erick Draayer
edraayer@nmsu.com
Computer Science, New Mexico State University
Las Cruces, New Mexico, USA

Huiping Cao
hcao@nmsu.com
Computer Science, New Mexico State University
Las Cruces, New Mexico, USA

Abstract

Time Series Segmentation (TSS) is a data mining task widely used in many applications to generate a set of change points for a time series. Current TSS performance analyses focus on accuracy and, therefore, fail to fully evaluate the reliability and originality of a segmentation. We investigate using uncertainty quantification (UQ) to fully evaluate TSS performance. We propose UQ-TSS, a framework to quantify uncertainties surrounding TSS. UQ-TSS captures uncertainties from different sources in an integrative manner. It incorporates a novel TS augmentation algorithm to address inherent uncertainty in the data. It uses ensemble learning in a novel way to create samples and estimate the probability distributions of changepoint presence and locations. We demonstrate the ability of UQ-TSS to guide hyperparameter selection, refine segmentations, and determine an algorithm's suitability for segmenting without the need for ground truth. We validate these claims through extensive experimentation using several well-established TSS algorithms and datasets.

CCS Concepts

• **Information systems** → Clustering; • **Computing methodologies** → *Ensemble methods*; **Uncertainty quantification**; • **Mathematics of computing** → **Time series analysis**.

Keywords

Time Series Segmentation, Changepoint Detection, Uncertainty Quantification, Time Series Augmentation

ACM Reference Format:
Erick Draayer and Huiping Cao. 2024. Towards Uncertainty Quantification for Time Series Segmentation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3627673.3679652

1 Introduction

Time series segmentation (TSS) is the data mining task of detecting changes between homogeneous data-generative states within a given time series (TS). For example, a recording of a person's movements can be segmented into walking, running, and sitting activities. TSS aims to detect changes in activities as a set of changepoints (CPs). The value of a CP indicates the location where the change occurred. TSS is synonymous with the problem of changepoint detection (CPD), and both methods are often grouped together in benchmark studies [36, 37]. Many TSS algorithms exist for specific domains, such as human speech [24], but the scope of our research focuses on domain-agnostic TSS.

The output of TSS usually feeds into other analytical tasks. For example, TS classification takes the segments obtained from TSS as input. Errors from TSS propagate into these tasks, so it is imperative to develop methodologies and measures to help users better understand the suitability of a segmentation. However, current TSS performance analysis focuses on using accuracy metrics that fail to assess the stability and distinctiveness of a segmentation fully.

We propose designing uncertainty quantification (UQ) techniques to help evaluate and gain insights into a TSS algorithm's performance. UQ can provide many benefits for TSS, which we discuss shortly. UQ is well established in classical statistics but is only recently being explored by the broader data science community [10]. UQ is not simply about attaching probabilities to outcomes. Probability measures the likelihood of certain outcomes, but uncertainty characterizes the distribution over those possible outcomes [33]. This entails treating the outcomes of a model as random variables, generating samples, and estimating probability distributions for characterization and measuring uncertainty.

UQ for TSS has several challenges that need to be addressed. One challenge is identifying the various sources of uncertainty associated with TSS. Each source requires its own strategies to capture its uncertainty. Capturing uncertainty from the data is especially challenging because it relies on perturbing the data in a reasonable manner. Current methods are not designed for TSS. They either distort the data too much or cannot be generalized. After capturing uncertainty, there are challenges to quantifying it for the various components of TSS. These components are: (1) the presence of a CP, (2) the CP location, and (3) the overall segmentation. There are also issues on how to interpret uncertainty. Uncertainty offers a different perspective than accuracy measures like the F_1-score and can lead to an opposite conclusion about a TSS algorithm's performance. We propose UQ-TSS to address these challenges and quantify TSS uncertainty.

To the best of our knowledge, UQ-TSS is the first generalized UQ method that can be used to infer the uncertainty of any given TSS method and dataset. UQ-TSS captures uncertainty from several sources of TSS, including from any dataset. UQ-TSS accomplishes this by using a novel TS augmentation algorithm that can perturb any TS for UQ despite different trends, seasonality, and noise hidden in the data. UQ-TSS uses a computationally efficient ensemble learning approach to generate samples

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679652

from multiple segmentations. These samples are used to estimate probability distributions for CP presence and location. These distributions allow UQ-TSS to accurately estimate the uncertainty of CPs' presence, their locations, and the overall segmentation. UQ-TSS is generalizable and usable with any TS dataset and TSS algorithm.

The uncertainty measures calculated by UQ-TSS provide many benefits. These measurements allow us to evaluate a segmentation without needing ground truth. This is especially beneficial because TSS is often an unsupervised method employed where ground truth is missing. We can also use UQ to guide hyperparameter selection, refine the segmentation, and create different interpretations of CP locations. We showcase these benefits in Section 4.

Our contributions are outlined below:

- We propose UQ-TSS, a UQ framework for TSS designed for use with any TS and TSS algorithm. UQ-TSS allows users to estimate the uncertainty of the presence of CPs, the CP locations, and the overall segmentation.
- We analyze the different sources of uncertainty for TSS and design mechanisms and strategies to estimate uncertainty from each. This includes a novel TS data augmentation algorithm for estimating TS data uncertainty.
- UQ-TSS provides many benefits including segmentation refinement, creating different interpretations for CP locations, evaluation measures that do not rely on ground truth, and guidance for hyperparameter selection.

The paper is organized as follows. Section 2 discusses the background and related works. Section 3 presents the proposed UQ-TSS framework. Section 4 provides extensive experimentation and discussion to validate and showcase the benefits of UQ for TSS. We also provide the code and datasets for replication of our experiments [11].

2 Background and Related Works

Domain-agnostic TSS research is surprisingly sparse. The most recently proposed methods include FLOSS [15] and ClaSP [12]. Benchmark studies have also shown older methods like BOCPD [1] remain competitive. However, these works ignore any aspect of UQ. Peterson et al. [26] is the only research to explore UQ for TSS, but their work is specific to seismic onset times. To the best of our knowledge, UQ-TSS is the first generalized UQ method that can be used to infer the uncertainty of any given TSS method and dataset.

There are very few UQ works towards TSS. Guédon proposes a novel TSS method that utilizes UQ to determine segmentations [16]. Their TSS method uses Bayesian statistics to estimate CP location distributions and measure the overall uncertainty of their segmentations. However, their UQ calculations cannot be generalized to other TSS methods. In contrast, we design our proposed method to be both model-agnostic and domain-agnostic. Bors et al. [6] propose a method to visualize the uncertainty of a segmentation. They generate colored heatmaps based on stacked segmentation derived from various hyperparameter settings. Their research is primarily concerned with visual analytics and focuses on inferring the uncertainty of a CP's presence due to uncertainty in hyperparameter selection. They do not analyze uncertainty from other major sources associated with TSS.

A variant of TSS is trajectory segmentation [5]. Trajectory segmentation typically focuses on applications towards domains of traffic and transportation analysis [2, 25]. The uncertainty estimations derived from trajectory segmentation revolve around the spatial dimensions associated with trajectory segmentation [5, 21]. Overall, these works are too domain-specific for segmentation and uncertainty estimation of purely temporal data.

A branch of TSS research tries to detect gradual changes based on principles of fuzzy set theory. For example, Bhaduri et al. [4] use rough-fuzzy set theory to model the uncertainty associated with gradual CPs for detection. These proposed methods may model vagueness and calculate probabilities to detect spans of gradual change. However, they do not attempt to analyze or fully measure and characterize the variance of their output, which is the goal of UQ.

UQ has greatly benefited different types of data mining and machine learning tasks. Kendall et al. [18] used UQ to create a new loss function for Bayesian deep learners to improve image segmentation performance. Pasaros et al. [27] explore UQ for neural networks to create new evaluation metrics and post hoc performance improvement techniques. Nemani et al. [23] study ways UQ can be used to solve and gain insight into engineering design and health prognostic problems.

Our research avoids categorizing sources of uncertainty as aleatoric or epistemic. These terms and their calculations are rooted in information theory and often used in other UQ research [9]. However, recent research has found several incoherencies in data mining and machine learning applications [39]. For this reason, we avoid them in our research and instead focus on the core idea of UQ, which is characterizing the distributions of outcomes [33].

Our proposed method relies on a novel TS data augmentation method tailored for UQ. We briefly review the related works for TS data augmentation here. Iglesias et al. [17] and Wen et al. [38] provide a comprehensive TS data augmentation methods taxonomy. TS augmentation methods fall into the following categories: slicing, jittering, scaling, rotation, permutation, statistical generative models, learning models, permutation, and decomposition. Jittering is the most naive approach, as it simply adds random noise to the TS. We avoid this approach because its degree of perturbation depends on the data-generative state within the TS. There is a major difference between the augmented and original TS that may add a bias to the data. Another reason we avoid jittering is that adding noise to low noise level states may have greater effects than higher noise states. For example, adding the same level of noise to a regime of someone sitting versus running in an accelerometer recording perturbs the sitting regime much more than the running. Slicing, rotation, and perturbation augmentation methods reorder the temporal dependencies of the TS. Since TSS datasets typically have several data-generative states, this reordering may cause some states to mix with others. This mixing generally leads to false CP detection. Scaling and statistical generative models require hyperparameter tuning. This tuning can be sensitive, and these methods tend to work only for specific data domains. Statistical generative models rely on prior knowledge of the TS and can become very domain-specific. Learning models require training data and ground truth. Decomposition methods separate the TS into seasonality, trend, and residuals. Decomposition methods currently focus on applications towards classification, forecasting, or anomaly detection. [3, 17]. They are not designed for TSS and can drastically distort a TS with several data-generative states.

3 UQ-TSS Framework

This section presents UQ-TSS, our proposed method for measuring TSS uncertainty. We first introduce the terminology.

Let a TS be represented as $T = \{t_1^d, t_2^d, t_3^d, ..., t_n^d\}$ where n is the number of observations and each observation is on d dimensions. Let $T[i]$ denote the n observations on dimension i. Given a time series T, a TSS algorithm returns a set of CPs, which is represented as $\Omega = \{\alpha_1, \alpha_2, \alpha_3, ..., \alpha_i, ...\}$ ($\alpha_i \in [1, n]$). The value of α_i indicates its location within T. UQ-TSS finds the range of possible outcomes for each $\alpha \in \Omega$ and uses them as samples to estimate probability distributions and derive uncertainty. We define these samples as a set, $\Omega' = \{C_1, C_2, ..., C_k\}$, where C_i contains the samples corresponding to CP α_i. The samples are generated from an ensemble with M members. M is a hyperparameter that users need to set.

3.1 Capture of Different Uncertainty Sources

We identify four sources of uncertainty in TSS: (1) the TS data, (2) data preprocessing, (3) algorithm calibration, and (4) inference of indicators for CPs. We discuss these sources and our methods for capturing their uncertainty in detail below. In general, many real-world scenarios of TSS need to handle all these sources of uncertainty, but some may not. UQ-TSS is designed to omit any source of uncertainty from its calculations freely.

3.1.1 Data Uncertainty. One source of uncertainty comes from the TS data itself. Each recorded observation has a random measurement error associated with it, also known as noise. Noise causes uncertainty in CP presence and location.

Many TS augmentation algorithms [17] exist for generating alternate versions of a TS. These algorithms are not appropriate to augment TS because of the reasons discussed in Section 2.

We propose a novel TS augmentation algorithm that particularly supports UQ for TSS. We design our algorithm to isolate noise and perturb it so that the underlying features of multiple regimes within a TS are preserved. Our technique separates the noise from the TS and perturbs it on an individual scale. Our algorithm prevents regimes from inheriting others' properties and noise biases while also inducing differences in TSS. Our algorithm also makes no assumptions about the TS and can be applied to any data domain for TSS.

Algorithm 1 shows the pseudocode for our TS augmentation algorithm. Our algorithm decomposes a given TS T using a 1-dimensional convolution operator across each dimension (Line 4). The convolution window averages w values within T to construct T'. We then isolate the noise by subtracting T' from T (Line 6). Each noise value is multiplied by an independent and identically distributed random value sampled from a continuous uniform distribution between 0.5 to 1.5 (Line 9). This step amplifies approximately half the noise (multiplied by a value between 1.0 and 1.5) and decimates the other half by the same amount (multiplied by a value between 0.5 to 1.0). The array of perturbed noise values is added to T' to yield $\hat{T}$, the augmented TS (Line 13).

Algorithm 1 UQ-TSS data augmentation.

```
1:  function AugmentTS($T$ , $w$)
2:      $window = [\frac{1}{w}, \frac{1}{w}, ..., \frac{1}{w}]$ of size $w$
3:      for $i = 1$ to $d$ do
4:          $T'[i] \leftarrow$ Convolution($T[i], window$)
5:      end for
6:      $noise = T - T'$
7:      for $i = 1$ to $d$ do
8:          for $j = 1$ to $n$ do
9:              $r \leftarrow$ Random value between 0.5 to 1.5
10:             $noise'[i][j] = noise[i][j] * r$
11:         end for
12:     end for
13:     $\hat{T} = T' + noise'$
14:     return $\hat{T}$
15: end function
```

The user can adjust the range for r. The minimum and maximum values need to be equal distance from 1.0; if not, the original TS's overall noise will be distorted. This means the maximum range for r is $0 - 2.0$. We found that ranges from $0.5 - 1.5$ to $0 - 2.0$ yield similar uncertainty estimations. Ranges too small fail to perturb the TS at all for uncertainty estimation.

Our decomposition method is relatively simple compared to others like Seasonal and Trend decomposition using Loess (STL) [28]. However, these methods require domain knowledge about the TS, such as its seasonality. In TSS, a dataset with multiple activities can have multiple seasonalities, reducing the effectiveness of these more advanced decomposition methods.

3.1.2 TSS Algorithm Uncertainty. We identify TSS algorithm uncertainty as the uncertainty surrounding hyperparameter calibration, not the selection of the algorithm itself. For UQ, we are interested in how minor adjustments in the hyperparameters affect TSS results. We are not interested in breaking the algorithm with extreme hyperparameter values outside the range of recommendations. We capture TSS algorithm uncertainty by testing a set of candidate values for each hyperparameter.

The hyperparameter candidate sets are defined based on a user's uncertainty. For example, let x be some hyperparameter, and guidelines suggest its value should be between 150 and 250. A user can capture this uncertainty with UQ-TSS by defining a candidate set as $\{150, 175, 200, 225, 250\}$. Ad hoc experimentation can reveal that each value produces similar segmentations to its neighbor. Defining the candidate values as a discrete set instead of a continuous one better reflects how a user would test a range of potential settings. Defining the candidate set is situational, but this is the general procedure. We provide a detailed example using a real dataset in Section 4.2.

3.1.3 Inference Uncertainty. Many TSS algorithms infer CP detections from a score over the TS. Local minimums/maximums imply a change has occurred. Figure 1 shows an example from the TSS method FLOSS [14]. The TS is labeled with one CP at timestamp 8000, and FLOSS detects that CP with its score over time. However, arguments could easily be made to include timestamps 12000 and

16000 as CPs. This uncertainty about which minimums/maximums are strong enough to be considered CPs is inference uncertainty.

In full-scale pipelines, these decisions fall upon peak detection algorithms. A Peak detection algorithm may use a variety of hyperparameters, but all typically share a *prominence* hyperparameter that controls its sensitivity. The *prominence* is a threshold value that describes the minimum prominence a peak should have. The value of the *prominence* depends on the peak detection algorithm but is typically derived from some combination of the width and depth of the peak. Similarly to TSS algorithm uncertainty, we are not interested in breaking the peak detection algorithm. Instead, we want to select values that include/exclude ambiguous local minimum/maximum.

We can capture inference uncertainty by defining a candidate set for *prominence* based on visual inspection of the score. For example, Figure 1 finds a CP at timestamp 8000 when *prominence*=0.25. Timestamps 12000 and 16000 are included when the sensitivity is raised to *prominence*=0.2 and *prominence*=0.15, respectively. Therefore, the candidate set for inference uncertainty should be $\{0.15, 0.2, 0.25\}$. Our framework automatically incorporates different hyperparameter values to generate different CPs, which serve as samples to estimate distributions for CP presence and location.

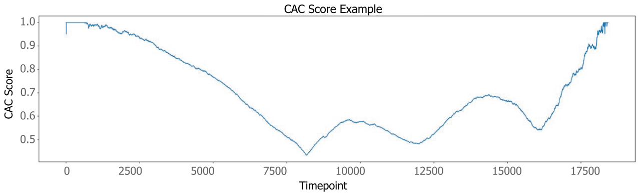

Figure 1: Example of CAC score returned by FLOSS.

3.1.4 Preprocessing Uncertainty. Uncertainty due to preprocessing refers to regularization methods associated with TSS [34]. Regularization methods are usually domain-specific. For example, digital signal recordings of someone talking may need a low-pass filter such as Butterworth or Bessel [30] to remove background noise. The list of regularization methods is extensive, often coming from several research fields, such as data science, signal processing, and statistics.

Capturing uncertainty from preprocessing requires creating a candidate set of potential regularization methods and candidate sets for their respective hyperparameters. Establishing these candidate sets is situational, and automating the approach is almost impossible. This approach may seem impractical, but other UQ research has touched on capturing preprocessing uncertainty. Peterson et al. [26] provides an example of capturing uncertainty from preprocessing. They define a reasonable candidate set of preprocessing methods and evaluate the resulting CPs for seismic onset times.

Our research considers all sources of uncertainty except preprocessing. The focus of this research is a general UQ method for TSS. As discussed before, preprocessing is very situational and domain-specific. Therefore, testing it is beyond the scope of this paper, but we still acknowledge it as a source of uncertainty.

3.2 Overview of UQ-TSS

In order to capture segmentation uncertainty, different sources must be used in conjunction with each other. Results from individual sources cannot simply be added together. This entails sampling combinations of hyperparameters from our candidate sets.

We propose an ensemble learning approach for UQ-TSS. Ensemble learning allows us to incorporate the random sampling of hyperparameters and our TS augmentation algorithm. Each ensemble member computes a segmentation using its own version of the TS generated by our augmentation algorithm and a random selection of hyperparameters from the candidate sets. The CPs from all segmentations calculated by the ensemble are grouped and treated as samples for estimating probability distributions. UQ-TSS characterizes these distributions to measure the uncertainty of CP presence, CP location, and overall segmentation.

Algorithm 2 shows the pseudocode for UQ-TSS. This algorithm has three major steps. In the first step, each ensemble member calculates a unique segmentation Ω_m ($1 \leq m \leq M$) (Lines 1-14), which consists of a list of CPs. The CPs from different ensemble members may not be the same ultimately, but they can still refer to the same CP. For example, assume three ensemble members compute the three CP sets: {24, 172}, {25, 178}, and {25}. These CPs correspond to two CPs around timestamps 25 and 175. The second step is designed based on this observation. The second step clusters the CPs from the ensemble members (Lines 15-22). Each cluster consists of CPs that serve as samples to estimate the actual CP. Clusters with few samples are removed to help improve accuracy. The third step estimates two probability distributions (Lines 23-27) from each CP corresponding to its presence and location. From these distributions, UQ-TSS calculates final CP predictions and the Shannon entropy [31], a measure of uncertainty.

The details of specific components, including ensemble learning, clustering, and prediction, are explained in the sections below.

3.3 Ensemble Learning

UQ-TSS uses randomization-based ensemble learning to combine multiple segmentations. We use this design for several reasons. Randomization-based ensemble learning has been proven to work well in other data-driven UQ research [20, 26, 27, 40]. The calculations of randomization-based ensemble learning are independent and easily parallelized. Therefore UQ-TSS does not impede the scalability of any TSS algorithm used with it. Ensemble learning has also been proven to help create more robust and accurate results in other machine learning tasks [8].

UQ-TSS creates an ensemble of M members. Each member segments their own augmented version of the TS based on a random selection from the candidate sets. The m-th ensemble member consists of four components, $\hat{T}^{(m)}$, $h^{(m)}(\lambda_1^{(m)}, \ldots)$, $f(\lambda_2^{(m)}, \ldots)$, and $g(\lambda_3^{(m)}, \ldots)$ where $\ldots$ represent irrelevant input. $\hat{T}^{(m)}$ is the augmented TS created by our novel TS augmentation algorithm (Section 3.1.1). The $h()$, $f()$, and $g()$ functions correspond to preprocessing, time series segmentation, and inference. Among these three functions, $h()$ is randomly selected from a candidate set of preprocessing functions $\mathbb{H}$, while $f()$ and $g()$ are fixed. Their hyperparameters are randomly selected from the three candidate sets Λ_1, Λ_2, and Λ_3, respectively.

Algorithm 2 UQ-TSS(T, w, M, f, g, $\mathbb{H}$, Λ_1, Λ_2, Λ_3, β)

Input: T: original time series; w: window length used for augmenting T; M: number of ensemble members; f: time series segmentation function; g: inference function; $\mathbb{H}$: candidate set of preprocessing methods; Λ_1: candidate set for $\mathbb{H}$ hyperparameters; Λ_2: candidate set for f hyperparameters; Λ_3: candidate set for g; β: threshold value filtering out small clusters

Output: Ω': the final CP predictions; w_{C_i}: the probability distributions for CP locations; q_{C_i} the probability distributions for CP presence; $\mathbb{U}$: the overall segmentation uncertainty

```
for m=1 ··· M do
    $\hat{T} \leftarrow AugmentTS(T, w)$                      ▷ Algorithm 1
    /* Select preprocess method $h()$ from set $\mathbb{H}$ */
    $h \leftarrow RandomSelection(\mathbb{H})$
    /* Select hyperparameters for $h()$ from set $\Lambda_1$ */
    $\lambda_1 \leftarrow RandomSelection(\Lambda_1)$
    $\hat{T} \leftarrow h(\hat{T}, \lambda_1)$                      ▷ Preprocess $\hat{T}$
    /* Select hyperparameters for $f()$ from set $\Lambda_2$ */
    $\lambda_2 \leftarrow Random\ Selection(\Lambda_2)$
    $score \leftarrow f(\hat{T}, \lambda_2)$                ▷ Calculate score over $\hat{T}$
    /* Select hyperparameters for $g()$ from set $\Lambda_3$ */
    $\lambda_3 \leftarrow RandomSelection(\Lambda_3)$
    $\Omega_m \leftarrow g(score, \lambda_3)$                      ▷ Find CPs
end for
$\Omega_{merged} \leftarrow [\Omega_1, \Omega_2, ..., \Omega_m, ...., \Omega_M]$
$\Omega' \leftarrow$ Cluster($\Omega_{merged}$)
/* Calculate final CP predictions */
for i=1 ··· |Ω'| do
    if $|C_i| < \beta$ then
        Remove $C_i$ from $\Omega'$
    end if
end for
for i=1 ··· |Ω'| do
    Estimate $q_{C_i}$ from $C_i$               ▷ CP presence uncertainty
    Estimate $w_{C_i}$ from $C_i$               ▷ CP location uncertainty
    $\Omega'_i \leftarrow$ Median of $C_i$                 ▷ CP prediction
end for
/* Overall Uncertainty of Segmentation */
$\mathbb{U} \leftarrow$ Average entropy of $C_i \in \Omega'$
return $\Omega', q_{C_i}, w_{C_i}, \mathbb{U}$
```

Each member calculates an array of CP predictions denoted as Ω_m. We merge each Ω_m into one array of predictions, Ω_{merged}. We view the CP predictions within Ω_{merged} as samples for estimating CP probability distributions. We need to find which samples within Ω_{merged} correspond to the same CP probability distributions. A naive approach might be to fit Gaussian mixture models to Ω_{merged}. However, this requires the number of distributions to be known and assumes these distributions are Gaussian. Instead, we opt to use density-based clustering to group predictions and post-process the clusters to estimate our CP probability distributions. Density-based clustering does not require the number of clusters to be known and makes no assumptions about the distribution shape.

3.4 Clustering CP Predictions

Given Ω_{merged}, we want to find which CP predictions correspond to the same CP across the M ensemble members. For example, an ensemble with three members may yield $\Omega_{merged} = [24, 25, 25, 172, 178]$. The CP predictions within Ω_{merged} indicate two CPs around observations 25 and 175. We use density-based spatial clustering [13] (DBSCAN) to automate this process. We selected DBSCAN because selecting its hyperparameters is very intuitive in our situation. A user only needs to know the sampling frequency, which is readily available information in most real-world situations. We explain this intuition shortly.

DBSCAN requires two hyperparameters: *radius* and *minPts*. The *radius* describes the maximum distance between CPs before they are no longer in the same neighborhood. *minPts* describes the minimum number of samples needed within a neighborhood for that sample to be considered a core sample. Core samples determine clusters, and samples are clustered together depending on whether they are within the same neighborhood as a core point. We are interested in all CP detections for UQ and, therefore, set $minPts = 1$. Tuning *radius* is usually difficult in most applications but trivial in ours because it directly references the time dimension. We can set the *radius* according to the expected length of the changes. For example, in human activity recordings, it is reasonable to assume that a change from walking to running takes no more than 30 seconds. Therefore, the *radius* should be set to 30 seconds of observations. We can also adjust *radius* according to cluster results. If distributions are found in quick succession, this suggests *radius* was set too low. If distributions span very long periods of time, this suggests that *radius* was set too high.

DBSCAN yields $\Omega' = \{C_1, C_2, ..., C_k\}$ where C_i is the i-th cluster found by DBSCAN and k is the total number of clusters found. The CPs within each cluster serve as samples for estimating our CP probability distributions and final CP predictions.

3.5 Final CP Predictions and Estimating Segmentation Uncertainty

To make UQ-TSS directly comparable to other TSS methods, we calculate its median to convert each $C_i \in \Omega'$ into a single CP prediction. UQ-TSS provides an opportunity to improve segmentation accuracy. Any CPs detected by only a minority of the ensemble can be removed. More formally, for any $|C_i| < \beta,$ remove $C_i \in \Omega'$.

3.5.1 *Measuring Changepoint Presence and Segmentation Uncertainty.*

This section describes how we derive our probability distributions for the presence of each CP and use it to measure segmentation uncertainty. Let X_i be a random variable representing the presence of the i-th CP. Given $C_i \in \Omega'$, we can estimate the probability distribution $q_{C_i}(X_i = x, p_i)$ where p_i represents the probability of $X_i = 1$. We view X_i as a binary random variable where $X_i = x, x \in \{0, 1\}$. A value of 1 indicates the presence of a CP, and 0 indicates its absence. Since X_i is binary, we can estimate $q_{C_i}(X_i = x, p_i)$ as a Bernoulli distribution defined in Equation 1.

$$q_{C_i}(X_i = x; p_i) = p_i^x * (1 - p_i)^{1-x} \text{ where } p_i = \frac{|C_i|}{M} \tag{1}$$

Since TSS algorithms are typically designed to avoid CPs near each other, we can assume $|C_i| \leq M, \forall C_i \in \Omega'$. Therefore, the probability p_i can be derived by dividing the size of cluster C_i by the size of the ensemble, M.

Since $q_{C_i}(X_i = x; p_i)$ is a Bernoulli distribution, we can calculate the Shannon entropy [31] to quantify the uncertainty of the CP's presence [32]. Shannon entropy is commonly used to measure uncertainty in other machine learning UQ research [39]. The Shannon entropy for the presence of CP X_i is derived as follows:

$$\begin{aligned} U(X_i) &= -p_i \log_2 p_i - (1-p_i) log_2(1-p_i) \\ &= -(\frac{|C_i|}{M} log_2 \frac{|C_i|}{M} - (1 - \frac{|C_i|}{M}) log_2(1 - \frac{|C_i|}{M})) \end{aligned} \tag{2}$$

Shannon entropy ranges from 0 to 1, where 1 indicates maximum uncertainty, and 0 indicates no uncertainty in a random variable's outcome. In UQ-TSS, the uncertainty is minimized when all ensemble members detect a CP (i.e., $|C_i| = M$) or only one member detects a CP (i.e., $|C_i| = 1$). Uncertainty is maximized when 50% of the ensemble detects a CP (i.e., $|C_i| = \frac{M}{2}$).

Calculating the Shannon entropy for the presence of each CP allows us to calculate the uncertainty of the overall segmentation. We propose averaging the entropies calculated for each CP presence. Equation 3 defines the segmentation uncertainty.

$$\mathbb{U}(\Omega') = \frac{1}{k} * \sum_{i=1}^{k} U(X_i) \tag{3}$$

3.5.2 Measuring Changepoint Location Uncertainty. This section defines our approach for quantifying the uncertainty of the CP locations. We use kernel density estimation [35] (KDE) to estimate the probability distribution for i-th CP location, $w_{C_i}(Y_i = y)$. We select KDE because it is computationally efficient and makes no assumptions about the shape of the probability distribution. We view the CP location for the i-th CP as a random variable denoted as Y_i. Our KDE is defined in Equation 4 where Y_i takes on the values in C_i. We use the Gaussian kernel defined as $K(u)$ in Equation 4. The Gaussian kernel highlights where CP locations coalesce, an important aspect of CP location UQ. These highlighted locations help us determine the different interpretations of where the change occurs within the TS. We provide a detailed discussion in Section 4.2. The hyperparameter h refers to the bandwidth. We automate the selection of b by using the improved Sheather Jones (ISJ) algorithm [7]. The ISJ algorithm is well suited for estimating various distributions, which we expect to find for our CP location distributions.

$$\begin{aligned} w_{C_i}(Y_i = y) &= \frac{1}{b * |C_i|} \sum_{j=1}^{|C_i|} K(\frac{y - y_j}{b}) \\ K(u) &= \frac{1}{\sqrt{2\pi}} \exp(-\frac{u^2}{2}) \end{aligned} \tag{4}$$

We can analyze w_{C_i} to infer the uncertainty of the CP location. For example, a small variance implies less uncertainty about the location of the CP and vice versa. However, this may not always be true. A flat uniform distribution may have a high variance, but it implies the change within the TS is more gradual, and trying to define it as a CP is improper. There is no uncertainty about the location of a CP; a uniform distribution just indicates that a more gradual change is occurring over that span of time. The number of modes of the distribution can also indicate uncertainty. A bimodal distribution implies two interpretations for the CP location and, therefore, more uncertainty compared to a normal distribution with a single mode. Overall, analyzing the shape of the distribution is just as important for accessing the uncertainty for the CP location as the statistical moments like variance.

4 Experiment Evaluation

We design all experiments to be reproducible, and the code is publicly available, documented, and ready to use [11].

To the best of our knowledge, this is the first study of a framework for estimating the uncertainty of any given TSS method and TS dataset. There are no other methods that we can directly compare to UQ-TSS. There are also no trivial solutions to adapt existing TSS methods for UQ. Additionally, there is no ground truth to tell us how much uncertainty a segmentation should have from a specific TSS method and dataset. Therefore, it would be difficult to say which measure of uncertainty is more correct between two generalized UQ frameworks for TSS. Instead, we opt to evaluate UQ-TSS to showcase

- how uncertainty measures of UQ-TSS differ from and are more useful than accuracy;
- how UQ-TSS can be used to guide hyperparameter performance;
- what interpretations of CP locations we can infer from UQ-TSS output.

In Section 4.2 and Section 4.3, we incorporate different baseline TSS algorithms including PELT [19], FLOSS [14], ClaSP [12], and BOCPD [1], which are detailed later, into UQ-TSS and conduct experiments on one long TS (often neglected in TSS literature) and 32 standard benchmark TS to demonstrate the above three aspects.

We also test the following: the effectiveness of our TS data augmentation for TSS compared against baselines (Section 4.4); selecting ensemble size (Section 4.5); and computational overhead (Section 4.6).

General settings. We set the data augmentation window length (w) to span 0.5 seconds of observations. Our supporting website [11] provides experiments on the sensitivity of w. Overall, its selection is robust, and we omit the results from this paper for brevity. β in Algorithm 2 is set to $0.15 * M$. We use covering to measure segmentation accuracy [37].

4.1 Datasets

We use a benchmark collection of 32 TS from other recent TSS research [12, 14]. This repository contains a mix of real, synthetic, and semi-real TS. However, much of this repository contains shorter TS with few CPs (1 in most cases). We provide the full dataset details on our supporting website [11].

We also use the long TS, PAMAP2 [29] to evaluate UQ-TSS. PAMAP2 is a 10.7-hour TS of a person performing various physical activities. The dataset consists of 3850505 observations over 45 variables and contains 25 CPs to detect. We showcase PAMAP2 because these types of TS datasets naturally have more opportunities for uncertainty to arise than shorter TS datasets. The higher uncertainty and more CPs allow us to better demonstrate the hyperparameter optimization and CP location distributions of UQ-TSS.

4.2 Uncertainty Quantification: PAMAP2

This experiment demonstrates how UQ-TSS helps guide hyperparameter selection, create different interpretations for CP locations, and how uncertainty metrics differ from traditional accuracy metrics.

4.2.1 Settings. We select Pruned Exact Linear Time (PELT) [19] as our TSS algorithm. PELT is well suited for long multivariate TS like PAMAP2. However, hyperparameter optimization is difficult due to PELT's lack of indicators. PELT uses two hyperparameters: Penalty and Jump. Jump determines how many observations are skipped in the calculation. Jump is used to control the running time but has some influence on CP location. Penalty is a threshold value. An observation above the penalty is considered a CP. When optimizing PELT, the main concern is how to set the penalty.

We test penalty settings from 17000 to 33000 (step size 1000). Bayesian Information Criterion (BIC) [22], often used to help set this hyperparameter, yielded ≈ 20000 for PAMAP2. BIC tends to underestimate. Thus, our selection of penalty values is skewed to include more high-end values. We set the ensemble size $M = 200$. We set the DBSCAN distance to 5 seconds of observations. The range of the penalty candidate set is 4000 with a step size of 1000. For example, when testing penalty=28000, a candidate set of [26000, 27000, 28000, 29000, 30000] is generated. These sets are established based on neighboring values yielding very similar segmentations. They reflect the user's uncertainty when trying to set the penalty. The jump candidate set is $[80, 90, 100, 110, 120]$. For each PELT segmentation, we measure the uncertainty and accuracy without UQ-TSS, as well as accuracy with UQ-TSS.

We ran this experiment 10 times. The variance between results was minuscule, showing UQ-TSS to be very stable. Full details of these results can be found on our supporting website [11].

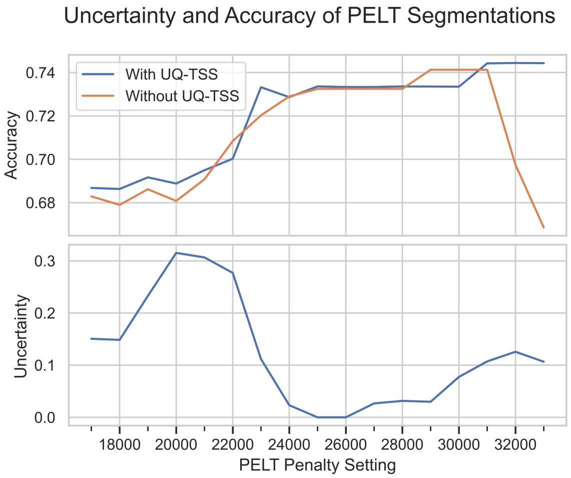

Figure 2: Uncertainty and covering of PELT segmentations over increasing penalty settings.

4.2.2 Interpreting Uncertainty with UQ-TSS. Figure 2 demonstrates how the uncertainty measure of UQ-TSS can guide hyperparameter selection. The top plot shows the accuracy of PELT's segmentations with and without UQ-TSS across the penalty settings 17000 to 33000. The bottom plot shows the uncertainty of the segmentations. The penalty=31000 achieves the highest accuracy. The penalty=26000 achieves the lowest uncertainty.

In a scenario where ground truth is known, we can evaluate the segmentations based on accuracy and uncertainty and tune the penalty accordingly. Without looking at the uncertainty, setting the penalty=31000 is the most reasonable choice since this is where segmentation accuracy is highest. However, this segmentation has much higher uncertainty compared to when penalty=26000. The difference between their accuracies is only 1%. With knowledge of the uncertainty and accuracy, a penalty=26000 is arguably the better choice since it still maintains a high accuracy but has much lower uncertainty.

In a scenario where ground truth is unknown, we can still evaluate segmentation performance based on measures of uncertainty alone. The penalty settings of 25000 and 26000 both yield low uncertainty segmentations. Unbeknownst to the user, either selection would result in high accuracy. This demonstrates that uncertainty can be effective for hyperparameter tuning.

Lastly, Figure 2 also shows that most PELT segmentations with UQ-TSS have slightly better or similar accuracy than those without UQ-TSS.

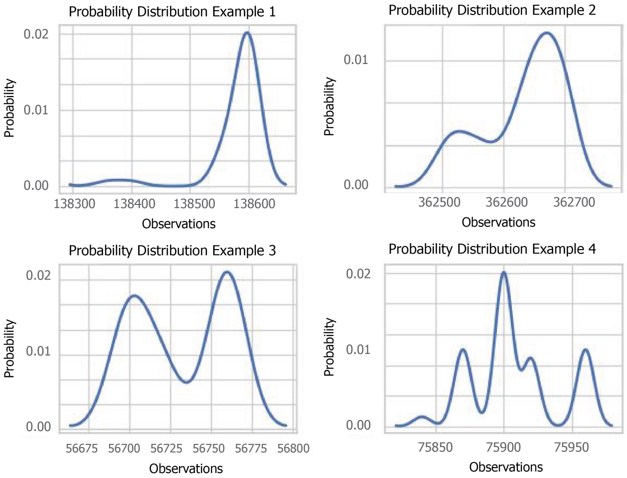

Figure 3: Examples of CP location probability distributions calculated by UQ-TSS.

4.2.3 Changepoint Location Uncertainty. The many CPs of PAMAP2 provide an opportunity to test UQ-TSS's CP location UQ. As mentioned before, we cannot validate due to the lack of ground truth available for such aspects of TSS. However, the main point of this experiment is to demonstrate how to infer the uncertainty of the CP location and highlight the valuable information one can gain from analyzing such information.

These distributions help a user identify spans of uncertainty. A user can interpret observations within the span as belonging to either the activity before or after the change. Excluding these spans of uncertainty can lead to purer examples on which other downstream models can train. We can also create different interpretations for where a change occurred. This information is useful in scenarios where establishing cause and effect is important, e.g., finding the exact trigger for a person's seizure in an EEG recording.

Figure 3 shows four examples of CP location probability distributions found by UQ-TSS with PELT (penalty=28000) arranged in increasing levels of uncertainty. The x-axis represents the index of PAMAP2, where UQ-TSS estimates the probability distribution. The y-axis represents the probability. Example 1 shows a unimodal distribution where nearly all ensemble members detected the CP at observation 138600. Example 2 is a bimodal where most of the ensemble detected the CP at observation 362700, but some also detected it earlier. Example 3 is another bimodal distribution but with an even split between two points in time. Example 4 is a multimodal distribution showing much disagreement among the ensemble and, therefore, high uncertainty.

4.3 Testing Wider Application of UQ-TSS

The purpose of this experiment is to test UQ-TSS on a variety of TS datasets and methods. We use a repository of 32 TS datasets and 3 TSS algorithms. We use an ensemble size of 200 for each TSS algorithm. We describe the algorithms and relevant hyperparameters below. We select these TSS algorithms because they are SOTA, or benchmark studies have shown they are very competitive [37].

FLOSS. FLOSS [14] uses the matrix profile data structure. This data structure relies on a subsequence length to calculate a score over the TS to infer CP locations. We create a candidate set with five values based on 0.8 to 1.2 times the optimal subsequence length provided by the authors [14]. The candidate set for *prominence* is [0.25, 0.30, 0.35].

ClaSP. ClaSP [12] classifies subsequences using self-supervised learning. ClaSP automates any hyperparameter selections and is parameter-free.

BOCPD. BOCPD [1] uses Bayesian statistics and a user-defined probability distribution to calculate the likelihood a change has occurred. We use a student-t distribution with three hyperparameters for variance, skewness, and kurtosis. The candidate set for all of them is [0.1,0.3,0.5]. BOCPD uses a hazard function with one hyperparameter to describe the time between CPs. We create a candidate set of five values based on 0.8 to 1.2 times one-fifth of the TS length. Lastly, The candidate set for *prominence* is [0.25, 0.30, 0.35].

General settings. We base the candidate sets for hyperparameters of the TSS algorithms and inference on the guidelines discussed in Section 3.1.2 and Section 3.1.3, respectively. We use the *find_peaks* algorithm from the SciPy library to detect peaks in scores over time.

	Methods		
	FLOSS	ClaSP	BOCPD
Better/Same/Worse	18 / 8 / 6	25 / 3 / 4	18 / 2 / 12

Table 1: TSS method performance changes across 32 datasets when used with UQ-TSS versus without UQ-TSS.

Table 1 shows how many dataset segmentations had better, same, or worse accuracy for each TSS method using UQ-TSS compared to without using UQ-TSS. We consider the accuracy to be the same if the difference is < 3%. We found differences of < 3% to have insignificant changes in CP location.

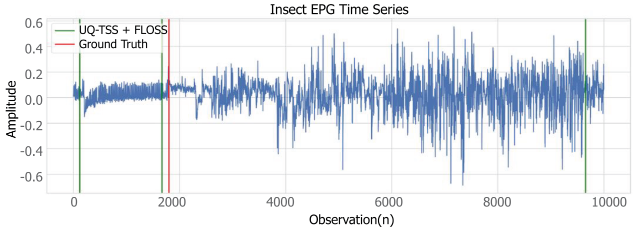

Figure 4: TS Dataset with high segmentation uncertainty among all three TSS methods.

Table 1 shows all three algorithms have an increase in performance across most of the 32 datasets when used with UQ-TSS. However, some datasets show worse performance. Most notably, BOCPD with UQ-TSS has poorer performance across 12 datasets. This is not necessarily a fault with UQ-TSS but an indicator that these specific datasets have much uncertainty associated with them. Alternatively, the TSS algorithm is simply a poor choice for segmenting the dataset. Datasets with worse performance among the three TSS algorithms strongly indicate a high uncertainty associated with them. Inspecting these datasets should reveal ambiguous CPs that are unlabeled in the given ground truth. We can use these datasets to verify UQ-TSS by showing that UQ-TSS detects these uncertain CPs.

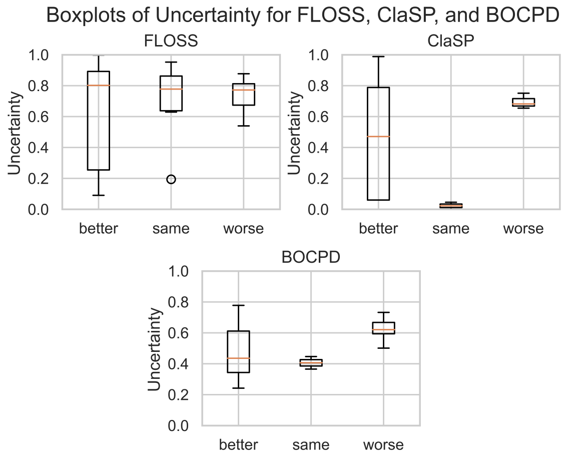

Figure 5: Uncertainty boxplots of better, same, and worse performance categories for the TSS methods FLOSS, ClaSP, and BOCPD.

Figure 4 shows an example of a TS with worse performance across all three TSS methods. The TS is the insect electrical penetration graph (EPG) dataset, which is a recording of an insect's movements. The y-axis is the amplitude, and the x-axis is the observations. This TS is labeled with a single CP indicated by the red vertical line. The green vertical lines indicate the approximate locations where FLOSS with UQ-TSS detected a CP. These areas are clearly ambiguous, displaying a sudden drop in mean at the first green vertical line and a sudden change in variance at the last green vertical line. These detections by UQ-TSS cause worse performance but also show that UQ-TSS successfully captures this uncertainty.

Our primary concern are datasets where the TSS algorithm with UQ-TSS performs worse and the uncertainty is low. To a user, a low uncertainty would falsely indicate that the TSS is well suited for segmenting that TS even though performance is worse. Figure 5 shows boxplots of the uncertainty across all three categories of performance change (better, worse, and same) for all three TSS algorithms. The boxplots show that the uncertainty for worse-performing datasets is high across all three TSS algorithms, with a minimum uncertainty greater than 0.5. This result indicates that these TSS algorithms are poor choices for segmenting those specific datasets.

Augmentation Algorithms	Accuracy
Original	0.967
UQ-TSS TS Augmentation	0.980
Jittering	0.415
Pooling	0.783
Time Warp	0.720
Block Bootstrapping	0.406

Table 2: Effects of augmentation algorithms on TSS accuracy.

4.4 Time Series Augmentation Effectiveness

Table 2 shows segmenting on our augmented TS maintains the accuracy of the original segmentation. Every baseline performs much worse than the original. This demonstrates the unsuitability of these baselines for TSS purposes.

4.5 Selecting Ensemble Size

This section tests the necessary ensemble size for UQ-TSS to accurately measure a segmentation's uncertainty. We use three TSS algorithms: ClaSP, FLOSS, and BOCPD. ClaSP uses no candidate sets, FLOSS uses two candidate sets, and BOCPD uses five candidate sets. We use the "Insect EPG" TS from the 32 dataset repository for this experiment because all three TSS algorithms measured high levels of uncertainty. We expect the number of candidate sets associated with each TSS method to affect the required ensemble size. Each candidate set is a source of uncertainty. TSS methods with more candidate sets likely require a bigger ensemble to accurately capture its uncertainty from all its sources. ClaSP, FLOSS, and BOCPD have candidate sets of 0, 2, and 5, respectively.

Figure 6 shows the uncertainty as ensemble size increases from 10 to 300 of ClaSP, FLOSS, and BOCPD. We see stability around an ensemble size of 75 for ClaSP and FLOSS. However, BOCPD requires an ensemble size of 175. We can conclude that the number of candidate sets does affect the required ensemble size for measuring uncertainty accurately. TSS with more candidate sets requires larger ensembles. The relationship seems to be around 25 members per candidate set, in addition to a minimum size of 75. We also see how BOCPD uncertainty measures are always relatively higher than those of ClaSP and FLOSS. Therefore, even small ensembles are useful for gauging the uncertainty of the segmentation.

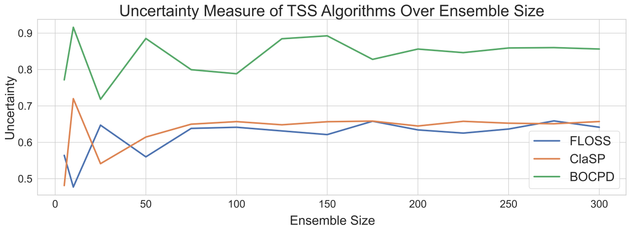

Figure 6: Uncertainty Measures of FLOSS, ClaSP, and BOCPD as Ensemble Size Increases.

Our results indicate an ensemble size of ≈ 100 members is required even for scenarios with no candidate sets. This may seem large, but most TSS algorithms are designed to be computationally inexpensive and running 100 instances is not an issue even in non-supercomputer settings.

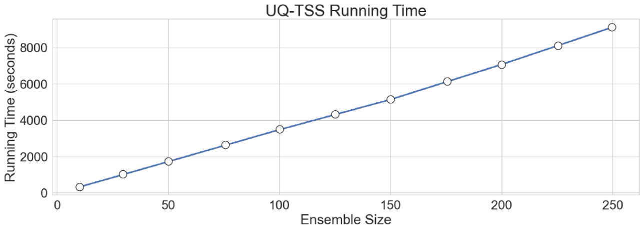

Figure 7: Running time of PELT with UQ-TSS as ensemble size increases.

4.6 Efficiency analysis

Figure 7 shows the running time of PELT with UQ-TSS and a TS with a length of 500000 observations as the ensemble size increases from 10 to 250. We chose PELT because of its capability to analyze a long TS. The linear trend shows that the computations for capturing and measuring uncertainty have negligible computational costs. Therefore, the running time of UQ-TSS mainly depends on the TSS algorithm and the size of the ensemble. This running time can easily be mitigated by parallelizing each ensemble member since each member is independent.

5 Conclusions

We propose UQ-TSS, a generalized framework for measuring segmentation uncertainty from any given TSS algorithm and dataset. We identify the sources of uncertainty in TSS and design mechanisms for UQ-TSS to capture them. This includes a novel TS augmentation algorithm for capturing TSS data uncertainty. UQ-TSS uses ensemble learning to combine multiple CP predictions and use them as samples to estimate probability distributions. UQ-TSS characterizes these distributions to determine CP predictions, the uncertainty of CP presence, the uncertainty of CP locations, and the overall uncertainty of the segmentation. UQ-TSS provides several benefits for TSS, including new evaluation measures, segmentation refinement, hyperparameter guidance, and the creation of different interpretations of CP locations.

Acknowledgments

This work is supported by NSF #1757207, #1914635, and GAANN DoEd grant number P200A180005.

References

[1] Ryan Prescott Adams and David JC MacKay. 2007. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742* (2007).

[2] Sander PA Alewijnse, Kevin Buchin, Maike Buchin, Andrea Kölzsch, Helmut Kruckenberg, and Michel A Westenberg. 2014. A framework for trajectory segmentation by stable criteria. In *Proceedings of the 22nd ACM SIGSPATIAL international conference on advances in geographic information systems*. 351–360.

[3] Christoph Bergmeir, Rob J Hyndman, and José M Benítez. 2016. Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation. *International journal of forecasting* 32, 2 (2016), 303–312.

[4] Ritwik Bhaduri, Subhrajyoty Roy, and Sankar K Pal. 2022. Rough-Fuzzy CPD: a gradual change point detection algorithm. *Journal of Data, Information and Management* 4, 3 (2022), 243–266.

[5] Xin Bi, Chao Zhang, Fangtong Wang, Zhixun Liu, Xiangguo Zhao, Ye Yuan, and Guoren Wang. 2021. An uncertainty-based neural network for explainable trajectory segmentation. *ACM Transactions on Intelligent Systems and Technology (TIST)* 13, 1 (2021), 1–18.

[6] Christian Bors, Christian Eichner, Silvia Miksch, Christian Tominski, Heidrun Schumann, and Theresia Gschwandtner. 2020. Exploring Time Series Segmentations Using Uncertainty and Focus+ Context Techniques.. In *EuroVis (Short Papers)*. 7–11.

[7] Zdravko I Botev, Joseph F Grotowski, and Dirk P Kroese. 2010. Kernel density estimation via diffusion. (2010).

[8] Leo Breiman. 2001. Random forests. *Machine learning* 45 (2001), 5–32.

[9] Bertrand Charpentier, Ransalu Senanayake, Mykel Kochenderfer, and Stephan Günnemann. 2022. Disentangling epistemic and aleatoric uncertainty in reinforcement learning. *arXiv preprint arXiv:2206.01558* (2022).

[10] Michael C Darling. 2019. Using uncertainty to interpret supervised machine learning predictions. (2019).

[11] Erick Draayer. 2024. Supporting website for this paper. https://github.com/ecdraayer/UQ-TSS.

[12] Arik Ermshaus, Patrick Schäfer, and Ulf Leser. 2023. ClaSP: parameter-free time series segmentation. *Data Mining and Knowledge Discovery* 37, 3 (2023), 1262–1300.

[13] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, Vol. 96. 226–231.

[14] Shaghayegh Gharghabi, Yifei Ding, Chin-Chia Michael Yeh, Kaveh Kamgar, Liudmila Ulanova, and Eamonn Keogh. 2017. Matrix profile VIII: domain agnostic online semantic segmentation at superhuman performance levels. In *2017 IEEE international conference on data mining (ICDM)*. IEEE, 117–126.

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

[16] Yann Guédon. 2015. Segmentation uncertainty in multiple change-point models. *Statistics and Computing* 25, 2 (2015), 303–320.

[17] Guillermo Iglesias, Edgar Talavera, Ángel González-Prieto, Alberto Mozo, and Sandra Gómez-Canaval. 2023. Data augmentation techniques in time series domain: a survey and taxonomy. *Neural Computing and Applications* 35, 14 (2023), 10123–10145.

[18] Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems* 30 (2017).

[19] Rebecca Killick, Paul Fearnhead, and Idris Eckley. 2012. Optimal detection of changepoints with a linear computational cost. *J. Amer. Statist. Assoc.* 107, 500 (2012), 1590–1598.

[20] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems* 30 (2017).

[21] Christos Markos, JQ James, and Richard Yi Da Xu. 2021. Capturing uncertainty in unsupervised GPS trajectory segmentation using Bayesian deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 390–398.

[22] Andrew A Neath and Joseph E Cavanaugh. 2012. The Bayesian information criterion: background, derivation, and applications. *Wiley Interdisciplinary Reviews: Computational Statistics* 4, 2 (2012), 199–203.

[23] Venkat Nemani, Luca Biggio, Xun Huan, Zhen Hu, Olga Fink, Anh Tran, Yan Wang, Xiaoge Zhang, and Chao Hu. 2023. Uncertainty quantification in machine learning for engineering design and health prognostics: A tutorial. *Mechanical Systems and Signal Processing* 205 (2023), 110796.

[24] Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J Han, Shinji Watanabe, and Shrikanth Narayanan. 2022. A review of speaker diarization: Recent advances with deep learning. *Computer Speech & Language* 72 (2022), 101317.

[25] Valerie J Pasquarella, Paulo Arévalo, Kelsee H Bratley, Eric L Bullock, Noel Gorelick, Zhiqiang Yang, and Robert E Kennedy. 2022. Demystifying LandTrendr and CCDC temporal segmentation. *International journal of applied earth observation and geoinformation* 110 (2022), 102806.

[26] Matt Peterson, Charlie Vollmer, Ronald Brogan, David J Stracuzzi, and Chistopher J Young. 2021. Generating Uncertainty Distributions for Seismic Signal Onset Times. *Bulletin of the Seismological Society of America* 111, 1 (2021), 11–20.

[27] Apostolos F Psaros, Xuhui Meng, Zongren Zou, Ling Guo, and George Em Karniadakis. 2023. Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *J. Comput. Phys.* 477 (2023), 111902.

[28] CLEVELAND RB. 1990. STL: A seasonal-trend decomposition procedure based on loess. *J Off Stat* 6 (1990), 3–73.

[29] Andreas Reiss and Didier Stricker. 2012. Introducing a New Benchmarked Dataset for Activity Monitoring. In *Proceedings of the 16th IEEE International Symposium on Wearable Computers (ISWC)*.

[30] Richard A Roberts and Clifford T Mullis. 1987. *Digital signal processing*. Addison-Wesley Longman Publishing Co., Inc.

[31] Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.

[32] Joram Soch et al. 2024. StatProofBook/StatProofBook.github.io: The Book of Statistical Proofs. https://doi.org/10.5281/zenodo.4305949.

[33] David John Stracuzzi, Maximillian Gene Chen, Michael Christopher Darling, Matthew Gregor Peterson, and Charlie Vollmer. 2017. *Uncertainty quantification for machine learning*. Technical Report. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).

[34] Amal Tawakuli, Bastian Havers, Vincenzo Gulisano, Daniel Kaiser, and Thomas Engel. 2024. Survey: Time-series data preprocessing: A survey and an empirical analysis. *Journal of Engineering Research* (2024).

[35] George R Terrell and David W Scott. 1992. Variable kernel density estimation. *The Annals of Statistics* (1992), 1236–1265.

[36] Charles Truong, Laurent Oudre, and Nicolas Vayatis. 2020. Selective review of offline change point detection methods. *Signal Processing* 167 (2020), 107299.

[37] Gerrit JJ Van den Burg and Christopher KI Williams. 2020. An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222* (2020).

[38] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. 2020. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478* (2020).

[39] Lisa Wimmer, Yusuf Sale, Paul Hofman, Bernd Bischl, and Eyke Hüllermeier. 2023. Quantifying aleatoric and epistemic uncertainty in machine learning: Are conditional entropy and mutual information appropriate measures?. In *Uncertainty in Artificial Intelligence*. PMLR, 2282–2292.

[40] Xixin Wu, Katherine Knill, Mark Gales, and Andrey Malinin. 2020. Ensemble approaches for uncertainty in spoken language assessment. ISCA.