Ensembling in Time Series Segmentation

Tobias Arndt
Supervised by: Prof. Dr. Ulf Leser, Arik Ermshaus

Humboldt-Universität zu Berlin

April 20, 2024

Contents

1 Introduction & Motivation

With the rise of cheap and readily available sensors in every persons lives comes a huge amount of sensory data that is collected from those sensors. This sensory data can be analysed in many different ways. Time series (TS) are sequences of datapoints that are ordered by time. These can for instance be extracted from stock price development, electrocardiograms and motion detection systems [6].

Time Series Segmentation (TSS) is the task of dividing a time series into semantically different regions. This task is closely related to Change Point Detection (CPD) where the goal is to find points within a time series that bound semantically different regions whereas Time Series Segmentation focuses on finding the intervals of those semantic regions [4, 13]. These points are called change points because they signify a change in the underlying system, e.g. the heartbeat of a patient that is changing from normal to abnormal behavior [13]. TSS in real-world-application is used as a preprocessing tool to partition the amount of data from the growing amount of available sensors [7]. This makes it easier for domain experts to analyse, evaluate and annotate data for tasks like Time Series Classification and Anomaly Detection. A wide range of TSS and CPD algorithms have been proposed, e.g. the state-of-the-art algorithms ClaSP [13] and FLUSS [7]. Another recent paper [10] proposes an Ensembling framework to use for the TSS-task.

Ensembling is a strategy that is widely used in many areas of machine learning. The core idea is to combine different methods to make the ensemble more reliable than each of the methods on its own [12]. Ensemble learning can enhance the base methods by reducing the effect bias and variance have on each method. This fundamental strategy has been shown to have big success in other fields of time series analysis. With the rise of more and better state-of-the-arts methods an ensembling approach will likely lead to better results in TSS aswell due to the increase in performance of the base methods used to form the ensemble.

The motivation for this bachelors thesis is to investigate how an ensembling approach can improve the current state-of-the-arts methods in time series segmentation. We will try to achieve this by exploring the current state-of-the-arts methods, implementing the framework for ensembling TSS-algorithms, evaluating the combinations of possible ensembling strategies on test datasets and comparing the findings against state-of-the-art competitors.

2 Background & Related Work

In this section we will formally describe the concepts related to TSS, as well as introduce some related works.

Definition 1. A time series T is a sequence of $n \in \mathbb{N}$ real values, $T = (t_1, \ldots, t_n), t_i \in \mathbb{R}$. The values are also called datapoints.

Definition 2. Given a TS T, a subsequence $T_{s,e}$ of T with start offset s and end offset e consists of the contiguous values of T from position s to position e,

$T_{s,e} = (t_s, \ldots, t_e)$ with $1 \leq s \leq e \leq n$. The length of $T_{s,e}$ is $T_{s,e} = e - s + 1$.

Definition 3. A segmentation of a TS T into $C + 1$ segments is an ordered sequence of change points (or splits) $t_{i_1}, \ldots, t_{i_C}$ with $1 < i_1 < \cdots < i_C < n$.

Definition 4. The problem of time series segmentation (TSS) is to find a meaningful segmentation of a given TS T under the assumption that T was generated by a process with discrete states. A segmentation is considered meaningful when the change points between two subsequent segments correspond to state changes in the underlying process.

A recent paper [10] in the field of time series segmentation introduced a framework for using the ensembling approach to increase the reliability of TSS algorithms. Traditional TSS algorithms use a framework of applying a search function to determine potential split points and a cost function to evaluate the homogeneity of the predicted segmentations. The approach in [10] introduces the use of multiple different cost functions, the outputs of which are then normalized by a scaling function. The scaled outputs are combined by an aggregation function to then be used by the search methods to predict the best change points. Since we use an Ensembling approach, we will introduce the state-of-the-art methods to be used.

2.1 ClaSP

ClaSP [13] is an unsupervised state-of-the-art TSS-algorithm. It creates a Classification Score Profile using concepts of self-supervision to annotate a given time series. The algorithm splits a time series into subsequences of length w and iterates over all possible splitpoints. For each possible splitpoint the subsequences to the left and right are annotated as 0 and 1 respectively and a binary classifier is trained and validated using Cross Validation. A high score obtained from Cross Validation is interpreted as low similarity between the left and the right subsequences and recorded in the Classification Score Profile. Thus the local maxima of the Classification Score Profile represent potential change points.

2.2 FLUSS

FLUSS [7] is another state-of-the-art TSS-algorithm. It creates a so called Arc Curve (AC) which annotates the raw time series about the likelihood of a state change at each point. It outputs an AC vector where the index i contains the number of "arcs" that cross over the respective point T_i of the time series. An arc is a pair (i, j) where j is the starting location of the subsequence that is the 1-NN of the subsequence starting at location i. This can be represented visually as an arrow pointing from index i of the time series to index j. The central idea is that if an underlying state change of the producing system occured that subsequences of the time series that are after the state change will have mostly 1-NN that are also after the state change. Thus if only a small number of arcs cross an index of the time series a change point is predicted to be there.

2.3 BinSeg

Binary segmentation [9] (BinSeg) is a low complexity algorithm as it aims to find the change point that reduces the sum of the cost functions the most. The time series is consequently split into two subsections at that point and the algorithm is then repeated on each subsection until a stopping criterion is met. The low complexity of the algorithm comes with the drawback that the solution it finds is only an approximation of the optimal solution because it employs a greedy strategy in each step. A variation of BinSeg is Wild Binary Segmentation [3, 9] where a single change point detection is employed on multiple intervals where start and endpoints are drawn uniformly. After weighting the found points only the change points are considered which minimizes the sum of costs.

3 Method of Implementation

As our goal is to use an ensembling approach we will use the ensembling framework provided in [10]. We will implement, evaluate and compare the impact of multiple different parts of the ensembling framework, namely the cost fuctions, the scaling functions, the aggregation functions and the search methods. Due to simultaneously calculating each of the cost functions seperately, our method results in an $N \times M$-dimensional array of scores where N is the number of cost functions and M is the length of each cost fuctions results. Those scores are then scaled independently across the Nth dimensions and finally aggregated into a single result of with dimension $1 \times M$. We will use the following cost functions:

- Median-Shift through Least Absolute Deviation - $l1$
- Mean-Shift through Least Squared Deviation - $l2$
- Mahalanobis-Type Metric - $Mahalanobis$
- Piecewise Autoregressive Model - ar
- Classification Score Profile - $ClaSP$
- Fast Low-cost Unipotent Semantic Segmentation - $FLUSS$

For the scaling functions we will compare:

- $\bar{S}_n^{(MinMax)} = \dfrac{S_n - min\{S_n\}}{max\{S_n\} - min\{S_n\}}$,
- $\bar{S}_n^{(Znorm)} = \dfrac{S_n - \mu\{S_n\}}{\sigma\{S_n\}}$,
- $\bar{S}_n^{(MinAbs)} = \dfrac{S_n - \{S_n\}}{\min\{S_n\}}$,
- $\bar{S}_n^{(Rank)} = argsort(S_n, \text{"}ascending\text{"})$,

where S_n is a result of the nth cost function, as a 1-dimensional array of M elements. $\mu\{S_n\}$ is the mean of S_n and $\sigma\{S_n\}$ is the standard deviation. The $argsort()$ function transforms the input into a set of ranks of values in ascending order. For the aggregation functions we will compare:

- $\Psi_{Min} = \{min\{\bar{S}_{n,m}\}\}_m$,
- $\Psi_{Sum} = \left\{\sum_n \bar{S}_{n,m}\right\}_m$,
- $\Psi_{WeightedSum} = \left\{\sum_n \lambda_n \bar{S}_{n,m}\right\}_m$,
- $\Psi_{ThreshholdSum} = \left\{\sum_n \bar{S}_{n,m_{\{\bar{S}_{n,m}<\mu\{\bar{S}_{n,m}\}\}}}\right\}_m$,

where μ is the mean of the sample. As for the search methods we will use the following methods:

- Greedy Extraction [7] - $GreExt$
- Binary Segmentation - $BinSeg$

4 Method of Evaluation

The datasets used for this benchmark will be the UTSA [5] and the TSSB [13] dataset. UTSA contains TS that capture biological and synthetic processes with only few unique segments. The TS are groupable into a wide variety of use-cases which inhibit real changes aswell as both semi- and fully-synthetic changes. The datasets in TSSB are semi-synthetically created from real-world datasets from the UCR Archive [6]. The metric by which we compare the performance of the algorithms is the Covering Score metric [11].

This metric focuses on dividing a TS into segments and reporting a measure for the overlap of the predicted vs the ground truth labels.

$$Covering = \frac{1}{\|T\|} \sum_{s \in segs_T} \|s\| \cdot \max_{s' \in segs_{pred}} \frac{\|s \cap s'\|}{\|s \cup s'\|},$$

where $segs_T$ and $segs_{pred}$ are the ground truth and predicted segmentations respectively. This reports the best-scoring weighted overlap between the ground truth and predicted segmentations as a normed value between 0 and 1 with higher being better. We will compare the our algorithm against six state-of-the-art competitors, namely FLUSS [7], ClaSP [13], BinSeg [9], BOCD [1], PELT [2] and ESPRESSO [8]. The results of this comparison will be ranked and then averaged, this will be displayed with the help of a critical difference diagram to compare ranks between all methods. In addition we will conduct an ablation study on each part of the Ensembling framework, a) the cost function, b) the scaling functions, c) the aggregation functions and d) the search functions. Because of the increased complexity of ensembling approaches we will also conduct a runtime comparison and evaluate how scalable the approach is.

References

[1] Ryan Prescott Adams and David J. C. MacKay. *Bayesian Online Changepoint Detection*. 2007. arXiv: 0710.3742 [stat.ML].

[2] P. Fearnhead R. Killick and I. A. Eckley. "Optimal Detection of Changepoints With a Linear Computational Cost". In: *Journal of the American Statistical Association* 107.500 (2012), pp. 1590–1598. DOI: 10.1080/01621459.2012.737745. eprint: https://doi.org/10.1080/01621459.2012.737745. URL: https://doi.org/10.1080/01621459.2012.737745.

[3] Piotr Fryzlewicz. "Wild binary segmentation for multiple change-point detection". In: *The Annals of Statistics* 42.6 (2014), pp. 2243–2281. DOI: 10.1214/14-AOS1245. URL: https://doi.org/10.1214/14-AOS1245.

[4] Samaneh Aminikhanghahi and Diane J. Cook. "A survey of methods for time series change point detection". In: *Knowledge and Information Systems* 51.2 (May 2017), pp. 339–367. ISSN: 0219-3116. DOI: 10.1007/s10115-016-0987-z. URL: https://doi.org/10.1007/s10115-016-0987-z.

[5] Shaghayegh Gharghabi et al. "Matrix Profile VIII: Domain Agnostic Online Semantic Segmentation at Superhuman Performance Levels". In: *2017 IEEE International Conference on Data Mining (ICDM)*. 2017, pp. 117–126. DOI: 10.1109/ICDM.2017.21.

[6] Hoang Anh Dau et al. *The UCR Time Series Classification Archive*. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/. Oct. 2018.

[7] Shaghayegh Gharghabi et al. "Domain agnostic online semantic segmentation for multi-dimensional time series". In: *Data Mining and Knowledge Discovery* 33 (Jan. 2019), pp. 1–35. DOI: 10.1007/s10618-018-0589-3.

[8] Shohreh deldari et al. "ESPRESSO: Entropy and ShaPe awaRe timE-Series SegmentatiOn for Processing Heterogeneous Sensor Data". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4 (Sept. 2020), pp. 1–24. DOI: 10.1145/3411832.

[9] Charles Truong, Laurent Oudre, and Nicolas Vayatis. "Selective review of offline change point detection methods". In: *Signal Processing* 167 (2020), p. 107299. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2019.107299. URL: https://www.sciencedirect.com/science/article/pii/S0165168419303494.

[10] Iurii Katser et al. "Unsupervised Offline Changepoint Detection Ensembles". In: *Applied Sciences* 11.9 (2021). ISSN: 2076-3417. DOI: 10.3390/app11094280. URL: https://www.mdpi.com/2076-3417/11/9/4280.

[11] Gerrit J. J. van den Burg and Christopher K. I. Williams. *An Evaluation of Change Point Detection Algorithms*. 2022. arXiv: 2003.06222 [stat.ML].

[12] Ibomoiye Domor Mienye and Yanxia Sun. "A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects". In: *IEEE Access* 10 (2022), pp. 99129–99149. DOI: 10.1109/ACCESS.2022.3207287.

[13] Arik Ermshaus, Patrick Schäfer, and Ulf Leser. "ClaSP: parameter-free time series segmentation". In: *Data Mining and Knowledge Discovery* 37.3 (May 2023), pp. 1262–1300. ISSN: 1573-756X. DOI: 10.1007/s10618-023-00923-x. URL: https://doi.org/10.1007/s10618-023-00923-x.