Novel Criteria to Measure Performance of Time Series Segmentation Techniques

André Gensler, Bernhard Sick

Intelligent Embedded Systems
University of Kassel
Kassel, Germany
Email: {gensler, bsick}@uni-kassel.de

Abstract. An important task in signal processing and temporal data mining is time series segmentation. In order to perform tasks such as time series classification, anomaly detection in time series, motif detection, or time series forecasting, segmentation is often a pre-requisite. However, there has not been much research on evaluation of time series segmentation techniques. The quality of segmentation techniques is mostly measured indirectly using the least-squares error that an approximation algorithm makes when reconstructing the segments of a time series given by segmentation. In this article, we propose a novel evaluation paradigm, measuring the occurrence of segmentation points directly. The measures we introduce help to determine and compare the quality of segmentation algorithms better, especially in areas such as finding perceptually important points (PIP) and other user-specified points.

1 Introduction and State of the Art

An important task in signal processing and temporal data mining is *time series segmentation*, the division of a time series in a sequence of segments. In order to perform tasks such as time series classification, anomaly detection in time series, motif detection, or time series forecasting, segmentation is often a pre-requisite. Depending on the application, segmentation can have arbitrary goals which can basically be divided in two sub-categories.

Segmentation for time series reconstruction and representation

The first category is segmentation for time series reconstruction and representation purposes. This category often uses algorithms which evaluate the approximation error in some form and often aim at representing a time series by a series of linear approximations. The existing algorithms can be categorized in

 In: T. Seidl, M. Hassani, C. Beecks (Eds.): Proceedings of the LWA 2014 Workshops: KDML, IR, FGWM, Aachen, Germany, 8-10 September 2014, published at http://ceur-ws.org

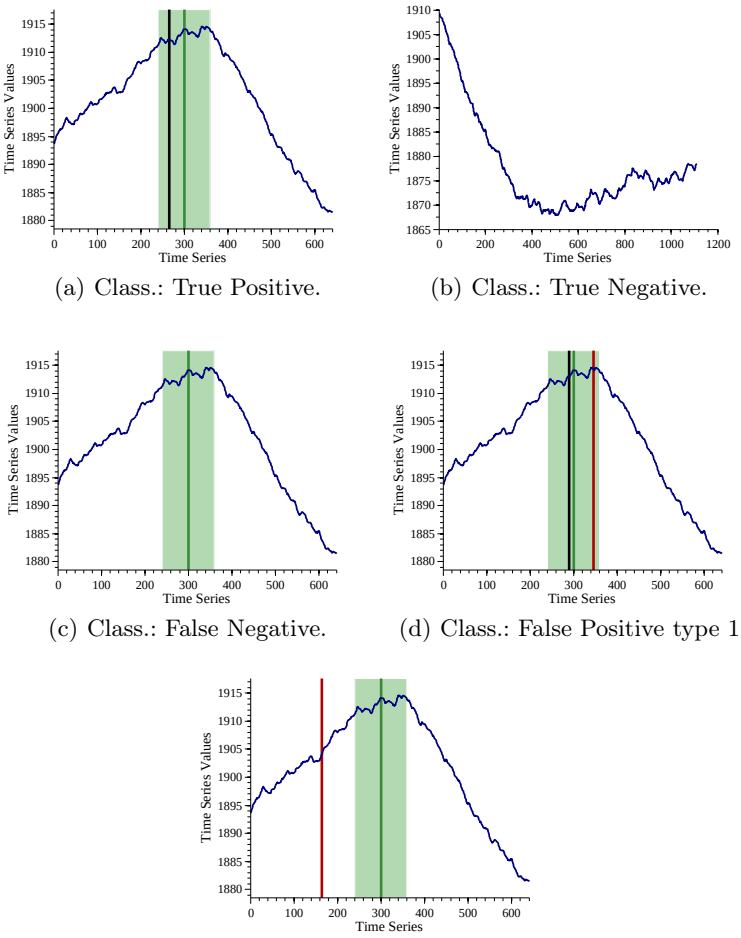
one of the two categories off-line or on-line segmentation [13]. Off-line algorithms have a global view on the data, they therefore know the development of the data points, and, in theory, can achieve better results than on-line techniques. Popular examples for off-line techniques include the Top-Down or the Bottom-Up Segmentation [14], or the k-Segmentation [4], which performs a perfect segmentation of a time series given k segments and an objective function (error function) in very high computing time. On-line techniques only have a local view on the data and have to decide about executing a segmentation without knowing the future development of data points. For many use cases, such as applications with harsh timing constraints, real-time applications, or the processing of large amounts of data, only on-line techniques are applicable. In this realm, the Sliding Window and Bottom-Up (SWAB) algorithm (and variants of it) is widely used [1, 14, 20]. Surveys comparing several time series segmentation techniques can be found in [8, 9, 14, 15]. It can be observed, that most segmentation algorithms evaluate the quality of a segmentation of a time series by the least-squares reconstruction error that an approximation algorithm makes when approximating the segments of a time series [7, 10, 14, 16, 17].

Segmentation at characteristical points of the time series

The second category contains algorithms which aim at performing a segmentation when the characteristics of the time series change in a certain way. This category contains applications, such as segmentation for higher efficiency, indexing long time series, or finding perceptually important points (PIP) [6] and other user-specified points. An algorithm of this category is for example the Sliding Window algorithm, which can deliver reasonably good results [10] very fast using systems of orthogonal polynomials [12]. Various error criteria can be used in this approach to determine the segmentation points, e.g., the approximation error or combinations of polynomial coefficients, such as slope or curvature.

However, there has been little research in the field of evaluation of segmentation in this realm. The frequently used reconstruction error measure is not optimal, as it usually declines with an increasing number of segments (though more segments usually are not necessarily related to a good segmentation) and highly depends on the approximation algorithm and its parameters. Furthermore, it also is an indirect measure, as it only rates the quality of the reconstruction rather than the segmentation points itself. To the best of our knowledge, there does not exist a measure that determines the quality of a segmentation with respect to points the user actually *wants* the algorithm to segment at. Therefore, we introduce a scheme for the evaluation which aims at quantifying the segmentation results with respect to user-defined segmentation points (i.e., labeled points).

The remainder of this article is organized as follows: In Section 2, we introduce a novel interpretation of evaluation of segmentation by treating the segmentation result as a classification problem. We then discuss measures which make sense in order to quantify the segmentation results. Section 3 discusses



(a) Class.: True Positive.

(b) Class.: True Negative.

(c) Class.: False Negative.

(d) Class.: False Positive type 1.

(e) Class.: False Positive type 2.

Fig. 1. Cases in the classification of segmentation points. The green vertical lines represents the segmentation zone (SZ) center while the green areas show the valid SZ area. The black and red lines represent segmentation points determined by a segmentation algorithm. While the black lines represent valid segmentation points (true positives), the red lines are false positives. The goal of a segmentation is to hit an SZ close to the SZ center exactly one time while not producing any segmentation point outside an SZ.

further measures not directly related to the classification interpretation, but nevertheless describe properties of segmentation results. In Section 4, we aim at giving further insight in how these measures work by conducting some experiments and evaluating our measures for the experimental results. Section 5 wraps up our findings.

2 Classification Related Measures

The segmentation of time series can be seen as a classification problem: In each time step, the algorithm has to decide whether to perform a segmentation (S^+) or not (S^-). Consequently, by comparing the performed segmentation to a user-specified target one, a standard metric such as a confusion matrix (see Table 1) can be applied. The algorithm will return a vector of predicted labels $\mathbf{p}$, e.g., $\mathbf{p} = \{S^-, S^-, S^-, S^+, ..., S^-\}$ for each evaluated time series with length N. For the evaluation of a performed time series segmentation, the criteria to determine the elements of the confusion matrix (shown in Table 1) differ from those of a standard classification task.

		Ground Truth		
		Positive	Negative	Total
Prediction	Positive	TP	FP	TP+FP
	Negative	FN	TN	FN+TN
	Total	TP+FN	FP+TN	

Table 1. A standard confusion matrix. Here, we propose an interpretation of the confusion matrix (normally used for standard classification tasks) for segmentation problems.

In a naïve approach, every point in time is given a ground truth label to form a vector $\mathbf{t} \in \{S^+, S^-\}^N$ with only a very small amount of target segmentation points S^+. For the sake of simplicity, our time series here is assumed to consist of equidistant data points in the time domain (though this is not required for the evaluation). By comparing $\mathbf{p}$ and $\mathbf{t}$ pairwisely, the confusion matrix can be formed. This approach, though, does not account for temporal adjacency. For most applications, a segmentation $\mathbf{p}(n) = S^+$ at point in time n of the time series would be considered as a good-enough hit if the target segmentation point was located in the immediate neighborhood $\mathbf{t}(n \pm \epsilon) = S^+$, ($\epsilon \in \mathbb{N}^+$, ϵ is a tolerated deviation). But, in our naive approach, such a result would lead to both a false positive (FP) and false negative (FN) result as they do not match exactly. Therefore, the evaluation metric has to be modified to incorporate temporal neighborhood in a small area around a target segmentation point as a valid segmentation. Additionally, the evaluation result depends not only on a single segmentation decision in time, but on the result in conjunction with the predicted labels in the temporal neighborhood, i.e., while one segmentation at the right location is desirable, multiple segmentation points at the same location have to be penalized.

Depending on the nature of the time series and the desired application, every segmentation task has its own requirements regarding its temporal accuracy of the segmentation, e.g., for some tasks, a too early segmentation may be unproblematic while a late segmentation must not be allowed. For an evaluation it

Algorithm 1 Calc. of Average Segmentation Count (ASC)

```
procedure CalculateASC(Segmentation Points, All SZ)
    Create counter variable v for every SZ
    for each Segmentation Point of Segmentation Points do
        if Segmentation Point is inside SZ then
            Increment variable v of this SZ by 1
        end if
    end for
    return Sum of all v divided by total number of SZ
end procedure
```

therefore makes sense to model each segment to be determined with an earliest and latest point which will still be considered as inside an allowed *segmentation zone (SZ)*. Fig. 1(a) visualizes a sample segmentation zone design, the green vertical line represents the target segmentation point while the green area shows the size of the SZ. The black and red lines represent segmentation points determined by an arbitrary algorithm. In the shown case, one SZ is assigned exactly one determined segmentation point, therefore it is treated as a true positive (TP). Another simple case is shown in Fig. 1(b). If the algorithm does not set a segmentation point in an area where none is expected, the sample is treated as a true negative (TN). If a SZ is not detected, i.e., no segmentation point is associated with it, it is treated as a false negative (FN, type II error), (see Fig. 1(c)). False positives (FP, type I error) can be created in two situations: (1) An SZ is assigned more than one segmentation point, each segmentation point more than one is then treated as a FP, Fig. 1(d). (2) A segmentation point is found in an area where no SZ exists, Fig. 1(e).

Due to the fact that in most cases there will be a lot more elements where there is no segmentation (S^-) than elements where there is a segmentation (S^+), we can assume a heavily imbalanced dataset regarding the class distribution, invalidating basic measures such as accuracy defined by

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}, \tag{1}$$

as it does not consider the distribution of the classes at all. A well-known measure to describe classification performance is the Receiver-Operating-Characteristic (ROC) curve, describing the development of the false-positive rate (FPR)

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{2}$$

and the true-positive rate (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{3}$$

This measure is also valid for imbalanced datasets, though only a small area of the total ROC curve is covered. The FPR remains problematic as it will adopt only small values.

Algorithm 2 Calc. of Absolute Segmentation Distance (ASD)

```
procedure CalculateASD(Segmentation Points, All SZ)
    Initialize variable ASD with 0
    for each Segmentation Point of Segmentation Points do
        if Segmentation Point is inside SZ then
            Add Dist. between Segmentation Point and this SZ center to ASD
        end if
    end for
    return ASD divided by number of found Segmentation Points
end procedure
```

Other prominent measures describing the confusion matrix in one single value beside accuracy are the Area Under (ROC) Curve (AUC) and F_n-score measures such as the F_1 score (both evaluated in [19]) or the Matthews correlation coefficient (MCC). The F_1 score calculated by

$$F_1 = \frac{2\mathrm{TP}}{(2\mathrm{TP} + \mathrm{FP} + \mathrm{FN})} \tag{4}$$

describes the harmonic mean of precision and sensitivity, which in turn means that the amount of TN is not taken into account. The MCC published in [18] calculates a correlation of a two-class classification prediction and is regarded as a balanced measure which can even be used if the class sizes are very different [3]. The MCC takes into account *all* elements of the confusion matrix and is calculated by

$$\mathrm{MCC} = \frac{\mathrm{TP} \cdot \mathrm{TN} - \mathrm{FP} \cdot \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}. \tag{5}$$

It adopts values from -1 to $+1$, where $+1$ represents perfect correlation, 0 a result not better than random guess and -1 absolute disagreement between prediction and ground truth. In contrast to the F_1 score, MCC also incorporates the TN, thus representing the overall structure of the confusion matrix in more detail. In general, it is regarded as a good measure for unbalanced classification problems [3].

To determine the overall quality of a segmentation result, the F_1 score and the MCC seem to be the most promising measures, they have different advantages and disadvantages, though they behave similar in general [2]. The MCC does not just account for (in)correct predictions, but measures correlation, which means that it takes into account systematic mispredictions by adopting values smaller than 0. This can be seen as an advantage for the MCC. Furthermore, it considers all elements of the confusion matrix, consequently representing the classification result in more detail. While this sounds appealing for standard classification tasks, for segmentation the incorporation of the standard case "TN" may turn out as a distracting factor in the evaluation. Depending on the nature of the data, segments are set in different frequencies, resulting in a different proportion

Algorithm 3 Calc. of Average Direction Tendency (ADT)

```
procedure CalculateADT(Segmentation Points, All SZ)
    Initialize variables PreSeg and PostSeg with 0
    for each Segmentation Point of Segmentation Points do
        if Segmentation Point inside SZ then
            Add 1 to PostSeg if after SZ center or
            add 1 to PreSeg if before SZ center
        end if
        return PostSeg / (PostSeg + PreSeg)
    end for
end procedure
```

of positives and negatives, modifying the *MCC*. In addition, factors such as the sampling rate of a time series can have an impact on the *MCC*, resulting in more negatives only (doubling the sampling rate of the same process leads to about twice as many TN). In other words, while the *MCC* considers the whole time series, the F_1 score just accounts for what the segmentation algorithm does (correct and false). Consequently, we think that the F_1 score is more appropriate for the evaluation of most segmentation tasks. Depending on the segmentation task, other forms of the F_n score can make sense (e.g., the F_2 score putting twice as much emphasis on recall). When it comes to adjusting a segmentation algorithm to different applications (with different constraints regarding FN and FP), it also makes sense to use *Precision* and *Recall*

$$\text{Prec.} = \frac{\text{TP}}{\text{TP} + \text{FP},} \tag{6}$$

$$\text{Rec.} = \frac{\text{TP}}{\text{TP} + \text{FN}.} \tag{7}$$

To set the operating point of an algorithm, the goal may not be the overall optimal classification performance (e.g., regarding F_1 score), but achieving the best possible performance when constraining one type of error. These two measures give insights on how an algorithm performs regarding only type I or type II error, respectively, and thereby help to determine the desired operating point.

3 Segmentation Zone Measures

Besides the measurements related to classification, there exist also other properties of segmentation algorithms which are worth examining. An important measure is the Average Segmentation Count (ASC), determining how many times an algorithm triggers a segmentation while being inside a SZ on average. The ASC can be calculated by Algorithm 1. The value of ASC ideally is close to 1, a value lower 1 means too little segments are set inside SZ while a value greater 1 means too many segments are found. It is normed by the SZ count, resulting in an easily understandable result ("per SZ, the algorithm sets ASC segments on average").

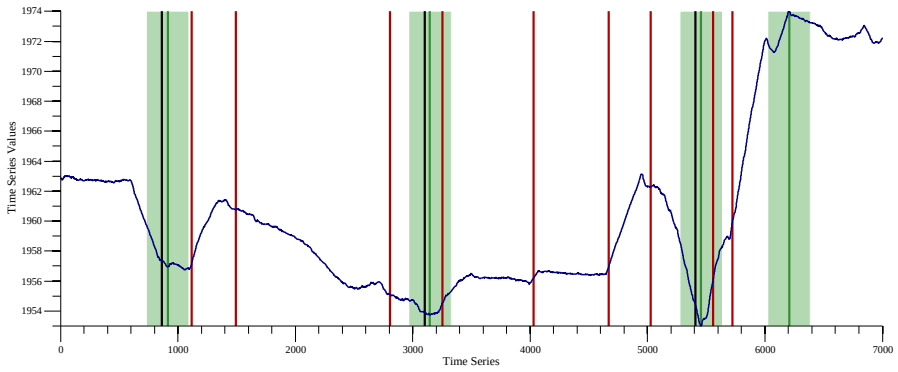


Fig. 2. Segmentation experiment I using excerpt from a real data example [5] (symmetric SZ with total size 180). A basic Sliding Window algorithm using a fast polynomial approximation [11] is utilized (Window Size = 90; Segmentation Criteria: $average < 1970$, $slope < 5$, $curvature > 10^{-4}$, re-segmentation suppression 150 steps). The segmentation points colored in black represent true positives while the red segmentation points represent false positives. The last Segmentation Zone is not hit at all, resulting in a false negative. The confusion matrix that can be calculated by summing up the four respective cases is shown in Table 2.

Furthermore, not only the number of segmentation points is important, but also how accurately they hit the target segmentation. To determine the distance between target segmentation point and found segmentation point, we introduce a measure called Absolute Segmentation Distance (ASD) calculated by Algorithm 2. It is normed by the number of segmentation points found. Finally, it could be of interest whether an algorithm tends to set its segmentation points too early or too late. To specify this characteristic of a segmentation algorithm, we introduce a measure called Average Direction Tendency (ADT) which is described in Algorithm 3. It describes a quotient of early and late segmentation points ("ADT% of segmentation points are too late"). If the algorithm tends to set its segments too early, the value will be below 0.5 while a late segmentation will result in a value greater 0.5. It is useful to use this measure in conjunction with the ASD measure to quantify the amount of segments and the direction of the deviation.

4 Examplary Evaluation

Now we want to briefly show the measures in action to get a better overall impression of how the measures perform. In order to do that, we extracted a time series from a real data set [5] showing activity data with a chest-mounted accelerometer and defined segmentation zones which we expect our segmentation algorithm to find. We used a basic Sliding Window segmentation algorithm evaluating polynomial approximations of the window content using fast update

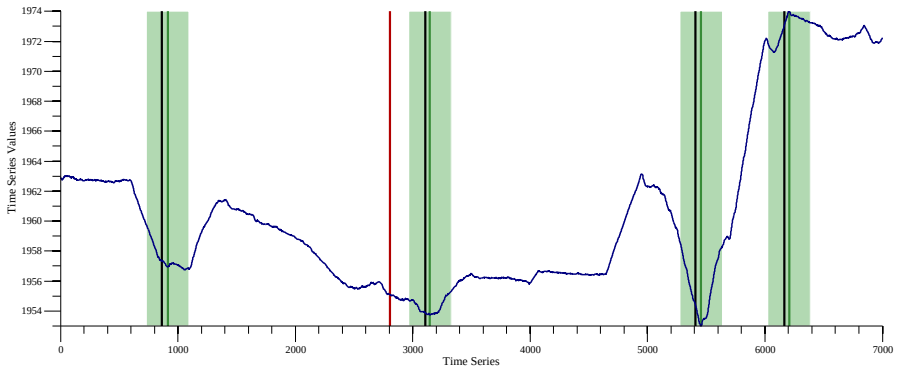


Fig. 3. Segmentation experiment II using excerpt from a real data example [5] (symmetric SZ with total size 180). A basic Sliding Window algorithm using a fast polynomial approximation [11] is utilized with two criteria (Window Size = 90; Segmentation Criteria: (1) $average < 1958$, $slope < 10^{-4}$, $curvature > 10^{-4}$, re-segmentation suppression 300 steps; (2) $average > 1972$, $slope > 10^{-2}$, re-segmentation suppression 200 steps). The segmentation points colored in black represent true positives while the red segmentation points represent false positives. It can be seen easily that this experiment produces less false positives and no SZ is missed. The confusion matrix that can be calculated by summing up the four respective cases is shown in Table 3. It can be expected that in the evaluation an improvement in relation to segmentation experiment I (Fig. 2) can be observed.

formulas as proposed in [10]. The capabilities of the approximation regarding run-time can be found in [12], an implementation of the approximation algorithm can be downloaded at [11]. To show how the measures behave, we performed the segmentation with two parameter combinations, one of which performs significantly better. The result of the time series for the first (worse) parameter combination (Experiment I) is shown in Fig. 2. As we can see from the image, the algorithm tends to set too many segmentation points, some of which are not inside a specified segmentation zone. The points outside the zones are counted as False Positives (FP). Additionally, some segmentation zones are hit multiple times. While the black (valid) segmentation points are counted as True Positives (TP), all further segmentation points are also treated as FP. Furthermore, one of the segmentation zones is not hit at all. This zone is counted as a false negative (FN). From the segmentation results, we can create a confusion matrix as shown in Table 2. Next, we performed the segmentation with a different, apparently better parameter combination (Experiment II, Fig. 3). All zones are hit exactly one time, every segmentation therefore counts as exactly one TP. We can see, that one determined segmentation point lies outside a segmentation zone. Consequently, it is treated as a FP. The confusion matrix for this segmentation is shown in Table 3.

		Ground Truth	
		Positive	Negative
Prediction	Positive	3	9
	Negative	1	6987

Table 2. Example confusion matrix resulting from the segmentation performed in Fig. 2. Standard performance measures can now be applied to the matrix.

		Ground Truth	
		Positive	Negative
Prediction	Positive	4	1
	Negative	0	6995

Table 3. Example confusion matrix resulting from the segmentation performed in Fig. 3. Standard performance measures can now be applied to the matrix.

To the confusion matrix elements, we can now apply the classification related measures described in Section 2. For both confusion matrices 2 and 3, we evaluated the Accuracy (ACC), the F_1 score, and the Matthews Correlation Coefficient (MCC). Additionally, we applied our new segmentation zone measures to the segmentation results, namely the Average Segmentation Count (ASC), the Absolute Segmentation Distance (ASD) and the Average Direction Tendency (ADT). The results are shown in Table 4. In addition we added some baseline results for algorithms performing a segmentation at no point in time (NeverSeg), on every time step (AlwaysSeg) or on random with $p(S^+) = 0.5$. In this table, the classification related measures are shown on the left hand side, while the segmentation zone measures are shown on the right hand side of the table. As we can clearly see, ACC is unable to discriminate between the results of experiment I and II. The *Precision* drastically increases from a value of 0.25 to 0.80. The *Recall* also increases from 0.75 to 1.00, as no FP are produced in Experiment II. The F_1 score has a range between 0 and 1, here the values are 0.375 or 0.888, respectively. We can see a clear difference between result I and II. The MCC behaves numerically similar: In experiment I, the value is 0.433, while experiment II results in a value of 0.894. Both the F_1 score and the MCC behave very similar here. In the realm of the segmentation zone measures, we can see that the algorithm behaves more appropriate with respect to the specified SZ in experiment II. The ideal value of ASC is 1 (one found segment for each SZ). While experiment I returned too many segments, segmentation II yields better results. For the ASD measure, we can see that the segmentation became more accurate regarding the SZ center. It improved from about 70 data samples from the center to only 45 samples on average. Last, the ADT also changed, though this is not a measure for segmentation quality, but more a description of the algorithm characteristics: In experiment I, the quotient between early and late segmentation points was roughly balanced (0.5 would be perfectly balanced) with a value of 0.4, which means a slight overweight for early segmentation points. In experiment II, all segmentation points were too early.

	Acc.	*Prec.*	*Rec.*	F_1	*MCC*	*ASC*	*ASD*	*ADT*
NeverSeg	0.999	1.000	0.000	0.000	0.000	0.0	0.00	-
RandomSeg	0.500	0.001	1.000	0.001	0.000	90.0	90.00	0.5
AlwaysSeg	0.001	0.001	1.000	0.001	0.001	180.0	90.00	0.5
Exp. (I)	0.999	0.250	0.750	0.375	0.433	1.25	70.40	0.4
Exp. (II)	0.999	0.800	1.000	0.888	0.894	1.00	44.75	0.0

Table 4. Segmentation measures extracted from experiments of Fig. 2, Fig. 3, and the respective confusion matrices (Table 2, Table 3). Additionally, some baseline measures were added, showing algorithms performing segmentation at no point in time (NeverSeg), on every time step (AlwaysSeg) or random with $p(S^+) = 0.5$ (RandomSeg). As we can see, the accuracy (ACC) clearly falls short of describing the segmentation result. *Precision* and *Recall* differ significantly, both favoring Experiment II. The F_1 score and the MCC behave similarly, though the MCC takes into account the TN in contrast to F_1. The other measures ASC and ASD improve from Experiment I to II. The segmentation characteristic of the algorithm changes as well, from a balanced segmentation to an early segmentation, as ADT describes.

All in all, these measures help to quantify the quality of a segmentation result. Depending on the application of the segmentation, different measures may be more or less important. Often, a combination of multiple measures helps to specify the characteristics of the segmentation algorithm.

5 Conclusion and Outlook

In this article, we proposed several new evaluation criteria for time series segmentation in the realm of segmentation for the sake of finding specific points such as perceptually important points (PIP), which we categorized in classification related measures and segmentation zone measures. We think our new measures will help to compare the quality of various segmentation approaches better, especially for applications such as motif detection and other applications where the detection of user-defined points turn out to be important. We hope authors will adopt these measures to further increase the comparability of segmentation algorithms. In our future work we aim to evaluate new algorithms for time series segmentation using the introduced measures. We also thought about defining gradual segmentation zones (e. g., by using gaussians) to further specify the quality of a segmentation algorithm.

References

1. Amft, O., Junker, H., Tröster, G.: Detection of eating and drinking arm gestures using inertial body-worn sensors. In: Proceedings of 9th IEEE International Symposium on Wearable Computers. pp. 160–163. IEEE, Osaka, JP (2005)
2. Arora, A.: Matthews Correlation Coefficient - How well does it do? `http://standardwisdom.com/softwarejournal/2011/12/`

matthews-correlation-coefficient-how-well-does-it-do/, last access 07/02/2014
3. Baldi, P., Brunak, S., Chauvin, Y., Andersen, C., Nielsen, H.: Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics 16(5), 412–424 (2000)
4. Bellman, R.: On the approximation of curves by line segments using dynamic programming. Communications of the ACM 4(6), 284 (1961)
5. Casale, P., Pujol, O., Radeva, P.: Activity recognition from single chest-mounted accelerometer data set. https://archive.ics.uci.edu/ml/datasets/Activity+Recognition+from+Single+Chest-Mounted+Accelerometer, last access 06/30/2014
6. Chung, F.L., Fu, T.C., Luk, R., Ng, V.: Flexible time series pattern matching based on perceptually important points. In: Proceedings of 17th International Joint Conference on Artificial Intelligence Workshop on Learning from Temporal and Spatial Data. pp. 1–7 (2001)
7. Chung, F.L., Fu, T.C., Ng, V., Luk, R.: An evolutionary approach to pattern-based time series segmentation. IEEE Transactions on Evolutionary Computation 8(5), 471–489 (2004)
8. Esling, P., Agon, C.: Time-series data mining. ACM Computing Surveys (CSUR) 45(1), 12–48 (2012)
9. Fu, T.: A review on time series data mining. Engineering Applications of Artificial Intelligence 24(1), 164–181 (2011)
10. Fuchs, E., Gruber, T., Nitschke, J., Sick, B.: Online segmentation of time series based on polynomial least-squares approximations. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(12), 2232–2245 (2010)
11. Gensler, A., Gruber, T., Sick, B.: Fast Approximation Library. http://ies-research.de/Software, last access 05/28/2014
12. Gensler, A., Gruber, T., Sick, B.: Blazing fast time series segmentation based on update techniques for polynomial approximations. In: Proceedings of 13th IEEE International Conference on Data Mining Workshops (ICDMW13). pp. 1002–1011. IEEE, Dallas, USA (2013)
13. Keogh, E., Chu, S., Hart, D., Pazzani, M.: An online algorithm for segmenting time series. In: Proceedings of 1st IEEE International Conference on Data Mining (ICDM2001). pp. 289–296. IEEE, San Jose, USA (2001)
14. Keogh, E., Chu, S., Hart, D., Pazzani, M.: Segmenting time series: A survey and novel approach. Data mining in time series databases 57, 1–22 (2004)
15. Keogh, E., Kasetty, S.: On the need for time series data mining benchmarks: A survey and empirical demonstration. Data Mining and Knowledge Discovery 7(4), 349–371 (2003)
16. Lemire, D.: A better alternative to piecewise linear time series segmentation. In: SIAM International Conference on Data Mining. pp. 545–550. SIAM (2007)
17. Liu, X., Lin, Z., Wang, H.: Novel online methods for time series segmentation. IEEE Transactions on Knowledge and Data Engineering 20(12), 1616–1626 (2008)
18. Matthews, B.W.: Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure 405(2), 442–451 (1975)
19. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Information Processing & Management 45(4), 427–437 (2009)
20. Van Laerhoven, K., Berlin, E., Schiele, B.: Enabling efficient time series analysis for wearable activity data. In: Proceedings of 8th International Conference on Machine Learning and Applications (ICMLA'09). pp. 392–397. IEEE, Miami, USA (2009)