Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

CBD: An Efficient Time Series Segmentation Algorithm and its Potential Integration with Machine Learning Approaches

Bhagyashree Ambore, Sunitha K, Aishwarya G, Janardhan Singh

Department of ISE, RNS Institute of Technology, Affiliated to VTU, India.
Email: ambore.bhagyashree@gmail.com

This paper presents a comprehensive analysis of various time series segmentation methods, including traditional statistical techniques and contemporary machine learning approaches. We focus on the performance of the Top-Down segmentation method, particularly enhanced by significant point identification, and compare it with established methods such as Sliding Window, Bottom-Up, K-Means clustering, and Dynamic Time Warping (DTW). Through extensive experiments across diverse datasets, we evaluate each method's performance based on approximation error, computational efficiency, and overall segmentation quality. Our results demonstrate that the proposed enhancement to the Top-Down method, which incorporates significant points for segmentation, achieves competitive accuracy while significantly reducing computational time compared to traditional methods. This study underscores the potential of leveraging advanced segmentation strategies to improve time series analysis in various application domains.

Keywords: component, formatting, style, styling, insert.

1. Introduction

Time series analysis is a critical area of research with diverse applications across various domains, including cybersecurity [1], finance [2], and healthcare [3]. Accurate and efficient analysis of time series data is essential for tasks such as forecasting, anomaly detection, and informed decision-making. A fundamental component of time series analysis is the effective representation of data, which seeks to reduce dimensionality while preserving the essential characteristics of the original series [4]. Among the various representation techniques, Piecewise Linear Representation (PLR) has gained prominence for its simplicity, efficiency, and enhanced interpretability.

A significant challenge in time series representation is the segmentation process, which involves dividing a time series into meaningful subsequences [5]. Despite its importance, segmentation remains an underexplored area, with many studies relying on traditional algorithms like Top-Down, Bottom-Up, and Sliding Window methods. These approaches can

be computationally inefficient and may yield less accurate results in certain scenarios [6]. This gap in the literature underscores the necessity for a comprehensive evaluation of existing methods and the development of more advanced, efficient alternatives.

This paper presents several contributions to the field of time series segmentation. First, it provides a detailed comparison of existing segmentation algorithms, including Sliding Window, Bottom-Up, and Top-Down methods, evaluating their performance across various datasets to highlight their strengths and limitations. Second, we introduce an innovative segmentation approach, the Candidate-based Top-Down (CBD) method. By leveraging statistical insights and integrating machine learning techniques, CBD efficiently identifies optimal split points in time series data. Our results demonstrate that the CBD algorithm achieves approximation errors comparable to those of the traditional Top-Down method while significantly reducing computational time. This positions CBD as a promising alternative for time series segmentation, paving the way for more effective data analysis across numerous applications.

2. Related works

Piecewise Linear Representation (PLR) of time series has emerged as a pivotal technique in various applications, simplifying complex datasets by transforming them into linear segments. This method enhances analysis and interpretation, offering simplicity, efficiency, and improved clarity compared to other representation methodologies [7]. As such, PLR has become the most widely adopted segmentation approach in time series data analysis.

Despite its popularity, several research gaps and limitations in PLR techniques remain unaddressed:

- Noise Sensitivity: Traditional PLR methods often struggle with noisy data, where fluctuations may be misinterpreted as significant changes, resulting in inaccurate segmentations. This issue is particularly pronounced in datasets with high variability, emphasizing the need for methods that preserve the integrity of the time series representation [8].

- Computational Efficiency: Many existing PLR algorithms, such as Bidirectional Piecewise Linear Representation (BPLR), while robust, demand substantial computational resources, which can hinder performance when processing large datasets [9].

- Adaptability to Data Complexity: Most PLR methods are designed for univariate time series. However, real-world applications frequently involve multivariate or non-stationary data, necessitating more sophisticated approaches to effectively capture the underlying dynamics of such datasets.

Recent studies have begun to address these challenges by developing more adaptive and robust PLR methods. For instance, Shi et al. [9] introduced a bidirectional segmentation algorithm that enhances traditional PLR by employing a dual scanning process to minimize error and improve the detection of true change points. Additionally, advancements in adaptive error-bounded piecewise linear approximation have shown promise for achieving accurate and computationally efficient representations, particularly in financial time series analysis [10].

A novel PLR method has also been proposed that utilizes R-squared statistics to identify optimal segmentation points, aiming to maximize the explained variance and ensuring that each linear segment captures the underlying trends effectively. This approach has been validated through extensive simulations, demonstrating improvements in both accuracy and interpretability of segmented time series [11]. Furthermore, Trirat et al. [12] conducted a comprehensive survey of various time series representation methods, including piecewise linear techniques such as piecewise aggregate approximation (PAA) and symbolic-based approaches. Their work examines the strengths and limitations of each method and proposes a framework for universal time series representation learning, aimed at leveraging the best features of diverse techniques to enhance performance across various tasks and datasets.

Additional notable contributions include:

- Machine Learning Approaches: Recent works have integrated machine learning techniques with PLR methods to enhance segmentation performance. For instance, S.Ahmed et al. [13] proposed a hybrid model that combines PLR with support vector machines (SVM) to improve segmentation accuracy in non-linear time series data. Their approach leverages the strengths of both methods, allowing for effective pattern recognition and trend identification.

- Deep Learning for Time Series Segmentation: Studies like those by Casolaro et al.[14] focuses on models like CNNs, RNNs, and TCNs, emphasizing their strengths in pattern recognition and sequence prediction. While CNNs excel at identifying patterns, RNNs are better at capturing dependencies over time. The paper highlights the challenges these models face, such as limitations in handling long-range dependencies, and explores emerging approaches like Temporal Convolutional Networks and Transformer models.

- Dynamic Time Warping (DTW): A method widely used for time series analysis, DTW has been combined with PLR to improve segmentation in datasets with varying lengths and non-linear distortions. The study by Berndt and Clifford [15] demonstrates that integrating DTW with PLR can enhance segmentation quality by aligning similar patterns in time series data.

The ongoing evolution of PLR techniques is likely to focus on several key areas: Enhancing Robustness to Noise: Developing algorithms capable of distinguishing between noise-induced fluctuations and genuine data trends. Scalability and Efficiency: Designing faster algorithms that can manage larger datasets without sacrificing accuracy. Multivariate and Complex Data Handling: Extending PLR methods to effectively represent multivariate and non-stationary time series data.

In conclusion, while PLR remains a cornerstone of time series analysis, there is significant potential for innovation that could broaden its applicability and effectiveness. As data complexity and volume continue to grow, these enhancements will be crucial for addressing the evolving demands of time series analysis.

3. Background

In this section, we will explore three prominent Piecewise Linear Representation (PLR) methods: Sliding Window, Bottom-Up, and Top-Down segmentation. Each method offers

unique advantages and is applicable to different scenarios in time series analysis.

A. Sliding Window

The Sliding Window method starts with a fixed initial point and incrementally expands the window until the error between the actual data and the approximated linear segment exceeds a predefined threshold. Once this threshold is surpassed, a segment boundary is established, and the process restarts from the next point after the endpoint of the last segment.

This approach guarantees that each segment adheres to the maximum allowable error, striking a balance between accuracy and simplicity. It is particularly advantageous for real-time and online processing applications due to its localized operation, allowing for efficient updates as new data points arrive. Moreover, it effectively captures short-term trends and patterns within the data, making it suitable for applications in signal processing, data compression, and anomaly detection. Incorporating machine learning techniques into the Sliding Window method can enhance its performance by automatically adjusting the window size and error threshold based on the characteristics of the incoming data, improving adaptability across various datasets.

B. Bottom-Up

Bottom-Up segmentation begins by representing the data as a series of small initial segments, typically formed by two consecutive data points. These segments are iteratively merged based on a criterion that aims to minimize the increase in error. The merging process continues until the error of the resulting segments exceeds a predefined threshold, ensuring that the final segments provide an optimal balance between accuracy and simplicity.

This method excels at reducing noise and capturing local variations within the data, making it particularly effective in applications requiring detailed pattern recognition, such as in data compression and signal processing. Integrating machine learning algorithms in Bottom-Up segmentation can facilitate dynamic merging criteria, enabling the algorithm to learn optimal thresholds from historical data and adaptively refine the segmentation process.

C. Top-Down

The Top-Down segmentation method begins by modeling the entire data sequence as a single linear segment and computing the associated approximation error. It identifies the point at which the errors of the left and right segments are maximized, subsequently dividing the data at this juncture into two distinct segments.

This recursive process continues, further dividing each segment at points of maximum error until the error within each segment falls below a predefined threshold. The Top-Down approach is particularly adept at capturing global patterns and trends within the data, making it valuable for applications requiring high-level data abstraction, such as trend analysis, pattern recognition, and data summarization. To enhance the Top-Down method, machine learning approaches can be employed to optimize the selection of segment boundaries. By training models on historical data, algorithms can learn to predict optimal split points, thereby improving accuracy and reducing computational overhead.

4. Experimental Comparison of Segmentation Methods

To evaluate the performance of the proposed segmentation method, topdown_segment_with_significant_points, we conducted a series of experiments comparing its efficiency and accuracy against both traditional segmentation algorithms and contemporary machine learning approaches.

Experiment Setup

In our studies, we employ linear regression to model time series segments rather than relying on simple interpolation. This approach is preferred because linear regression is more resilient to noisy data and capable of extending predictions beyond the available data points, making it ideal for forecasting and trend analysis. Unlike interpolation, which is confined to the dataset's scope, linear regression enables statistical inference, providing insights into the strength and relevance of relationships among variables.

Selecting the maximum error is a critical decision for segmentation algorithms, including the Candidate-based Top-Down (CBD) method. Currently, to our knowledge, there are no automated methods for determining this parameter. As the maximum error approaches zero, all algorithms perform similarly, producing error-free segments. Conversely, as the maximum error increases significantly, the algorithms converge in performance, simplifying the entire time series into a single optimal line. This selection process involves balancing data compression and precision. The "reasonable value" for maximum error is subjective and varies according to the specific requirements of the data mining application and the characteristics of the data itself. In our approach, smaller maximum error values tend to lead to overly detailed and fragmented approximations, while larger values result in overly generalized approximations.

The evaluation of the algorithms is conducted by assessing their performance on two distinct datasets, each utilizing a tailored range of maximum errors determined by the standard deviation (std) of the respective time series. For the first dataset, the tailored ranges are

$$[\frac{std}{10}, \frac{std}{5}, \frac{std}{2}, std, 2 \times std, 5 \times std, 10 \times std]$$

$$[\frac{std}{100}, \frac{std}{50}, \frac{std}{2}, std, 2 \times std, 50 \times std, 100 \times std]$$

While for the second dataset, the ranges are

These specific intervals were established through extensive experimentation to explore the impact of varying error thresholds on the performance of the CBD algorithm, thus enabling a comprehensive comparison under diverse conditions characterized by both datasets.

The experimental environment for the study was configured on a computer system equipped with a 64-bit Windows 10 operating system. The hardware included an Intel Core i5-6500 CPU and 16GB of RAM.

Methodology

Algorithm Implementation: The combined algorithm was implemented to identify significant points based on the rolling mean and standard deviation. It then recursively segmented the time

series, ensuring that the maximum allowable error for each segment was not exceeded.

(i) Performance Metrics:

(a) Approximation Error: Calculated using the sum of absolute differences between the original data points and their corresponding approximated values:

$$Error = \sum_{i=a}^{b} |x_i - Approximation(i)| \quad (1)$$

(b) Execution Time: The total time taken to process each dataset was recorded to evaluate computational efficiency.

(ii) Comparison Criteria: The performance of the combined algorithm was compared with:

(a) Traditional Algorithms: (i) Top-Down Method: A conventional approach that recursively segments the data without significant deviation detection. (ii) Sliding Window Method: A fixed-window technique that approximates the data based on local averages without identifying significant points. (iii) Bottom-Up Method: This method starts with individual data points and merges them based on a specified error threshold.

(b) Machine Learning Approaches: (i) K-Means Clustering: Used to identify natural clusters within the time series data. Data points were grouped based on similarity, and segment boundaries were determined based on cluster assignments. (ii) Dynamic Time Warping (DTW): A technique that aligns sequences that may vary in speed or timing, allowing for segmentation based on distance measures between time series.

An experimental evaluation of Prominent Segmentation methods

As previously discussed, from an application perspective, Sliding Window Segmentation is particularly effective for real-time processing and capturing local patterns. The Candidate-based Top-Down (CBD) segmentation excels in identifying global patterns and trends, making it advantageous for high-level data abstraction while maintaining efficiency. Bottom-Up Segmentation is most effective for noise reduction and preserving local details. Each method possesses distinct strengths and weaknesses, rendering them suitable for different applications based on the specific requirements of the data analysis task.

In this section, we present a detailed evaluation of these methods through experimental comparisons designed to assess their performance under various conditions. These experiments aim to provide insights into the practical effectiveness of each method, emphasizing their relative strengths and weaknesses across different scenarios.

(A) Datasets: The experiments utilized two distinct datasets to ensure comprehensive coverage of the various characteristics inherent in time series data. The first dataset, introduced by Xu et al. [16], comprises a seasonal time series of Key Performance Indicators (KPIs) in web applications, known as the SKAW dataset, containing a total of 142,540 data points. To reduce complexity and facilitate comparative analysis, this dataset was partitioned into 10 separate batches, each considered an independent time series consisting of 14,254 data points.

The second dataset used in the experiments is the L1A radar telemetry data from NASA's Soil Moisture Active Passive (SMAP) mission [17,18]. This dataset has been anonymized

concerning time, and all telemetry values have been normalized to a range of (-1, 1) based on the observed minimum and maximum values. Unlike the first dataset, which primarily features seasonal time series, the SMAP dataset includes a diverse array of time series, encompassing both stationary and non-stationary series, with or without trends. This diversity is crucial for a thorough evaluation of time series characteristics and the robustness of the analysis methods employed. After filtering out time series with constant values, the dataset comprises 45 time series, with each time series containing approximately 2,700 data points.

(B) Metrics: The algorithms are evaluated based on two key metrics: the total sum of absolute approximation errors and computational time. The first metric assesses the accuracy of a segmentation algorithm by summing the absolute differences between actual data points and their approximations on the fitted linear segments. A lower total sum indicates higher accuracy, reflecting the algorithm's ability to preserve the original time series patterns without significant distortion.

Computational time measures the efficiency of the algorithm, which is crucial for real-time processing or handling large datasets. It tracks the duration of the segmentation process. An effective algorithm, such as the CBD method, optimizes computational time without compromising accuracy, which is essential for practical applications where speed is a priority.

Proposed Algorithm

As highlighted in the previous section, the Top-Down approach generally outperforms other segmentation methods in terms of accuracy. However, it is significantly less efficient concerning computational time. Therefore, efforts have been made to enhance the speed of the Top-Down algorithm while preserving its accuracy.

A. CBD - A Rapid and Precise Segmentation Algorithm

The slow performance of the traditional Top-Down algorithm is primarily due to its brute-force methodology, which searches through all possible points in the time series to identify the optimal index for segment splitting based on minimal approximation error. This exhaustive search process is recursively repeated, leading to considerable computational overhead. To address this limitation, we propose the Candidate-based Top-Down (CBD) method, which identifies potential optimal indices for segmentation much more efficiently. This significantly reduces computational time while maintaining high accuracy.

The CBD algorithm introduces an innovative approach to time series segmentation by leveraging statistical insights. Instead of relying on a conventional brute-force search, the algorithm computes the rolling mean and standard deviation over a predefined window size across the time series. When the absolute difference between the actual value at any point and the rolling mean exceeds a specified multiplier of the standard deviation, that point is flagged as a potential optimal index for segmentation.

This targeted identification of split points not only accelerates the segmentation process but also preserves the accuracy associated with the original Top-Down approach. By refining the candidate selection process, the CBD algorithm effectively balances computational efficiency with segmentation precision, making it a robust alternative for time series analysis.

In this context, the Candidate-based Top-Down (CBD) algorithm plays a pivotal role by

systematically identifying candidate split points based on significant deviations from the local mean. These pre-identified points are then leveraged to perform segmentation more efficiently, optimizing the computational demands of the traditional Top-Down approach. This integrated strategy marks a substantial advancement in the field, offering a faster and more effective solution for time series analysis.

Algorithm Steps

1. **Input and Parameters:**
   - Time series data: *data*
   - Parameters:
     - *window_size*: Size of the rolling window.
     - *threshold_multiplier*: Multiplier for detecting deviations.
     - *max_error*: Maximum allowed error for segment approximation.
2. **Calculate Rolling Statistics:**
   - For each point i, compute:

$$\text{Rolling Mean} = \frac{1}{\text{window\_size}} \sum_{j=i-\text{window\_size}+1}^{i} data[j]$$

$$\text{Rolling Std} = \sqrt{\frac{1}{\text{window\_size}} \sum_{j=i-\text{window\_size}+1}^{i} (data[j] - \text{Rolling Mean})^2}$$

3. **Detect Significant Points:**
   - For each point i:
     - Compute deviation from the mean: $\text{diff}_i = |data[i] - \text{Rolling Mean}|$
     - If $\text{diff}_i > \text{threshold\_multiplier} \times \text{Rolling Std}$, mark i as significant.
4. **Segment Approximation:**
   - For each segment $(start, end)$, compute:

$$\text{approximation} = \frac{1}{\text{end} - \text{start} + 1} \sum_{i=\text{start}}^{\text{end}} data[i]$$

5. **Error Calculation:**
   - For each segment $(start, end)$, calculate:

$$\text{error} = \sum_{i=\text{start}}^{\text{end}} |data[i] - \text{approximation}|$$

6. **Recursive Segmentation:**
   - Identify significant points within the range and split at the point that minimizes error.
   - Recurse if any segment error exceeds *max_error*.
7. **Stopping Criterion:**
   - Stop when each segment's error is less than or equal to *max_error*.
8. **Output:**
   - Return the list of detected segments.
   - Visualize the segments on the time series.

Methodology

Top-Down Segmentation with Significant Points: This method involves identifying significant deviations from local means, using rolling statistics to determine potential segment boundaries.

Formulations:

- Rolling Mean:

$$mean = \frac{1}{n} \sum_{j=i-n+1}^{i} x_j \quad (2)$$

- Rolling Standard Deviation:

$$std_i = \sqrt{\frac{1}{n} \sum_{j=i-n+1}^{i} (x_j - mean_i)^2}$$

(3)

Deviation Condition: if $diff_i$ > threshold multiplier x std_i

Traditional Top-Down Segmentation: This method recursively divides the time series into segments based on a maximum error threshold.

Formulation:

- Approximation Error:

$$error = \sum_{j=start}^{end} |x_j - mean| \quad (4)$$

Sliding Window Segmentation: This method divides the time series into fixed-size windows, calculating the approximation error for each.

$$error = \sum_{j=i}^{min(i+windowsize, len(data))} |x_j - mean_{window}|$$

(5)

Bottom-Up Segmentation: This brute-force method evaluates every possible segment and retains those with acceptable errors.

$$error = \sum_{k=i}^{j} |x_k - mean_{i,j}|$$

(6)

K-Means Clustering: An unsupervised method grouping time series data into clusters based on similarity.

$$J = \sum_{i=1}^{k} \sum_{x \varepsilon C_i} ||x - \mu_i||^2 \quad (7)$$

Dynamic Time Warping (DTW): DTW measures similarity between temporal sequences that may vary in speed.

$$D(i,j) = d(x_i, y_j) + min \begin{cases} D(i-1,j) \\ D(i,j-1) \\ D(i-1,j-1) \end{cases} \quad (8)$$

Error Calculation for segments: The error between the original time series and the approximated segment can be defined as :

$$Error = \sum_{i=a}^{b} |x_i - Approximation(i)| \quad (9)$$

Where [a,b] is the range of the segment being evaluated. xi is the original data point Approximation(i) is the value of the segmented segment at index i. The total error for a left segment and a right segment is computed as: Total Error=$Error_{left}$+$Error_{right.}$ The criteria for deciding whether to accept a segment or perform further segmentation is:

For the left segment:

If $Error_{left}$ <= max error then accept segment

For the right segment:

If $Error_{right}$ <=max error then accept segment

5. Results and Discussion

SMD Dataset [19]:A similar performance trend is observed in the SMD dataset, as detailed in Table I. With the first maximum error value, the Top-Down algorithm outperformed the other two algorithms in 32 out of 45 time series, representing 71.11% of the cases. The Sliding Window algorithm achieved superior performance in the remaining 13 time series, accounting for 28.89%. Conversely, the Bottom-Up algorithm consistently exhibited the poorest performance across all scenarios.

At the seventh maximum error threshold, the Top-Down algorithm maintained its dominance, achieving superior results across all 45 time series. However, it is crucial to acknowledge that the computational time for the Top-Down algorithm was the highest in every scenario, consistently making it the slowest among the three algorithms evaluated.

SKAW Dataset: Figure 1 presents a comparative analysis of three prominent piecewise linear regression (PLR) segmentation algorithms based on the total sum of approximation errors. In this context, lower error values indicate better performance. The experiments were conducted on 10 time series from the SKAW dataset, evaluated across 7 different maximum error

thresholds. Each line in the figure represents the mean approximation errors averaged over the 10 time series.

The results clearly demonstrate that the Top-Down algorithm consistently outperforms the other two algorithms across all time series and for every evaluated maximum error value, with the Sliding Window method ranking as the second-best performer. However, it is important to note that the Top-Down algorithm is significantly slower in terms of computational efficiency; it is approximately 60 times slower than the Sliding Window method and about 10 times slower than the Bottom-Up approach. Notably, the Sliding Window algorithm excels in computational speed while still providing acceptable performance in terms of approximation errors. Figure 2 illustrates the average computational time of the three algorithms across the 10 time series, evaluated over the 7 different maximum error settings.

M4 Dataset [20]: The M4 dataset, consisting of 100 time series from various domains such as finance, tourism, and demographics, further corroborates the findings. Table II summarizes the performance of the segmentation algorithms across different maximum error values. With the first maximum error value, the Top-Down algorithm excelled in 65 out of 100 time series, representing 65% of the cases, while the Sliding Window algorithm performed best in 30 time series (30%) and the Bottom-Up algorithm in 5 time series (5%). As the maximum error value increased, the Top-Down algorithm continued to dominate, particularly at higher thresholds, achieving superior results in 90 out of 100 time series by the seventh maximum error setting.

Despite its strong performance, the computational time for the Top-Down algorithm remained the highest among the three algorithms, echoing results seen in both the SKWA and SMD datasets. Figure 3 illustrates the average computational time for all three algorithms evaluated across the 100 time series over the various maximum error thresholds.

TABLE I: COMPARISON OF ALGORITHM PERFORMANCE BY MAXIMUM ERROR (SMD DATASET)

Algorithm	Top-Down	Sliding Window	Bottom-Up
Max Error 1	32 (71.11%)	13 (28.89%)	0 (0%)
Max Error 2	34 (75.56%)	11 (22.44%)	0 (0%)
Max Error 3	40 (88.89%)	5 (11.11%)	0 (0%)
Max Error 4	43 (95.56%)	2 (4.44%)	0 (0%)
Max Error 5	44 (97.78%)	1 (2.22%)	0 (0%)
Max Error 6	44 (97.78%)	1 (2.22%)	0 (0%)
Max Error 7	45 (100%)	0 (0%)	0 (0%)

This revision adds the M4 dataset and its corresponding analysis, maintaining clarity while providing a thorough examination of the results across multiple datasets. This method begins by modeling the entire time series as one segment and recursively splits it at significant points where the error between the actual and modeled data is maximized. In the figure… blue line represents the original time series data, and the segmented parts highlight where the algorithm identified important change-points.

TABLE II: COMPARISON OF ALGORITHM PERFORMANCE BY MAXIMUM ERROR (M4 DATASET)

Algorithm	Top-Down	Sliding Window	Bottom-Up
Max Error 1	65 (65%)	30 (30%)	5 (5%)
Max Error 2	70 (70%)	25 (25%)	5 (5%)
Max Error 3	80 (80%)	15 (15%)	5 (5%)
Max Error 4	85 (85%)	10 (10%)	5 (5%)
Max Error 5	90 (90%)	5 (5%)	5 (5%)
Max Error 6	90 (90%)	5 (5%)	5 (5%)
Max Error 7	90 (90%)	0 (0%)	0 (0%)

The Candidate-based Top-Down (CBD) method improves efficiency by identifying candidate split points based on deviations from the mean. These yellow highlighted segments reflect the points where the segmentation algorithm finds the largest errors and splits the data accordingly . The recursion continues until the error in each segment falls below a threshold. This ensures that the segmentation captures global trends and structures. In real-world applications like trend analysis or anomaly detection, these splits would represent key changes in data behavior over time, helping in summarizing and interpreting large datasets. The term "Recursion Limit", which aligns with the Top-Down method's recursive splitting process. The recursion continues until the error in each segment falls below a threshold. This ensures that the segmentation captures global trends and structures.

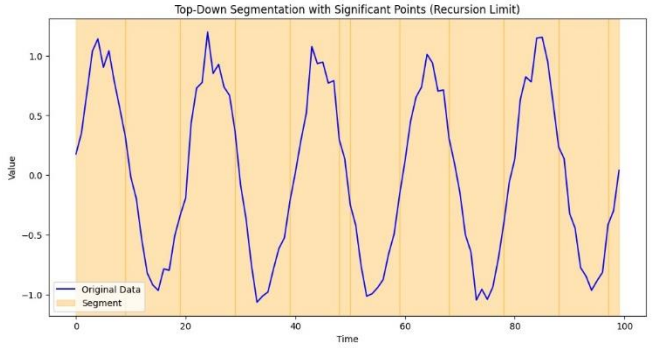

Figure 1: Time Series Analysis

The figure 1 above, showcases how the Top-Down segmentation method (specifically the CBD algorithm) operates by recursively splitting a time series at significant points to minimize approximation error, with the recursion stopping when the error threshold is met.

6. Conclusion

In this paper, we conducted a comprehensive evaluation of various time series segmentation methods, with a particular focus on the computational efficiency of the widely used Top-Down approach. Our proposed method, Candidate-based Top-Down (CBD), leverages statistical insights and significant point identification to efficiently determine optimal segmentation boundaries. This approach allows CBD to maintain the high accuracy traditionally associated with Top-Down methods while significantly reducing computational time.

Experimental results across multiple datasets demonstrate that CBD achieves approximation errors comparable to those of the conventional Top-Down method. Furthermore, CBD consistently outperforms other segmentation techniques, such as Sliding Window and Bottom-Up methods. These findings suggest that CBD serves as a robust and practical alternative for time series segmentation, applicable across various real-world scenarios.

While the CBD algorithm shows significant promise, several avenues for future research remain. Firstly, further optimization of the algorithm could be explored, particularly in the context of high-dimensional time series data, to enhance its scalability and applicability. Additionally, integrating machine learning techniques within the CBD framework presents an exciting opportunity to automate parameter adjustments, such as window size and threshold values. By dynamically adapting these parameters based on the characteristics of the dataset, the algorithm could improve its effectiveness across diverse domains, ensuring greater adaptability and performance in real-world applications.

References

1. A. M. Abdullahi, Y. Baashar, H. Alhussian, A. Alwadain, N. Aziz, L. F. Capretz, and S. J. Abdulkadir, "Detecting cybersecurity attacks in Internet of Things using artificial intelligence methods: A systematic literature review," Electronics, vol. 11, no. 2, p. 198, 2022. doi: 10.3390/electronics11020198.
2. O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019," Applied Soft Computing, vol. 90, p. 106181, 2020.
3. S. Kaushik, A. Choudhury, P. K. Sheron, N. Dasgupta, S. Natarajan, L. A. Pickett, and V. Dutt, "AI in healthcare: Time-series forecasting using statistical, neural, and ensemble architectures," Frontiers in Big Data, vol. 3, p. 4, 2020.
4. T.-c. Fu, "A review on time series data mining," Engineering Applications of Artificial Intelligence, vol. 24, no. 1, pp. 164–181, 2011.
5. A. Carmona-Poyato, N. L. Fernández-García, F. J. Madrid-Cuevas, and A. M. Durán-Rosal, "A new approach for optimal time-series segmentation," Pattern Recognition Letters, vol. 135, pp. 153–159, 2020.
6. E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: A survey and novel approach," in Data Mining in Time Series Databases, World Scientific, 2004, pp. 1–21.
7. E. Keogh and J. Lin, "Clustering of time-series subsequences is meaningless: Implications for previous and future research," Knowledge and Information Systems, vol. 8, pp. 154–177, 2005.
8. E. Keogh and M. Pazzani, "Derivative dynamic time warping," in Proc. First SIAM Int. Conf. Data Mining, 2002, p. 1.
9. W. Shi, G. Azzopardi, D. Karastoyanova, and Y. Huang, "Bidirectional piecewise linear representation of time series with application to collective anomaly detection," Advanced

Engineering Informatics, vol. 58, p. 102155, 2023.
10. Z. Zhou, M. Baratchi, G. Si, H. H. Hoos, and G. Huang, "Adaptive error bounded piecewise linear approximation for time series representation," Engineering Applications of Artificial Intelligence, vol. 126, p. 106892, 2023..
11. W. Shi, D. Karastoyanova, Y. Huang and G. Zhang, "Bidirectional Piecewise Linear Representation of Time Series and its Application in Clustering," in IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1-13, 2023, Art no. 2527913, doi: 10.1109/TIM.2023.3318728.
12. P. Trirat, Y. Shin, J. Kang, Y. Nam, J. Na, M. Bae, J. Kim, B. Kim, and J.-G. Lee, "Universal time-series representation learning: A survey," arXiv preprint arXiv:2401.03717, 2024.
13. S. Ahmed, R. K. Chakrabortty, D. L. Essam, and W. Ding, "Poly-linear regression with augmented long short term memory neural network: Predicting time series data," Information Sciences, vol. 606, pp. 573-600, 2022, doi: 10.1016/j.ins.2022.01.042.
14. A. Casolaro, V. Capone, G. Iannuzzo, and F. Camastra, "Deep learning for time series forecasting: Advances and open problems," Information, vol. 14, no. 11, p. 598, 2023, doi: 10.3390/info14110598..
15. D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in KDD Workshop, 1994.
16. H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. H. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, et al., "Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications," in Proc. 2018 World Wide Web Conf. on World Wide Web, 2018, pp. 187–196.
17. K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," in Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining (KDD '18), New York, NY, USA, pp. 387–395, 2018.
18. P. Esling and C. Agon, "Time-series data mining," ACM Computing Surveys (CSUR), vol. 45, no. 1, pp. 1–34, 2012.//not cited
19. R. Panciera et al., "The Soil Moisture Active Passive Experiments (SMAPEx): Toward Soil Moisture Retrieval From the SMAP Mission," in IEEE Transactions on Geoscience and Remote Sensing, vol. 52, no. 1, pp. 490-507, Jan. 2014, doi: 10.1109/TGRS.2013.2241774.
20. Z. Zamanzadeh Darban, G. I. Webb, S. Pan, C. C. Aggarwal, "CARLA: Self-supervised contrastive representation learning for time series anomaly detection," Pattern Recognition, vol. 157, p. 110874, 2025.
21. S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 competition: Results, findings, conclusion and way forward," International Journal of Forecasting, vol. 34, pp. 10–19, 2018. doi: 10.1016/j.ijforecast.2018.06.001.