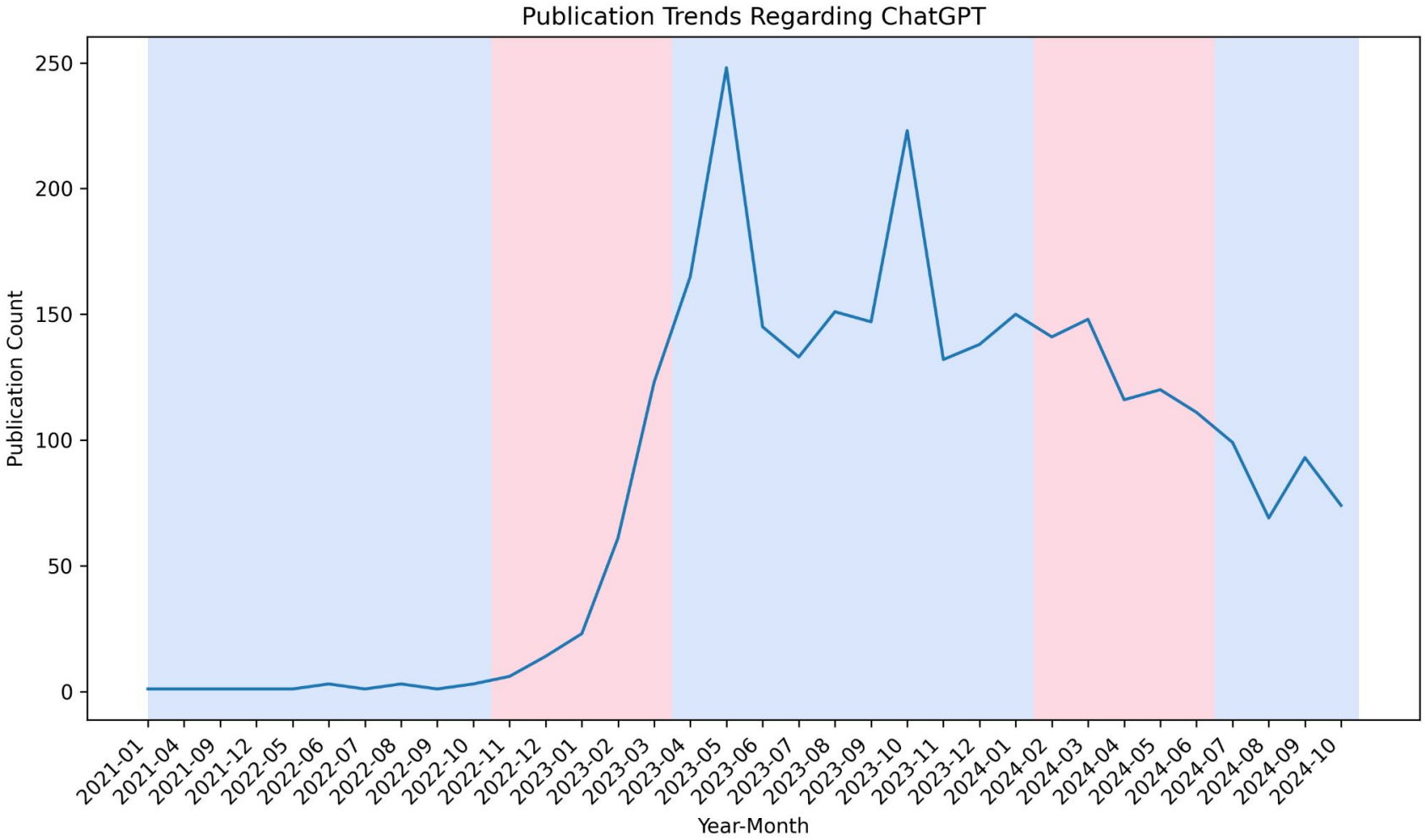
Math x AI Hackathon ♾️

Signal Select

Can VLMs evaluate change point detection models?


Publication Trends Regarding ChatGPT
Publication Count
0
50
100
150
200
250
2021-01
2021-04
2021-09
2021-12
2022-05
2022-06
2022-07
2022-08
2022-09
2022-10
2022-11
2022-12
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
2023-08
2023-09
2023-10
2023-11
2023-12
2024-01
2024-02
2024-03
2024-04
2024-05
2024-06
2024-07
2024-08
2024-09
2024-10
Year-Month

Change Point Detection

Change point detection is a statistical technique used to identify points in a time series where the underlying statistical properties of the data change significantly. In simple mathematical notation, we can describe change point detection as follows:

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a time series of length n. The goal of change point detection is to find a set of change points $\tau = \{\tau_1, \tau_2, \ldots, \tau_m\}$ where $1 < \tau_1 < \tau_2 < \cdots < \tau_m < n$. These change points divide the time series into $m + 1$ segments:

$$[1, \tau_1), [\tau_1, \tau_2), \ldots, [\tau_m, n]$$

https://github.com/deepcharles/ruptures

PELT: Pruned Exact Linear Time

Because the enumeration of all possible partitions impossible, the algorithm relies on a pruning rule. Many indexes are discarded, greatly reducing the computational cost while retaining the ability to find the optimal segmentation. The implementation follows [Killick2012]. In addition, under certain conditions on the change point repartition, the avarage computational complexity is of the order of $\mathcal{O}(CKn)$, where K is the number of change points to detect, n the number of samples and C the complexity of calling the considered cost function on one sub-signal.

PELT

Mathematical Formulation

PELT aims to minimize a penalized cost function:

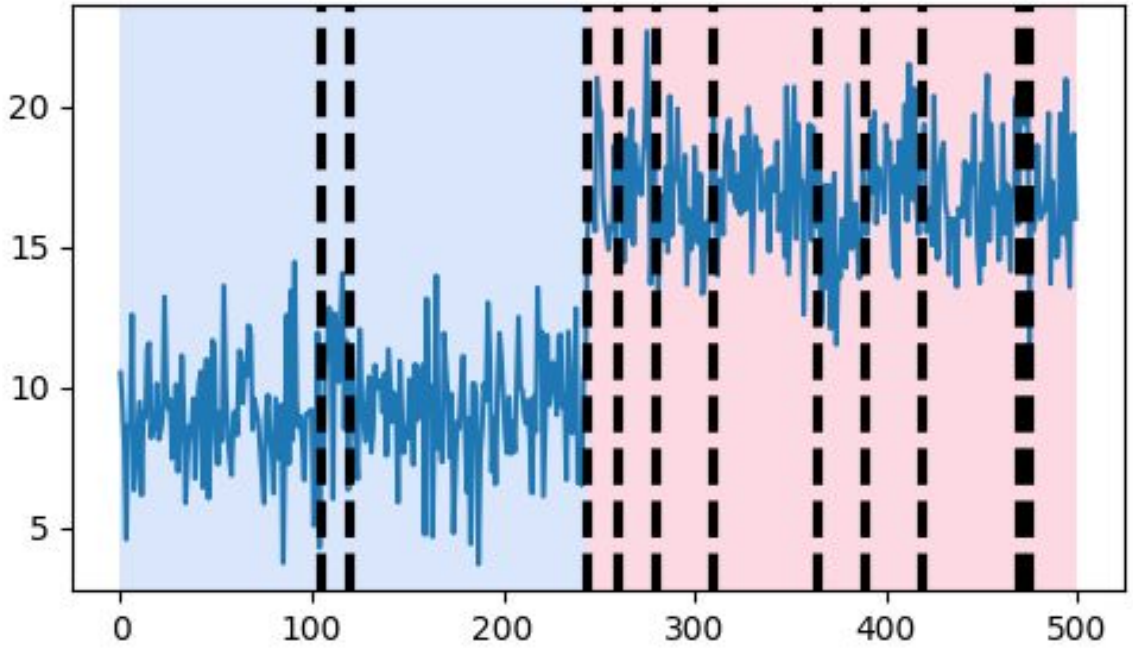
$$\min_{K,\tau_1,\ldots,\tau_K} \left[\sum_{i=1}^{K+1} \mathcal{C}(x_{(\tau_{i-1}+1):\tau_i}) + \beta K \right]$$

Where:

- K is the number of change points
- $\tau_1, \ldots, \tau_K$ are the positions of the change points
- $\mathcal{C}(\cdot)$ is a cost function for a segment
- β is a penalty to prevent overfitting

The algorithm efficiently solves this optimization problem by using dynamic programming and applying a pruning rule to reduce the search space.

Hyperparameter Tuning can be a Challenge!



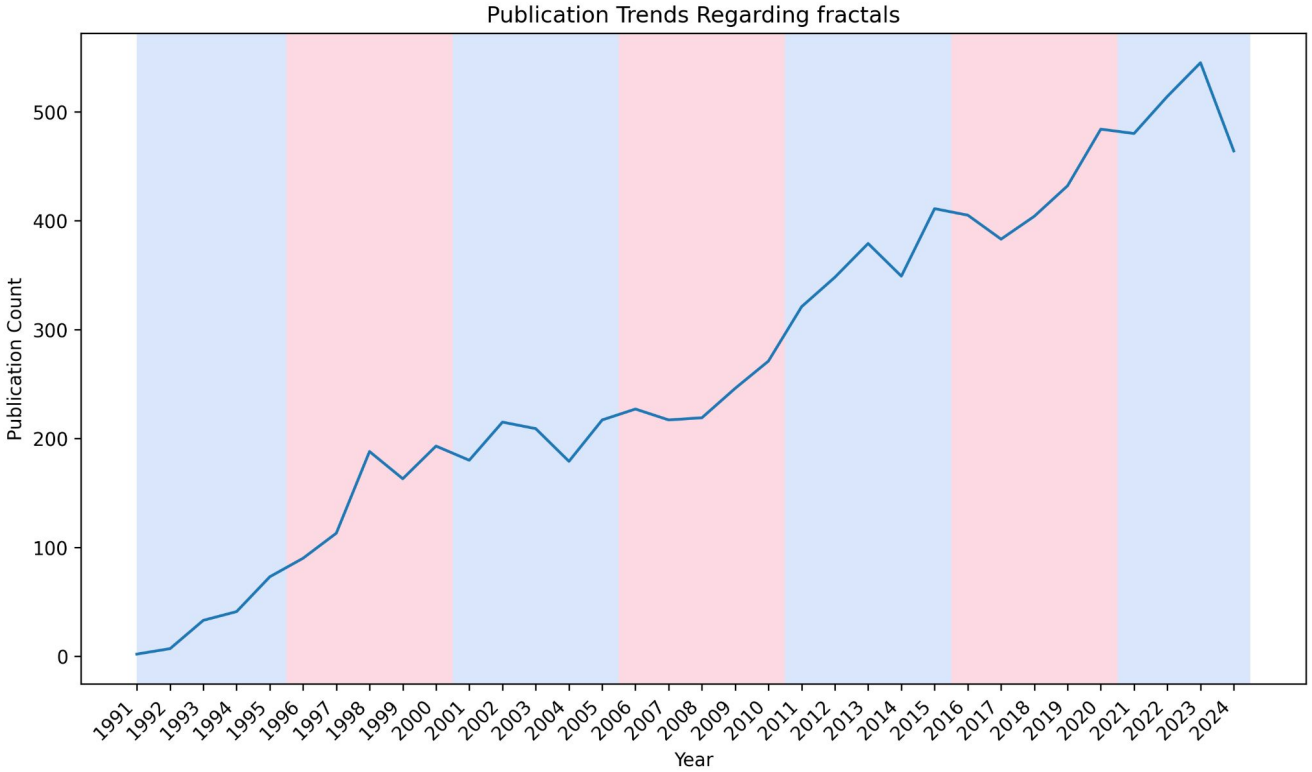
1. **Number of Dotted Lines**: There appear to be multiple dotted lines, which suggests that there may be multiple change points identified. For binary segmentation, ideally, there should only be one dotted line.

2. **Location of Dotted Lines**: The placement of the dotted lines should ideally separate the two segments clearly. It seems like the lines are spread across the range rather than demarcating a clear change.

Based on these observations, I would rate the appropriateness of the change point detection as a **2 out of 10**. The detection is poor because it does not adhere to the requirement for change point segmentation.

Final score: **0** (indicating that the change point detection was poor).

PELT can be Informative



"evaluation": {

"dotted_lines_placement": "The dotted lines appear to delineate segments that show differing trends in the data, indicating potential change points.",
"trend_shift": "A noticeable trend shift occurs at the segments, with increasing values in later segments."
},
"final_evaluation": 8,
"binary_indicator": 1
}