Algorithmic Adaptive Podcast Highlight Generator Pitch

Koussay Jaballah (AUC)

Spring 2025

I. Problem Statement & Application Context

Problem: Listeners often skip through long podcasts. Create a feature that automatically generates highlight reels based on listener engagement patterns.

Social media is widely used and the way we use it has been crucial in determining someone's time management and influence on wellbeing. That's why choosing specific slots of the video to watch has become more and more important. With no clear indication of where the most interesting or popular content lies, many users either abandon the episode or randomly skip ahead, leading to a suboptimal listening experience.

Solution:

An adaptive podcast highlight generator that automatically identifies and extracts the most engaging parts of a podcast based on real-time user interaction data (such as pauses or replays). This feature would fit naturally into media and entertainment applications like Spotify …

Application Context:

- Videos that provide data analytics to their views and the analytics of the video to analyze and produce the output

Formal Problem Definition:

Input:

- Total length of podcast in seconds
- Engagement scores for each 30-second segment (derived from user interaction analytics)
- A constant k indicates the number of top highlights to extract

Output:

- A list of k segments with the highest engagement

Constraints:

- Segments are fixed for 30 seconds
- Engagement scores are positive integers

II. Two Algorithmic Approaches

- **Greedy Approach:**

This greedy algorithm processes podcast segment engagement scores **sequentially** and maintains a **min-heap** of the top k most engaging segments.

1. Loop through each engagement score.
2. Maintain a min-heap of size k:

- If fewer than k elements, push the current segment.
- Else, if current engagement is higher than the smallest in the heap, replace it.

3. At the end, you get the top k segments with the highest engagement

Time Complexity:

Each of the n segments is processed in O (log k)

Total time complexity**:** O (n log k)

Space Complexity:

Heap size: O(k)

- **Divide and Conquer:**

1. Splitting the podcast into 30-second segments based on the total length in seconds.
2. Collecting engagement scores for each segment.
3. Using a Divide and Conquer approach to recursively:

➔ Break the array of engagement scores into halves,
➔ Find the top k (3) most engaging segments in each half
➔ Merge the results and keep the overall top k.

Time Complexity:

Sorting typically takes O (n log(n)) and merging the two arrays takes O(n). Thus, total time complexity is O (n log(n))

Space Complexity:

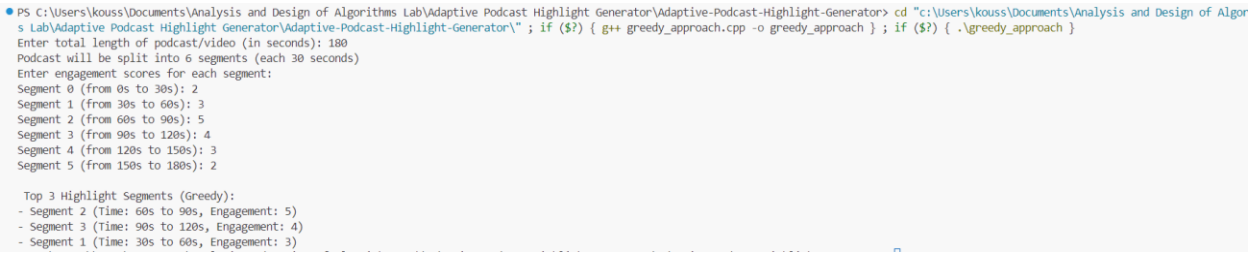
O(n) for storing the engagement array

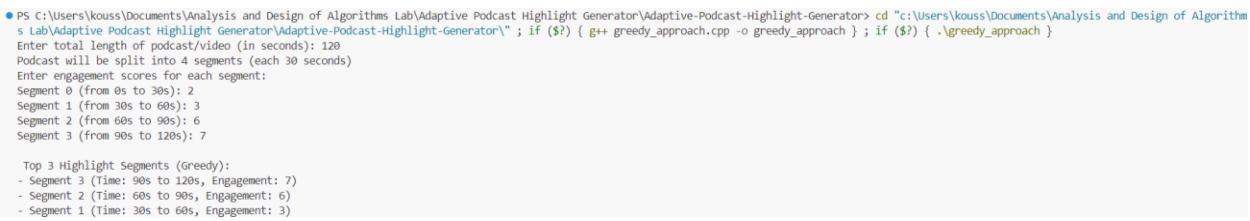
III. Situational Evaluation

Scenario	Approach	Justification
Real-time playback analysis	Greedy	Fast, low overhead, ideal for continuous streams
Offline highlight generation	Divide & conquer	Full dataset available, needs accuracy
Small podcasts (few segments)	Either	Negligible difference
Large podcasts with many users	Greedy	Scales better with size

IV. Sample Test Cases & Demonstration

Test cases are t=180s, 90s, 120s





V. Reflection

I collected the following runtime for the program:

n = 10, 100, 1000

Greedy Time (ms) = 0.04, 0.5, 4.1

Divide and Conquer (ms) = 0.12, 1.8, 22.3

Challenges:

- Modeling real-world engagement data (simulated manually for the prototype)
- Ensuring fair evaluation between two approaches with different strengths

Assumptions:

- Engagement data is already pre-processed and available
- All segments are uniformly 30 seconds

Limitations:

- Does not account for context across segments (continuity in conversation)
- Assuming engagement score is an accurate proxy for interest

Possible Enhancements:

- Integrate with a real engagement tracking system
- Apply NLP to label highlights with keywords
- Add UI to visualize highlights on a timeline

VI. Final Notes

➔ The adaptive Podcast Highlight Generator is a useful tool that may be improved and optimized so that it becomes very useful software. We need to have analytics from media providers like YouTube and Spotify. There is also the possibility this tool can be integrated into the videos before playing them.

➔ The algorithms implemented are simple in terms of the complexity of the ideas. However, analyzing the analytics of the data can be challenging (may need Big Data algorithms) to detect accurately the results we need.