Find The Episode!

Sreeja Cheekireddy  
 *Computer Science*,

*Southern Illinois University Edwardsville*Edwardsville, USA  
[scheeki@siue.edu](mailto:scheeki@siue.edu)

*Abstract*— Video-sharing platforms, like YouTube, host a vast amount of content, including ‘clips’ from TV shows and web series created by YouTubers for various purposes. These clips often lack specific information about the source material, making it challenging for viewers to identify the original content and its season and episode details. To address this problem, comments from these videos are used. YouTube Comments API can be the source to retrieve these comments. Various algorithms can then be used to process these comments to identify the episode and season number along with the series name. Initially, comments are filtered based on keywords. The methods use this data and finally metrics are used to compare them. The pros and cons of these methods are analyzed.

# **Big Problem**

YouTube and other video-sharing platforms have a massive amount of content. This content also includes TV shows or web series. Though the source of the original content is hosted elsewhere (other streaming websites, depending upon the show/series), it is common for some YouTube content creators(‘YouTubers’) to create clips (1 minute or fewer videos) with scenes from the shows. This is done for viewership purposes or to attack people to that series (marketing purposes).   
For example, a clip can be created using a scene from the series ‘Good Doctor’ Season 3 Episode 3 by a YouTuber. But this information about the actual show may or may not be provided by the creator in the description or the title. There might be multiple reasons for this issue. Often, these clips are created in bulk and have a generalized description about the series and links to that, but not specific to that particular episode. Also, some clips are created by people who do not have proper copywrites. Such creators do not add any information about the clips. They do so only for the purpose of viewership and money.

As a content viewer, this leads to a lot of curiosity, especially if the shot is interesting. It is often natural to have a feeling to watch the entire episode. The viewer would like to find information about the origin of the shot. But it is hard to find the season and episode number of the series.

In order to find the season and episode information, a comments list can be a useful source (if not provided in the description or title). Different methods can be tried for this use case. So, to solve this problem, Different methods like greedy ,divide and concur,KMP will be used. The methods will be compared.

# **Dataset**

A set of 10 shots that contain scenes from some medical series are selected for the dataset. Most of the medical series deal with similar stuff and often difficult to identify which is which. These videos are solely selected based on personal interest. Table 1 and Table 2 contain the ground truth.

The comments are gathered using YouTube Data API called “CommentThreads: list.” The video ID must be provided for the request. 100 comments can be retrieved at a time. Figure 1 is a sample of the JSON retrieved from the API request. Figure 2 contains some screenshots of sample videos from the dataset.

Different parameters can be used while performing the API request. For example, Only Comments containing certain keywords can be retrieved. This is helpful in many ways. Also, The comments can be sorted by relevance and retrieved.

Hence, for each video ID, 3 different JSONs were created by doing various requests.

* **“1.json”:** This contains the first 100 comments from the default API request using that VideoID
* **“episode.json”:** The *searchTerms* parameter is used in the API request. It is made equal to “episode.” Hence the name for the JSON. As usual, only the first 100 comments with “episode “ in them are retrieved.
* **“Relevance.Json”:** The *order* parameter is set to “relevance” here.

More comments are retrieved only if the answer is not found in the first 100.

This can be done using the *pageToken* and *nextPageToken* parameters in the API request.

The following is the dataset structure:

>>**Dataset**

>> =>**VideoID name**

>> => => **1.json**

>> => => **episode.json**

>> => => **relevance.json**

Similarly based on the requirement more JSONs can be created.

Like using “series” keyword etc.

A screenshot of a computer program

Description automatically generated

**Figure 1: JSON sample from the API request**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Video Id | Link | Series | Season | Episode |
| IAZSqXhVgk0 | <https://www.youtube.com/shorts/IAZSqXhVgk0> | Good Doctor | 1 | 7 |
| oHXl4mKTHbA | <https://www.youtube.com/shorts/oHXl4mKTHbA> | Good Doctor | 1 | 2 |
| nK51l6GSkG4 | <https://www.youtube.com/shorts/nK51l6GSkG4> | Grey's Anatomy | 18 | 13 |
| fYJDDLB45c8 | <https://www.youtube.com/shorts/fYJDDLB45c8> | House | 3 | 12 |
| RYYJnYLMbzQ | <https://www.youtube.com/shorts/RYYJnYLMbzQ> | Good Doctor | 3 | 3 |
| phOHc0u-5DA | <https://www.youtube.com/shorts/phOHc0u-5DA> | Grey's Anatomy | 12 | 16 |
| 2j7apjKaemo | <https://www.youtube.com/shorts/2j7apjKaemo> | House | 1 | 4 |
| DODBr-8Cw\_U | <https://www.youtube.com/shorts/DODBr-8Cw_U> | Good Doctor | 3 | 5 |
| t9ntcd-hfL8 | <https://www.youtube.com/shorts/t9ntcd-hfL8> | Good Doctor | 1 | 6 |
| bBNyWRe5KVA | <https://www.youtube.com/shorts/bBNyWRe5KVA> | House | 5 | 10 |
|  | **Table 1:**  **Dataset with Ground Truth** |  |  |  |

A screenshot of a video chat

Description automatically generatedA screenshot of a video chat

Description automatically generated

**A screenshot of a video chat

Description automatically generated**

**Figure 2 :Screenshots of YouTube videos from**  **Good Doctor, Grey's Anatomy and House(Top to Bottom)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

# **Methodology**

After the API calls, the JSONs are stored in the respective folders.

These JSONs must be processed to get only the text values of the comments.

Figure 3 contains the flowchart of the entire procedure.

A diagram of a flowchart

Description automatically generated

**Figure 3: Flow chart of the process**

Three different methods are used to solve this problem. The results of these will be compared and contrasted.

* **Divide and Conquer:**

Comments can be divided and processed to find what is needed. Here, natural language processing (NLP) can be used. Some keywords like “episode,” “season,” show”, etc. can be used for the search. Regex can be used here.

“episode.json” would be of great use here, as we will be only looking at comments that have ‘episode’ in them. The Comments list can be processed using divide and conquer.Time complexity is O(n \* log(n) \* p), where n is the number of comments and p is the number of patterns.

Figure 4 is the pseudocode of this method.

A screenshot of a computer program

Description automatically generated

**Figure 4: Pseudocode of divide and conquer**

* **Greedy strategy:**

It is observed that episode/season information is often given in the comments after a question regarding the same.

For example, if “What is this episode?” is the comment. The replies to this comment may have the correct information. Using the greedy method, we can first search for those replies. If we get an answer here, we don’t search the remaining quires.

A screen shot of a computer

Description automatically generated

**Figure 5: Pseudocode of Greedy method**

* **Most Commonly used:**

The most commonly used numbers and words are retrieved from all the comments to gather essential words about the series. For example, if ‘3’ is very commonly used, it might be a season number. The actor’s information may also be available in this way.

2 most commonly used numbers are retrieved. They can be treated as episode and season numbers.

Also, most commonly used word is retived, it is “series” name.

But we must give in advance a list of words to be ignored (like pronouns, articles etc.).

A screenshot of a computer

Description automatically generated

**Figure 6: Pseudocode of Most Commonly used**

* **Knuth-Morries\_Pratt Algorithm [3]:**

This algorithm can be used to search through the comment strings to find the series name and episode number. This method has better worst case time complexity than brute force[3]. It is essential to compare this method with the others to prove the same.

Search happens using a pattern “series.” Hence used to find only series name. A list of comments is returned, manually we need to identify the series name based on those. The time complexity is O(n \* (n + p))which is better worst-case time complexity than that of brute force. Figure 7 is the Pseudocode of the KMP algorithm.

A screenshot of a computer program

Description automatically generated

**Figure 7: Pseudocode of KMP**

Based on the video and its comments, different methods can be used. Sometimes, more than one method may also be required. In some cases, the Greedy strategy may seem fast as it searches only the critical comments and replies, but this may not always be optimum. KMP can be faster than brute force. But It might not be reliable. It is necessary to compare and contrast between the above-mentioned methods. The metrics will include episode Accuracy, season accuracy and series accuracy.

# **Experiment design**

The first thing that comes to mind is that, does the number of comments matter? In other words, a video with around 6000 comments has a greater chance to find the episode number than the episode with only some hundred comments. So, to check this, a greedy approach was applied on the videos in the dataset, their correct responses were then checked with the number of comments of that dataset. Table 3 gives the results of these experiments.

Another question that arises is,is any one method enough to get all the information? Is greed or divide and conquer or any of the other two methods alone is sufficient to get the results? To check that the individual metrics are retrieved. Also, the advantages and disadvantages of all these methods are also considered.

All the methods use the “episode.json” file. This means that the comments are filtered based on the “episode” keyword beforehand. Overall comments are not investigated. Is epiode .json alone sufficient to find the results? Experiments are conducted only using episode.json, to check if either of the methods get the right information about the video.

**Table 3: Results of Greedy Method Using episode.json**

# **Results and discussion**

As seen in Table 3, the results for the greedy method, the number of comments does not affect the results. The video of video id “bBNyWRe5KVA**”** has only 722 comments**.** Greedy could correctly identify the episode number and season number. Whereas “RYYJnYLMbzQ” with 6600 comments was not enough for greedy to identify the results. So quality over quantity matters here. Episode.json was enough for most except 2 videos (RYYJnYLMbzQ,t9ntcd-hfL8).

Now when all the other methods are tried, it is clear that episode.json is enough. Table 4 gives all the responses from all the methods. The KMP is only for series identification, it does not predict episode/season numbers. Finally, metrics are calculated.

Table 5 gives the percentages of the accuracy for episode, season and series. Based on the accuracy it is clear that divide and conquer is best while trying to find out episode number and season number. But for Series name we need KMP or most commonly used. In KMP a little bit of manual effort is required at the end. But most commonly used, the major disadvantage is that only one word of the series name is displayed.

Table 6 clearly explains the advantages and disadvantages of all the methods used in this project.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Video Id | Approximate number of comments as of 9/11/2023 | Episode identified | Season Identified | Found in episode.json? |
| IAZSqXhVgk0 | 2000 | 7 | 1 | Yes |
| oHXl4mKTHbA | 2600 | 2 | 1 | Yes |
| nK51l6GSkG4 | 8900 | 13 | 18 | Yes |
| fYJDDLB45c8 | 759 | 12 | 3 | Yes |
| RYYJnYLMbzQ | 6600 | - | - | Yes |
| phOHc0u-5DA | 1000 | 16 | 12 | Yes |
| 2j7apjKaemo | 9800 | 4 | 1 | Yes |
| DODBr-8Cw\_U | 383 | 5 | 3 | Yes |
| t9ntcd-hfL8 | 21 | 6 | - | Yes |
| bBNyWRe5KVA | 722 | 10 | 5 | Yes |

A screenshot of a computer

Description automatically generated

**Table 4: Results of all methods**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** |
| **Episode** | **90** | **90** | **80** | **-** |
| **Season** | **80** | **90** | **60** | **-** |
| **Series** | **0** | **10** | **30** | **30** |

**Table 5: Accuracy across all methods**

|  |  |  |
| --- | --- | --- |
|  | **Advantages** | **Disadvantages** |
| **Divide and conquer** | Can be done in parallel | Stack overflow might occur. |
| **Greedy** | Simple strategy | Not always right |
| **Most commonly used** | Successful if we ignore certain words | There is no distinction between season number and episode number. |
| **KMP** | Used for series identification | Manual intervention is required after searching for the series |

**Table 6: Advantages and disadvantages of the methods**

# **Gaant Chart**

Created Dataset using the Youtube API requests by video id.  
Completed the development of all the methods as per the schedule. Also added a new method called KMP and did analysis.

The following is the gaant chart.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Find the Episode! | | | DAYS |
| START DATE | END DATE | DESCRIPTION |
| 9/18/23 | 9/27/23 | Comments Gathering | 9 |
| 9/28/23 | 10/5/23 | Greedy strategy | 7 |
| 10/1/23 | 10/11/23 | Divide and conquer | 10 |
| 10/12/23 | 10/18/23 | Mid Report prep | 6 |
| 10/19/23 | 10/25/23 | Most commonly used | 6 |
| 10/26/23 | 11/1/23 | Comparitive analysis | 5 |
| 11/2/23 | 11/12/23 | final tests | 10 |
| 11/13/23 | 11/19/23 | Report writing | 6 |
| 11/20/23 | 11/26/23 | Presentation and report | 6 |
|  |  |  |  |

# **conclusion and future work**

The project is very interesting to work on, but the comment-gathering process was tough. Also, since most of the work includes working with JSONs, it has become increasingly tough to work using C++.

After careful analysis of all the methods, Divide and conquer is best for episode and season identification. “Most commonly used” is better for series identification.

In future, these methods can be used for various other videos from other platforms. Thumbnails and subtitles can be used to search further using google image API. Wikipedia and IMDb can be other sources while searching. That means other APIs can also be explored. Machine learning Algorithms can also be tried.

##### Acknowledgment

I would like to thank Dr Rubi Quinones for accepting the proposal for this project. She helped me a lot by giving regular feedback on the approaches and methods.

##### **References**

1. Google Developers. "Comments." YouTube Data API, 23 Sept. 2023, <https://developers.google.com/youtube/v3/docs/comments>.
2. Google Developers. "Comment Threads." YouTube Data API, 4 Nov. 2022, <https://developers.google.com/youtube/v3/docs/commentThreads>.
3. Wikipedia contributors. "Knuth–Morris–Pratt Algorithm." Wikipedia, <https://en.wikipedia.org/wiki/Knuth%E2%80%93Morris%E2%80%93Pratt_algorithm>.

**APPENDIX**

Loved the entire course. Had feedom to come up with different projects such as this. Also the coding assignments helped me learn c++ among other things. Had support throghout the making of this project.