Developing a Specialized Search Engine for True Crime Datasets

Davide Pinto Teixeira
Faculdade de Engenharia da Universidade do Porto
Marco de Canaveses, Portugal
up202109860@fe.up.pt

Pedro Filipe Pinho Oliveira
Faculdade de Engenharia da Universidade do Porto
São João da Madeira, Portugal
up202108669@fe.up.pt

João Guimarães Mota
Faculdade de Engenharia da Universidade do Porto
Couto de Cucujães, Portugal
up202108677@fe.up.pt

João António Teixeira Coelho
Faculdade de Engenharia da Universidade do Porto
Matosinhos, Portugal
up202004846@fe.up.pt

ABSTRACT

The popularity of true crime content on YouTube has led to the creation of an information retrieval system for easy access to real-life videos. This study examines a dataset of 21 YouTube channels, including metadata like views, likes, comments, captions, and transcripts, to analyze video performance and audience engagement. The research evaluates the system's performance, emphasizing the importance of a robust indexing and querying strategy. Results show that an improved schema, compared to a basic one, as well as advanced queries that make use of Solr's search functionalities and semantic search provide more relevant results, as demonstrated by query metrics and P-R Curves.

KEYWORDS

Information Processing, Information Retrieval, Web Scrapping, Data Analysis, Dataset, JSON, True Crime, Youtube, Transcript

1 INTRODUCTION

In recent years, interest in true crime has grown significantly, attracting a global audience from various backgrounds[7]. Platforms like YouTube have played a big role in this surge, becoming a favorite space for creators to share stories about mysteries, cases, and criminal investigations. True crime content has become a thriving genre on YouTube, drawing in millions of viewers and sparking discussions across the platform.

With so much content being produced, this growing interest also provides a chance to better understand how people engage with true crime stories online. To support this, a dataset[3] was created, featuring more than 10,000 videos from 21 YouTube channels dedicated to true crime. This dataset includes information like views, likes, comments, captions, and transcripts, offering a detailed look into how true crime videos perform and connect with their audiences. It opens the door to exploring trends, audience interactions, and the overall impact of true crime content in the digital world.

2 DATASET

The dataset chosen to be explored for the project was the True Crime Channel Statistics[3]. Table 1 has the structure of the raw dataset that came directly from Kaggle[11] platform.

PRI'24, February 4, 2025, Porto, Porto, Portugal
2024.

Table 1: Kaggle's raw dataset[3].

Title	Published Date	Views	Likes	Month	Channel	Comments	Description
String	String	Integer	Integer	String	String	Integer	String

2.1 Origin

Kaggle[11] is the platform that hosts this dataset. The dataset has a collection of over 10000 true crime videos from 21 different YouTube channels. This data was collected by its owner using the YouTube API[19]. The dataset came in a CSV file format.

2.2 Pipeline

Figure 1 has the overall pipeline of the system. From Kaggle data retrieval[11] to the final JSON file ready to be processed.

The system starts with the initial dataset, in a comma-separated values format. The dataset was taken from Kaggle with that original format. The first processing step of this dataset was to transform the CSV file into a JSON format file, to create a Non-Relational scheme. The JSON format was chosen to simplify document manipulation and allow the use of document-oriented database technologies, like MongoDB[2].

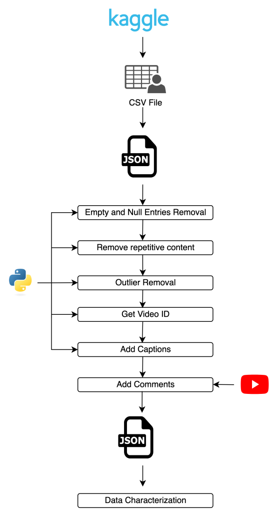


Figure 1: Data processing pipeline. From source to final useful dataset.

After this format transformation, the next step was to remove the empty and NULL values from the dataset. For that task, the Python Programming Language[15] was used.

After this first pre-processing part, the next step was removing repetitive content and some outliers from the dataset. Outliers represent some videos that belong to some true crime channels but are not considered true crime videos. This processing was also performed with the Python Programming Language.

2.3 Cleaning Process

To improve the dataset's quality, a few cleaning steps were applied. One issue was the presence of missing values in important columns like video titles, descriptions, and published dates. Depending on how crucial the missing data was, we either filled in defaults or removed the rows entirely.

We also found duplicate entries, mainly based on Video ID and published date. These were likely due to errors during data collection or enrichment, so we wrote a Python script to identify and remove them.

Categorical variables, such as channel names and video categories, were standardized to fix inconsistencies in formatting, like case differences, extra spaces, and varied naming conventions. This made the data easier to filter and analyze.

Upon closer examination of the dataset, it was evident that many video descriptions contained irrelevant content unrelated to the core focus of the study. These descriptions often followed predictable patterns, such as recurring social media links or separators unique to certain channels. To address this, a specialized tool (written in Python 3[15]) was employed to recognize and remove these recurring elements, ensuring that only meaningful and relevant information was preserved. This process was crucial in refining the dataset and increasing the quality of data used for analysis, particularly by eliminating noise such as promotional material and unrelated content from video descriptions.

Lastly, we normalized numerical columns like views, likes, and comments. Outliers were addressed by either removing them or capping them at reasonable limits to prevent them from distorting the analysis. After these cleaning steps, the dataset was much more reliable and ready for the next stages of processing.

2.4 Data Collection and Enrichment

As previously discussed, the original dataset sourced from Kaggle was provided in CSV format. However, to better align with the evolving needs of this project, the dataset was converted to JSON. This decision was based on the growing reliance on tools and libraries that favor JSON due to its flexibility and ease of use, especially in web applications, machine learning workflows, and RESTful APIs. The hierarchical structure of JSON also allowed for more complex data representation, essential for future data analysis.

To enrich the textual content and provide more context to the data, additional information such as user comments and video captions were incorporated.

Video IDs, a core component of this enrichment, were obtained through web scraping[10]. By automating the process of searching for each video by title, we could extract the video ID directly from the HTML response. These IDs were then used to interact with the YouTube Data API. The API provided access to metadata like comments, likes, and other video-specific statistics. Incorporating this data enhanced the dataset significantly by adding both qualitative (comments and captions) and quantitative (views, likes) attributes, transforming it into a more comprehensive resource for analysis.

The captions of the videos were obtained using the YouTube Transcript API[4], a tool that provides programmatic access to various YouTube resources, including video details, comments, captions, and channel statistics, while the comments were obtained via YouTube Data API[19]. These APIs allowed for a systematic retrieval of user-generated content, which enriched the dataset with insights directly from the viewers of the videos. They required the IDs from the videos in order to fetch their information.

The Data API was not used for other tasks like obtaining video IDs due to its limited quota, which only allows up to 10000 request units per day. For reference, a single search takes 100 of those units.

2.5 Final Dataset

Table 2 describes the structure of the final dataset:

Table 2: Hierarchical structure of the dataset.

Level	**Attribute**
Dataset	
Video	
	Channel
	Title
	Publish date
	Views
	Number of likes
	Number of comments
	Description
	ID
	Transcript
Comment	
	Content
Replies	
	Content

.

2.6 Conceptual Model

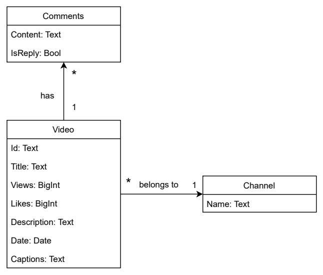


Figure 2: Conceptual data model in UML notation.

The domain model shown in Figure 2 represents the structure of the final dataset. Each video is associated with a channel, and a single channel can host multiple videos. A unique ID identifies these and includes attributes such as title, view count, like count, description, publish date, and captions containing the video's content. Channels are identified by their name. Comments are linked to specific videos and can either be standalone or part of a thread. To distinguish replies from original comments, the reply flag was introduced.

Each video in the dataset is a document. Documents can then be defined as unique JSON objects, with the attributes Channel (string), Title (string), Description (string), Published Data (string), Likes (float), Comments (float), Views (float), transcript (string), comments (list of strings) and videoId (string).

2.7 Data Characterization

As the dataset is about videos published on YouTube, it seems reasonable to examine the channels that published these videos. As mentioned before, this dataset consisted of videos from 21 YouTube channels. When looking at the average views per channel in figures 3 and 4, we can clearly see that 'JCS - Criminal Psychology' is by far the biggest one, having a view average larger than the sum of all other 20 channel's averages.

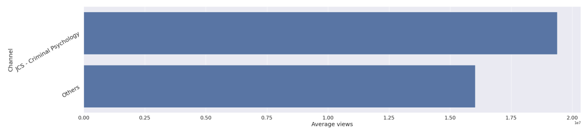


Figure 3: Criminal psychology and all the other channels views comparison.

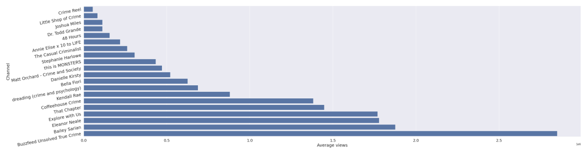


Figure 4: All channels except criminal psychology views comparison.

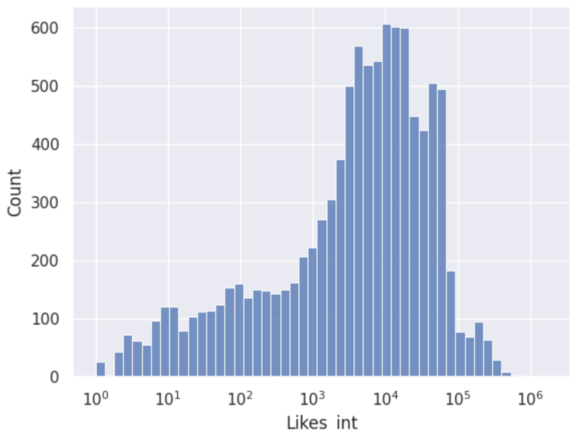


Figure 5: Videos' likes distribution.

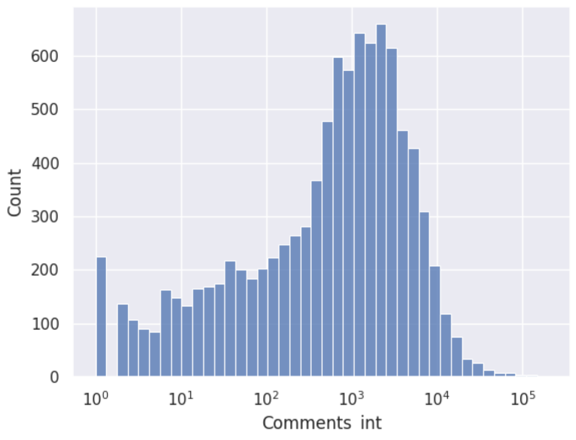


Figure 6: Videos' comments distribution.

Analyzing the videos, we took a look at the distribution of their likes and comments. These distributions are skewed to the right, as shown in figures 5 and 6, most likely as a result of the much larger channel among the 21.

Another metric we took a look at is the year of publication of the video, to analyze the 'recency' of the dataset. As it is possible to verify in figure 7, the majority of the recordings are from 2021, followed by 2020, but we have videos from 2012 to 2022.

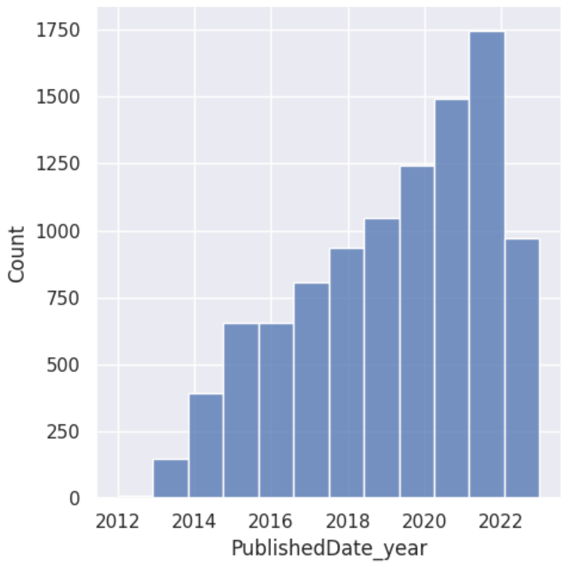


Figure 7: Published date year distribution.

The dataset was not entirely complete after data collection and enrichment; specifically, video transcriptions were lacking. Since the transcriptions are obtained through the captions (whether they are automatically generated or not), the transcriptions for certain videos are absent. Thus, only 6072 (60.08%) of the samples contain transcriptions.

In figure 8, the distribution of the length (in words) of the transcriptions we were able to obtain is visible. The mean value of this statistic is 16752.39, with the longest transcription obtained having 371347 words.

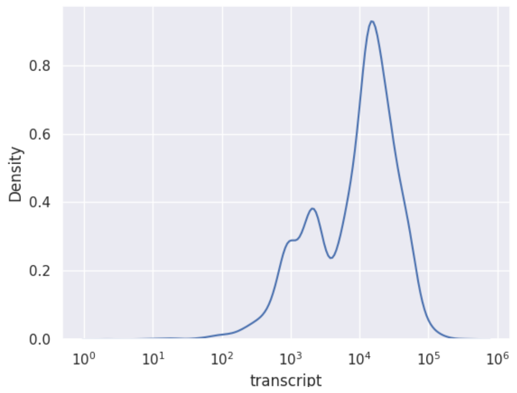


Figure 8: Transcription's length distribution (in words).

Taking a look at the word cloud created from the transcriptions, we can see the main themes of True Crime (figure **??**):

Table 3: Most Frequent Words

Word	Frequency
police	34610
case	24181
family	21034
house	19795
years	18968
life	17667
take	16991
murder	16239
mean	15712
man	14638
made	14507
wanted	14268

2.8 Search Scenarios

In order to explore the extensive collection of true crime videos, we analyze a variety of search scenarios in this study. This dataset offers a diverse range of content, making it ideal for addressing different types of inquiries. To illustrate the versatility of the dataset, we propose four search scenarios.

General Scenarios:

Searching for videos that discuss mysteries. This scenario focuses on finding content that explores cases that remain unsolved, ranging from unexplained crimes that have confused detectives to disappearances.

Searching for videos that discuss murders. In this case, the goal is to filter videos covering homicide cases, offering an overview of this popular subject in this genre.

Specific Scenarios:

Searching for videos about murders that occurred in the U.S. during the 90s. This more targeted scenario seeks to retrieve videos related to murders that occurred exclusively in the United States in the 1990s.

Searching for videos about serial killers from each U.S. state. This scenario narrows down the search to videos discussing infamous serial killers, arranged by place of origin.

These examples illustrate both general and specific queries that demonstrate the variety and depth of analysis that can be done with the dataset.

3 COLLECTION AND INDEXING

This section provides an overview of the process of collecting and organizing video metadata for efficient indexing and retrieval. Structuring the data systematically makes it easier to perform queries, analyses, and operations on the dataset. Each video is a document with detailed metadata, enabling a comprehensive view of its content and associated information.

3.1 Document Definition

Each video is a document.

In this context, each video is defined as a unique JSON object containing structured metadata. The metadata includes essential attributes such as the video's title, channel, description, publication date, engagement statistics, transcript, and comment details. Table 4 presents the fields and their respective descriptions, which serve as a blueprint for understanding and utilizing the video data effectively.

Table 4: Fields and Descriptions for Video Metadata

Field	Type	Description
Channel	string	The name of the video channel.
Title	string	The video's title.
Description	string	The video's description.
PublishedDate	string	The publication date of the video.
Likes	float	The number of likes the video has received.
Comments	float	The number of comments on the video.
Views	float	The total view count of the video.
transcript	string	The transcribed text of the video's content.
comments	list of strings	A collection of individual user comments.
videoId	string	A unique identifier for the video.

3.2 Indexing Process

To index the dataset, a shell script was created to automate the necessary steps using *Solr's*[1] tools and APIs. The script performs the following actions:

(1) **Core Creation**: A new core named *"true_crime"* was created using *Solr's create_core* command;
(2) **Schema Upload**: The schema for the core was uploaded using *Solr's* Schema API to define custom field types and ensure proper processing of the data fields;
(3) **Data Population**: The dataset, formatted as JSON, was uploaded to the core using *Solr's post* tool.

This process ensures the data is correctly indexed and ready for retrieval and querying.

3.3 Schema Details

The schema was defined as a JSON file, where a type for each field was created. For textual fields, such as Title, Channel, or Description, various Tokenizers and Analyzers were utilized to optimize search performance and enhance the user experience when querying the dataset. These analyzers ensured consistency in the text processing pipeline, supporting features like case normalization, stop-word removal, and synonym handling.

The analyzers used were as follows:

(1) **Tokenizer:** `solr.WhitespaceTokenizerFactory` Splits text by whitespace.
(2) **Filter:** `solr.LowerCaseFilterFactory` Converts text to lowercase.
(3) **Filter:** `solr.StopFilterFactory` Removes stopwords using a predefined list.
(4) **Filter:** `solr.SynonymFilterFactory` Expands terms using synonyms.
(5) **Filter:** `solr.ASCIIFoldingFilterFactory` Converts accented characters to their ASCII equivalents.

Below is a list with every type created and their description:

Table 5: Schema Field Definitions

Field Name	Field Type	Multi-Valued	Stored	Indexed
Channel	string	No	Yes	Yes
Title	text_general	No	Yes	Yes
PublishedDate	date	No	Yes	Yes
Views	float	No	Yes	Yes
Likes	float	No	Yes	Yes
Comments	float	No	Yes	Yes
Description	text_general	No	Yes	Yes
videoId	string	No	Yes	Yes
transcript	text_general	No	Yes	Yes
comments	text_general	Yes	Yes	Yes

4 RETRIEVAL

4.1 Retrieval Process and Ideas Explored

In the context of the retrieval process, the research focused on addressing the specified information needs.

4.2 Information needs

4.2.1 Videos about Mysteries.

. For the task of retrieving videos related to discussions about mysteries, the final optimized query developed was:

The query utilizes a flexible search approach, employing the ‘edismax‘ query parser to improve accuracy and result relevance. Specific fields, such as ‘Title‘ and ‘transcript‘, are given boosted importance through the ‘qf‘ (query fields) parameter, enhancing the prioritization of terms within those fields.

```
{
    "query": "[unsolved mystery]",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title^3.0 transcript^1.0
            Description",
        "q.op": "OR"
    }
}
```

Listing 1: Query for Unsolved Mysteries

This query employs the following techniques to enhance search relevance and flexibility:

- **Field Boosts**: Applied to the *Title* field with *^3* and to the *transcript* field with *^1.* These boosts increase the importance of matches in the *Title* and *transcript* fields, making them more relevant than matches in other fields.

4.2.2 Videos about Murders.

. For the task of retrieving videos related to discussions about murders, the final optimized query developed was:

```
{
    "query": "[murder homicide]",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title^3.0 transcript^1.0
            Description",
        "q.op": "OR"
    }
}
```

Listing 2: Query for Murder Homicide

This query employs the following techniques to improve the relevance and flexibility of the search results:

- **Field Boosts**: Applied to the *Title* field with *^3* and to the *transcript* field with *^1*. These boosts increase the importance of matches in the *Title* and *transcript* fields, making them more relevant than matches in other fields.

4.2.3 Videos about Serial Killers from Each State in the USA.

. For the task of retrieving videos discussing serial killers from specific states in the USA, the following optimized query was developed. Note that "California" is used as an example state, and the query would need to be modified to search for videos related to other states:

```
{
    "query": "[(serial AND killer) California CA
        ]",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title^3.0 transcript^1.0
            Description",
        "q.op": "OR"
    }
}
```

Listing 3: Query for Serial Killer California CA

This query employs the following techniques to achieve precision and relevance:

- **Field Boosts**: Applied to the *Title* field with *^3* and to the *transcript* field with *^1*. These boosts increase the importance of matches in the *Title* and *transcript* fields, making them more relevant than matches in other fields.

4.2.4 Videos about Murders in the USA During the 90s.

. To retrieve videos about murders that occurred in the USA during the 1990s, the following optimized query was designed:

```
{
    "query": "[murder homicide 90s [1990 1999] (
        United AND States) America]",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title^3.0 transcript^1.0
            Description",
        "q.op": "OR"
    }
}
```

Listing 4: Query for Murders in the US during the 1990's

This query employs the following techniques to ensure precision and relevance:

- **Field Boosts**: Applied to the *Title* field with *^3* and to the *transcript* field with *^1*. These boosts increase the importance of matches in the *Title* and *transcript* fields, making them more relevant than matches in other fields.
- **Range Searches**: The use of the range syntax *[1990 TO 1999]* captures all numeric references to years in the 1990s.

The query combines filters for geographic location (*USA, United States, America*) and time period (*90s, [1990 TO 1999]*) using logical operators. This ensures that results are specific to murders discussed in the context of the USA during the 1990s.

4.3 Demo

Listings 5 and 6 illustrate the result of running 2 queries in the system:

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 1
  },
  "response": {
    "numFound": 2220,
    "start": 0,
    "maxScore": 12.701204,
    "docs": [
      {
        "Title": ["The Circleville Letters: Creepy
             Unsolved Mystery"],
        "videoId": "pApoKE-YF3c",
        "score": 12.701204
      }
    ]
  }
}
```

Listing 5: Demo for Query for Unsolved Mysteries

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 2
  },
  "response": {
    "numFound": 3593,
    "start": 0,
    "maxScore": 11.038336,
    "docs": [
      {
        "Title": ["Meredith Chapman Murder Case
            Analysis | Infidelity-Motivated
            Homicide"],
        "videoId": "Lr0420kQeck",
        "score": 11.038336
      }
    ]
  }
}
```

Listing 6: Demo for Query for Murder Homicides

5 EVALUATION

5.1 Different setups

To evaluate the performance of the system, two different setups were compared: a basic schema and an advanced schema. The basic schema employed fewer indexed fields and relied on simpler index and query analyzers, while the advanced schema, detailed in the sections above, utilized a more comprehensive and sophisticated approach. The outcomes, presented below, highlight the significant improvements achieved with the advanced schema, demonstrating the importance of adopting a well-designed indexing and querying strategy.

5.2 Process manual evaluation

The retrieval evaluation process involved manually inspecting the results of each query to determine whether the fetched documents met the information need. This meticulous allowed for a detailed assessment of the relevance of each document retrieved by the system. To facilitate this, we created qrels (query relevance judgments) by manually evaluating the relevance of documents to specific queries. These qrels provided a structured way to represent the ground truth for the evaluation process.

Once the qrels were established, they were used to perform various computations and derive performance statistics, described in subsection 5.3. These metrics enabled a quantitative assessment of the system's retrieval effectiveness. While the manual creation of qrels ensured a high-quality evaluation dataset, it was time-consuming and limited the scale of the analysis to a subset of the total results. This limitation highlights a critical area for potential improvement, such as incorporating automated or semi-automated techniques to scale the evaluation process without compromising accuracy.

5.3 Precision Metrics

To assess the performance of the information retrieval system, standard evaluation metrics were utilized, including Precision at 10 (P@10)[5, 16], Mean Average Precision (MAP)[5, 6], and Average Precision (AvP)[5, 12].

5.3.1 Rudimentary Schema - Simple Queries.

Table 6: Performance of Simple Queries in the Rudimentary Schema

Query	P@10	AvP
Unsolved Mysteries	1.0000	0.6788
Serial Killers	0.6000	0.3333
90s Killers	0.9000	0.4264
Murders	0.5000	0.4796

. Table 6 presents the performance of simple queries in the rudimentary schema. The metrics include precision at 10 (P@10) and average precision (AvP).

The Mean Average Precision (**MAP**) was also calculated, amounting to 0.4795.

5.3.2 Advanced Schema - Simple Queries.

Table 7: Performance of Simple Queries in the Advanced Schema

Query	P@10	AvP
Unsolved Mysteries	1.0000	0.6788
Serial Killers	0.6000	0.3333
90s Killers	0.9000	0.4272
Murders	0.5000	0.4794

. Table 7 presents the performance of simple queries in the advanced schema. The metrics include precision at 10 (P@10) and average precision (AvP).

The Mean Average Precision (**MAP**) was also calculated, amounting to 0.4797.

The following subsection compares the Precision-Recall Curves (P-R Curves)[9] of both schemas, for queries of the same information need (murders in the 90's):

5.3.3 Rudimentary Schema. Figure 9 describes the P-R curve of the basic schema, illustrating the trade-off between precision and recall.

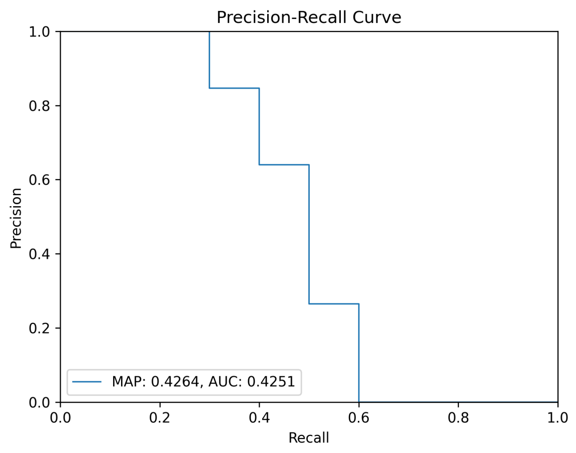


Figure 9: P-R Curve for the rudimentary schema

5.3.4 Comprehensive Schema. Figure 10 describes the P-R curve of the advanced schema, showing a more refined balance between precision and recall compared to the rudimentary approach.

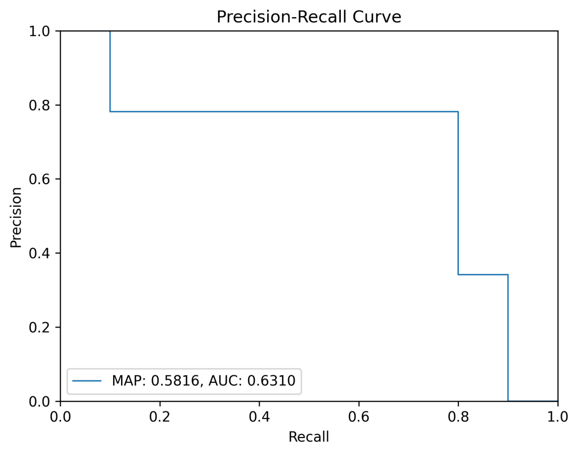


Figure 10: P-R Curve for the comprehensive schema

5.4 Results Discussion

Tables 6 and 7 summarize the performance of simple queries under the rudimentary and advanced schemas, respectively. While both schemas exhibit similar trends in Precision at 10 (P@10) and Average Precision (AvP) for individual queries, the advanced schema marginally outperforms the rudimentary schema in terms of Mean Average Precision (MAP), with values of 0.4797 compared to 0.4795. Although the difference in MAP is minimal, it suggests that the advanced schema slightly improves the relevance ranking of retrieved documents.

Queries such as "Unsolved Mysteries" achieve consistently high precision and average precision across both schemas, indicating that the system handles broad and well-defined information needs effectively. However, more specific queries like "Serial Killers" or "Murders" show a notable drop in precision and average precision, suggesting challenges in retrieving relevant results for less-defined or ambiguous information needs. This observation underscores the importance of query formulation in influencing retrieval performance.

The Precision-Recall (P-R) curves (Figures 9 and 10) provide a visual representation of the trade-off between precision and recall for the rudimentary and advanced schemas. The rudimentary schema's curve exhibits a steep initial drop-off, indicating that while the top-ranked results are highly relevant, the precision diminishes rapidly as more documents are retrieved.

In contrast, the advanced schema demonstrates a more stable precision-recall relationship.

6 INFORMATION RETRIEVAL IMPROVEMENTS

6.1 Query Improvements

In order to improve the performance of the search system, the existing queries were refined in order to use more Solr features and extract more precise results. Listings 7, 8, 9 and 10 describe the upgrades made.

```
{
    "query": "(Title:(unsolved~1 OR mystery*~1)^2
        OR transcript:(unsolved~1 OR mystery*~1))
        AND (Title:\"unsolved mystery\"~5^3 OR
        transcript:\"unsolved mystery\"~5) AND (
        Title:(unsolved AND mystery~10)^1.5 OR
        transcript:(unsolved AND mystery~10)^1.5)
        ",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title transcript",
        "q.op": "AND",
        "sort": "Views desc"
    }
}
```

Listing 7: Improved Query for Unsolved Mysteries

This query now includes the following features to achieve precision and relevance:

- **Term Boosts**
  Example: `"unsolved mystery"~5^3` (boosts the phrase `"unsolved mystery"` within 5 words by 3).
- **Independent Boosts**
  Example: `(Title:(unsolved~1 OR mystery*~1)^2 OR transcript:(unsolved~1 OR mystery*~1))`.
- **Phrase Match w/ Slop**
  Example: `Title:"unsolved mystery"~5`
  (matches `"unsolved mystery"` with up to 5 intervening words).
- **Wildcards/Fuzziness**
  Example: `mystery*~1` (matches terms like `"mysteries"`

with fuzziness).

- **Proximity Searches**
  Example: `Title:(unsolved AND mystery~10)` (matches `unsolved` and `mystery` within 10 terms).

```
{
    "query": "(Title:(murder*~1 OR homicide*~1)^2
        OR transcript:(murder*~1 OR homicide*~1))
        AND ((Title:\"murder\"~5^3)^2.5 OR (
        transcript:\"murder\"~5))",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title transcript",
        "q.op": "AND",
        "sort": "Likes desc"
    }
}
```

Listing 8: Improved Query for Murder Homicide

This query was improved to use the following features:

- **Fields Boosts**
  Example: `(Title:m̈urder¨ 5^3)^2.5` (boosts matches in `Title` by a factor of 2.5).
- **Term Boosts**
  Example: `"murder"~5^3` (boosts the phrase `"murder"` within 5 words by 3).
- **Independent Boosts**
  Example: `(Title:(murder*~1 OR homicide*~1)^2 OR transcript:(murder*~1 OR homicide*~1))`
  (boosts `Title` and `transcript` independently).
- **Phrase Match w/ Slop**
  Example: `Title:"murder"~5` (matches `"murder"` with up to 5 intervening words).
  Example: `transcript:"murder"~5` (same for `transcript`).
- **Wildcards/Fuzziness**
  Example: `murder*~1` (wildcard `*` for matching terms like `"murders"`, combined with fuzziness `~1` for small edit distances).
  Example: `homicide*~1` (same for `homicide`).
- **Proximity Searches**
  Example: `Title:"murder"~5` (matches the exact phrase `"murder"` with terms within 5-word proximity).

```
{
    "query": "(Title:\"serial killer\"~5^3 OR
        Description:\"serial killer\"~5^2 OR
        transcript:\"serial killer\"~5) AND (
        Description:(California~1 OR CA~1)^2 OR
        transcript:(California~1 OR CA~1))",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title Description transcript",
        "q.op": "AND"
    }
}
```

Listing 9: Improved Query for Serial Killer California CA

Improvements to this query include the following:

- **Term Boosts**
  Example: `California~1` (boosts the term `California` with fuzziness of 1).
- **Independent Boosts**
  Example: `(Title:"serial killer"~5^3 OR Description:"serial killer"~5^2 OR transcript:"serial killer"~5)`
  (each clause has its own boost factor).
- **Phrase Match w/ Slop**
  Example: `Title:"serial killer"~5` (matches the phrase `"serial killer"` with up to 5 intervening words).
- **Wildcards/Fuzziness**
  Example: `California~1` (fuzzy matching for terms similar to `California`).
  Example: `CA~1` (fuzzy matching for `CA`).
- **Proximity Searches**
  Example: `Title:"serial killer"~5` (matches the exact phrase `"serial killer"` with terms within 5-word proximity).

```
{
    "query": "(Title:(murder*~5)^2 AND transcript
        :(murder*~5)) AND ((Description:(USA~1 OR
        \"United States\"~1 OR America~1)^3 OR
        transcript:(USA~1 OR \"United States\"~1
        OR America~1)) AND (Description:(90s OR
        [1990 TO 1999])^3 OR transcript:(90s OR
        [1990 TO 1999])))",
    "fields": ["Title", "videoId", "score"],
    "params": {
        "defType": "edismax",
        "rows": 100,
        "qf": "Title Description transcript",
        "q.op": "AND"
    }
}
```

Listing 10: Improved Query for Murders in the US during the 1990's

The techniques used were as follows:

- **Term Boosts**
  Example: `murder*~5^2` (boosts the term `murder*` with fuzziness of 5 and a boost factor of 2).
- **Independent Boosts**
  Example:
  (murder*~5))`(Title:(murder*~5)^2 AND transcript:(murder*~5)) (each clause has its own boost factor).`
- **Phrase Match w/ Slop**
  `Example: Description:(USA~1 OR "United States"~1 OR America~1) (matches the terms USA, "United States", or America with fuzziness of 1).`
- **Wildcards/Fuzziness**

`Example: murder*~5 (wildcard * for matching terms like murders, combined with fuzziness ~5 for larger edit distances).`

- **`Range Searches`**
  `Example: [1990 TO 1999] (matches the range of years from 1990 to 1999).`

6.2 Precision Metrics for Improved Queries

After improving the queries, the precision metrics were recalculated to evaluate the system's evolution. The results were as follows:

Table 8: Performance of Advanced Queries in the Advanced Schema

Query	P@10	AvP
Unsolved Mysteries	1.0000	0.7955
Serial Killers	1.0000	0.6119
90s Killers	0.7000	0.5816
Murders	1.0000	0.7131

Table 8 presents the performance of advanced queries in the comprehensive schema. The metrics include precision at 10 (P@10) and average precision (AvP).

The Mean Average Precision (**MAP**) was also calculated, amounting to 0.6755.

Figure 11 illustrates what a P-R curve for the murders in the 90's query looks like:

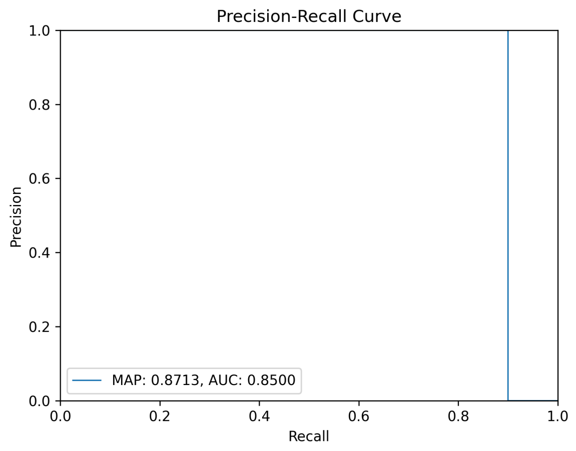


Figure 11: P-R Curve for the advanced queries in the comprehensive schema

The introduction of improved queries resulted in a noticeable enhancement in the retrieval performance, as shown in Table 8. The metrics demonstrate higher Precision at 10 (P@10) and Average Precision (AvP) values compared to the results obtained with the simple queries in both the rudimentary and advanced schemas. The calculated Mean Average Precision (MAP) of 0.6755 further underscores the improvements, representing a significant increase from the MAP values of 0.4795 and 0.4797 seen previously.

Comparison of Results Across Different Scenarios:

1. Precision at 10 (P@10)

For the advanced queries in the comprehensive schema, P@10 reaches perfect precision (1.0000) for three out of four queries, with only a slight reduction for "90s Killers" (0.7000). In contrast, the simple queries under both rudimentary and advanced schemas showed more variability, with P@10 values ranging between 0.5000 and 1.0000. This indicates that the improved queries significantly enhanced the relevance of top-ranked documents, particularly for previously ambiguous or specific queries like "Serial Killers" and "Murders," which now consistently achieve P@10 of 1.0000.

2. Average Precision (AvP)

The advanced queries show marked improvements in AvP. For example:

- "Unsolved Mysteries" improved from 0.6788 (simple queries) to 0.7955.
- "Serial Killers" rose from 0.3333 to 0.6119.
- "90s Killers" increased slightly from 0.4264 to 0.5816.
- "Murders" improved from approximately 0.4796 to 0.7131.

These gains reflect the system's ability to rank relevant documents more effectively when using improved query formulations.

3. Precision-Recall (P-R) Curves

Figure 11 illustrates the P-R curve for the "90's Killers" query under the advanced queries scenario. Compared to the curves for simple queries in the rudimentary and comprehensive schemas (Figures 9 and 10), this curve starts with higher precision and recall, indicating a more consistent retrieval of relevant documents.

The jump in performance showcases the importance of using Solr's built-in search improvement features.

6.3 Semantic Search

The all-MiniLM-L6-v2 model[13] from the sentence-transformers[14] library in Python was used to generate embeddings for the documents. The fields used were Channel, Title, Description and Transcript.

The model creates a new JSON document that includes a new vector field. This vector field translates the combined textual content of the specified fields into a high-dimensional numerical representation. These embeddings capture semantic meaning, enabling tasks like clustering, similarity comparison, or downstream machine learning applications.

The schema was then updated to incorporate a new field of the type DenseVectorField. Cosine was the chosen similarity function and hnsw the selected algorithm for the K-Nearest Neighbors search. The final result is visible in table 9.

All fields are stored and indexed.

Dense vector embeddings allowed for Semantic Querying.

The improved queries described in the last subsection were reused in order to analyze the true impact of the semantic embeddings. The results were as follows:

Table 9: Schema Field Definitions after Semantic Analysis

Field Name	Field Type	Multi-Valued
Channel	string	No
Title	text_general	No
PublishedDate	date	No
Views	float	No
Likes	float	No
Comments	float	No
Description	text_general	No
videoId	string	No
transcript	text_general	No
comments	text_general	Yes
vector	DenseVectorField	Yes

Table 10: Performance of Advanced Queries in the Advanced Schema with Semantic Search

Query	P@10	AvP
Unsolved Mysteries	0.3000	0.2750
Serial Killers	0.4000	0.4516
90s Killers	0.6000	0.6272
Murders	0.4000	0.3416

Table 10 presents the performance of advanced queries in the comprehensive schema with Semantic Search. The metrics include precision at 10 (P@10) and average precision (AvP).

The Mean Average Precision (**MAP**) was also calculated, amounting to 0.4189.

Figure 12 illustrates what a P-R curve for the 90's killers query looks like:

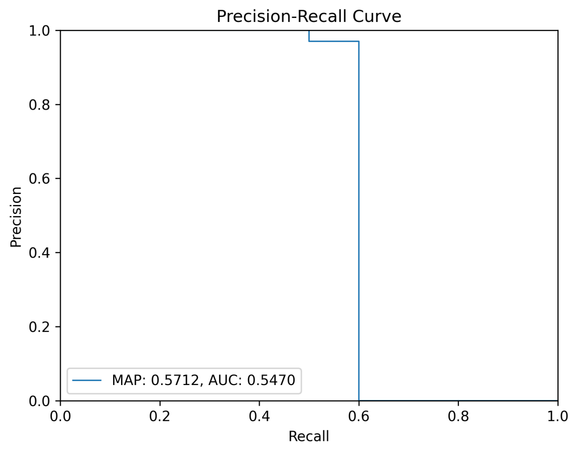


Figure 12: P-R Curve for the advanced queries in the comprehensive schema

Unlike traditional keyword-based search, semantic search leverages dense vector embeddings, which capture the underlying meanings of words and phrases. While this approach improves the ability to retrieve contextually relevant documents, it introduces a trade-off between enhanced contextual understanding and the potential dilution of exact matches, as reflected in the metrics.

Comparison of Results Across Different Scenarios:

1. Precision at 10 (P@10)

Semantic search resulted in a significant drop in P@10 for most queries compared to keyword-based advanced search, as shown in Table 10. For instance:

- Queries like “Serial Killers” and “Unsolved Mysteries,” which previously achieved perfect precision (1.0000), now show much lower precision of 0.4000 and 0.3000, respectively. This is because semantic matching often retrieves contextually similar documents that lack exact term matches, reducing relevance at the top ranks.
- For the “90s Killers” query, P@10 remains relatively higher at 0.6000, highlighting semantic search's strength in broader contextual alignment but still trailing the performance of traditional search.

2. Average Precision (AvP)

Semantic search also resulted in a notable decline in AvP:

- Queries such as “Unsolved Mysteries” dropped from 0.7955 to 0.2750, and “Murders” fell from 0.7131 to 0.3416, reflecting a significant loss of ranking precision.
- Even for “Serial Killers” and “90s Killers,” AvP values of 0.4516 and 0.6272, respectively, are far below the corresponding values of 0.6119 and 0.5816 under keyword-based search.

The drop in AvP highlights the impact of misaligned semantic embeddings, where documents deemed contextually relevant by the model may not directly address user intent.

3. Precision-Recall (P-R) Curves

Figure 12 shows the P-R curve for the “90's Killers” query under semantic search. A Key differences compared to traditional search is Lower early precision, where top-ranked documents are less relevant.

. *Analysis of the Decline and Opportunities for Improvement*

The Mean Average Precision (**MAP**) dropped significantly from 0.6755 to 0.4189, underscoring the limitations of semantic search in its current implementation. This decline can be attributed to several factors:

- **Embedding Granularity:** Dense vector embeddings prioritize semantic relationships but fail to capture exact matches or nuanced query terms, leading to mismatches at top ranks.
- **Field Combination:** Combining fields such as Title, Description, and Transcript into a single embedding dilutes the weight of specific terms critical to the query.

To improve semantic search performance:

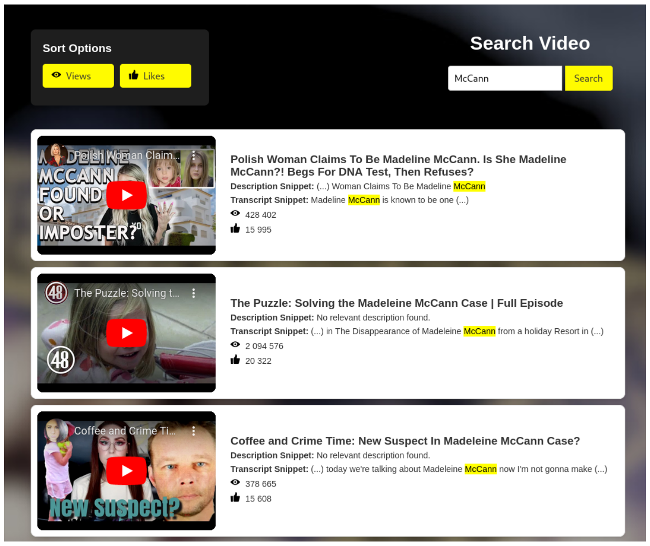
(1) **Field Weighting:** Introduce field-specific weights in embeddings to prioritize critical fields like Title and Description.
(2) **Hybrid Approaches:** Combine semantic embeddings with traditional keyword-based search to balance contextual relevance with term specificity.
(3) **Algorithm Optimization:** Explore alternative ranking algorithms or adjust HNSW parameters to better reflect query relevance.

Despite the current performance drop, semantic search remains a promising approach that, with refinements, can outperform traditional search in both precision and recall.

6.4 User Interface

In order to improve the user experience, an interface was developed in HTML[17] + JS[8] + CSS[18]. It features options to sort results by views and likes, as well as the option to reproduce videos directly and has the ability to search for and highlight snippets in fields like Description and Transcript.

Figure 13 illustrates what the frontend looks like.



Polish Woman Claims To Be Madeline McCann. Is She Madeline McCann?! Begs For DNA Test, Then Refuses?
Description Snippet: (...) Woman Claims To Be Madeline McCann
Transcript Snippet: Madeline McCann is known to be one (...)
428 402
15 995

The Puzzle: Solving the Madeleine McCann Case | Full Episode
Description Snippet: No relevant description found.
Transcript Snippet: (...) in The Disappearance of Madeleine McCann from a holiday Resort in (...)
2 094 576
20 322

Coffee and Crime Time: New Suspect In Madeleine McCann Case?
Description Snippet: No relevant description found.
Transcript Snippet: (...) today we're talking about Madeleine McCann now I'm not gonna make (...)
378 665
15 608

Figure 13: True Crime YouTube Video Search Engine Frontend

6.5 Conclusion

In conclusion, this project showed how important it is to get schema design and query optimization right to improve search performance. From the early stages of processing raw data to building a functional interface where users can make queries, every step came with its own challenges and lessons. It highlighted just how much work goes into creating a search system that balances speed, accuracy, and usability.

This system is a solid foundation for delivering relevant results and handling a wide range of data and user needs. While semantic analysis played a role, the bigger picture involved optimizing the entire system, from algorithms and data structures to the way users interact with it.

There's a lot of room to make this even better in the future, like adding new data sources, refining how the data is processed, introducing more signals to improve accuracy or improving semantic search. With these improvements, the system could grow even more flexible and reliable, ready to handle evolving demands.

REFERENCES

[1] [n. d.]. Apache Solr Documentation — solr.apache.org. https://solr.apache.org/guide/. [Accessed 20-11-2024].
[2] [n. d.]. MongoDB Documentation — mongodb.com. https://www.mongodb.com/docs/. [Accessed 16-10-2024].
[3] [n. d.]. True Crime Channel Statistics — kaggle.com. https://www.kaggle.com/datasets/mhapich/true-crime-channel-statistics. [Accessed 09-10-2024].
[4] [n. d.]. youtube-transcript-api — pypi.org. https://pypi.org/project/youtube-transcript-api/. [Accessed 16-10-2024].
[5] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 1999. *Modern Information Retrieval*. ACM Press / Addison-Wesley. http://www.ischool.berkeley.edu/~hearst/irbook/glossary.html
[6] Chris Buckley and Ellen M. Voorhees. 2000. Evaluating evaluation measure stability. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Athens, Greece) *(SIGIR '00)*. Association for Computing Machinery, New York, NY, USA, 33–40. https://doi.org/10.1145/345508.345543
[7] Alexis M Durham, H.Preston Elrod, and Patrick T Kinkade. 1995. Images of crime and justice: Murder and the "true crime" genre. *Journal of Criminal Justice* 23, 2 (1995), 143–152. https://doi.org/10.1016/0047-2352(95)00002-8
[8] ECMA International. 2023. *ECMAScript Language Specification*. https://www.ecma-international.org/publications-and-standards/standards/ecma-262/ JavaScript Standard Specification.
[9] Peter Flach and Meelis Kull. 2015. Precision-Recall-Gain Curves: PR Analysis Done Right. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2015/file/33e8075e9970de0cfea955afd4644bb2-Paper.pdf
[10] Daniel Glez-Peña, Anália Lourenço, Hugo López-Fernández, Miguel Reboiro-Jato, and Florentino Fdez-Riverola. 2013. Web scraping technologies in an API world. *Briefings in Bioinformatics* 15, 5 (04 2013), 788–797. https://doi.org/10.1093/bib/bbt026 arXiv:https://academic.oup.com/bib/article-pdf/15/5/788/17488715/bbt026.pdf
[11] Kaggle. [n. d.]. Kaggle - Data Science and Machine Learning Platform. https://www.kaggle.com. Accessed: 2024-10-12.
[12] Stefano Mizzaro. 1997. Relevance: The Whole History. *J. Am. Soc. Inf. Sci.* 48 (1997), 810–832. https://api.semanticscholar.org/CorpusID:12793190
[13] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. https://www.sbert.net. Version used: all-MiniLM-L6-v2 from the Sentence-Transformers library in Python.
[14] Nils Reimers and Iryna Gurevych. 2020. *Sentence-Transformers*. https://github.com/UKPLab/sentence-transformers
[15] Guido Van Rossum and Fred L. Drake. 2009. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.
[16] Ellen M. Voorhees. 2001. Evaluation by highly relevant documents. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New Orleans, Louisiana, USA) *(SIGIR '01)*. Association for Computing Machinery, New York, NY, USA, 74–82. https://doi.org/10.1145/383952.383963
[17] World Wide Web Consortium (W3C). 2017. *HTML5: A vocabulary and associated APIs for HTML and XHTML*. https://www.w3.org/TR/html5/ HTML Language Specification.
[18] World Wide Web Consortium (W3C). 2018. *CSS: Cascading Style Sheets*. https://www.w3.org/Style/CSS/ CSS Language Specification.
[19] YouTube. n.d.. YouTube Data API (v3). https://developers.google.com/youtube/v3. Accessed: 2024-10-12.

Received February 4, 2025