YouTube Analysis on Hadoop Ecosystem and Performance Comparison

Author : Shravya D’souza

Date : 11/11/24

ABSTRACT

The rapid growth of online platforms like YouTube has led to substantial data processing challenges, necessitating scalable, efficient solutions. This project, *YouTube Analysis on Hadoop Ecosystem and Performance Comparison*, uses the Hadoop ecosystem to process and analyze a large-scale YouTube dataset, providing insights into views, ratings, and trends. Leveraging HDFS, HBase, Apache Spark, Hive, Pig, and MongoDB, the project demonstrates efficient data handling, storage, and SQL-like querying for unstructured metadata. By comparing query execution times across Spark, Hive, Pig, HBase, and MongoDB, we identify the most effective tools for various data operations, showcasing the scalability, efficiency, and flexibility of the Hadoop ecosystem and MongoDB in addressing big data challenges.

LIST OF FIGURES

TABLE OF CONTENTS

PROBLEM STATEMENT

With the exponential growth of digital content on platforms like YouTube, managing and analyzing vast amounts of video data has become increasingly challenging. YouTube generates millions of records daily, including information on video views, ratings, upload details, and user interactions. Traditional data processing methods are not equipped to handle such large-scale, complex datasets efficiently. To extract meaningful insights from this massive data, a robust and scalable solution is required. The primary goal of this project, *YouTube Analysis on Hadoop Ecosystem and Performance Comparison*, is to address these challenges by leveraging distributed technologies such as Hadoop, Apache Spark, HDFS, HBase, Hive, Pig, and MongoDB to perform large-scale data processing and analysis.

This project aims to analyze millions of YouTube video records to derive key insights, such as identifying the top-rated videos, computing category-wise average video lengths, and analyzing viewership patterns. Each technology in the Hadoop ecosystem serves a unique purpose in achieving these goals: HDFS provides the foundation for distributed data storage, Spark enables high-speed processing of large datasets, Hive and Pig facilitate querying and data transformations, and HBase supports fast NoSQL-style data retrieval for efficient handling of unstructured video metadata. Additionally, MongoDB offers an alternative NoSQL solution, enabling flexible schema management and providing a basis for comparative performance analysis. Together, these technologies offer a powerful framework for processing massive datasets with high speed and accuracy, making it possible to derive actionable insights from the data.

Furthermore, this project incorporates a performance comparison of the execution times for each query across different technologies within the Hadoop ecosystem, including Hive, Pig, Spark, HBase, and MongoDB. This comparison aims to identify the most efficient tools for different types of queries, providing insights into optimizing query performance for large datasets. By understanding the execution times, this project offers a practical guide to selecting the right technology based on the complexity of the queries and dataset size, thereby maximizing performance and minimizing resource consumption. These findings will be instrumental in improving data analysis processes in large-scale video platforms like YouTube, enabling more informed decision-making for content creators, advertisers, and platform managers.

INTRODUCTION

In recent years, the exponential growth of digital content on platforms like YouTube has transformed how people consume and interact with video media. With millions of videos uploaded daily, the sheer volume of data generated poses significant challenges for analysis and insight extraction. Traditional relational databases often fall short when it comes to handling such large-scale data due to limitations in processing speed, storage capacity, and flexibility. This necessitates the adoption of advanced data processing frameworks that can efficiently manage and analyze big data, making it crucial for organizations to harness the power of distributed computing technologies.

The Hadoop ecosystem, comprising tools such as Apache Spark, HDFS, Hive, Pig, and HBase, provides an ideal solution for processing large datasets. HDFS enables distributed storage, allowing organizations to store vast amounts of video data across multiple nodes. Apache Spark enhances this capability by facilitating high-speed data processing through in-memory computing, significantly reducing query execution times. Hive and Pig offer powerful querying capabilities, enabling complex analyses using familiar SQL-like syntax or flexible scripting approaches. Meanwhile, HBase provides a NoSQL storage solution for unstructured data, ensuring rapid access to video metadata. In addition to these Hadoop tools, MongoDB is included in this project as a NoSQL alternative, providing flexibility in schema management and adding a comparative perspective to the performance analysis.

This project, titled *YouTube Analysis on Hadoop Ecosystem and Performance Comparison*, aims to leverage these technologies to conduct a comprehensive analysis of YouTube video data. The goal is to derive actionable insights related to video performance, such as identifying top-rated

videos, analyzing viewer engagement, and understanding content trends. Furthermore, the project includes a performance comparison of query execution times across Spark, Hive, Pig, HBase, and MongoDB, offering valuable insights into the strengths and limitations of each technology in handling different types of queries. By exploring these aspects, this project highlights the potential of the Hadoop ecosystem and MongoDB to tackle big data challenges, ultimately enabling data-driven decision-making for content creators, advertisers, and platform managers in the evolving landscape of online video consumption.

PURPOSE

The primary purpose of the *YouTube Analysis on Hadoop Ecosystem and Performance Comparison* project is to efficiently process and analyze large-scale video data from YouTube using a distributed computing framework. By leveraging technologies such as Apache Spark, HDFS, HBase, Hive, Pig, and MongoDB, the project aims to extract meaningful insights from millions of video records, providing valuable information about viewership trends, video performance, and content engagement. This project offers an efficient solution for handling vast datasets that traditional data processing methods struggle to manage, thereby enabling faster and more accurate data-driven decisions.

Additionally, the project serves to demonstrate the strengths and limitations of various tools within the Hadoop ecosystem, particularly regarding query execution time. By comparing the performance of Spark, Hive, Pig, HBase, and MongoDB for different queries, the project highlights which tools are best suited for specific types of data processing tasks. This analysis aims to optimize the use of big data technologies, helping stakeholders in content platforms like YouTube make informed choices about which tools to use based on the complexity of their data and analysis requirements.

Ultimately, the project seeks to showcase how distributed computing technologies can transform the management and analysis of large datasets. The insights generated from this analysis will not only benefit content creators and advertisers in optimizing their strategies but also contribute to enhancing the performance of big data analytics on video platforms.

SCOPE

1. **Data Cleaning and Transformation**: Perform data cleaning and transformation tasks using Apache Spark to prepare the YouTube dataset for analysis, ensuring accuracy and consistency.
2. **Distributed Data Storage with HDFS**: Use the Hadoop Distributed File System (HDFS) to store large YouTube datasets across multiple nodes, providing scalability and fault tolerance.
3. **High-Speed Processing with Apache Spark**: Implement fast, distributed, in-memory data processing with Apache Spark, enabling efficient querying and analysis of large-scale datasets.
4. **Query Execution with Multiple Technologies**: Run analytical queries across multiple Hadoop ecosystem components, including Apache Spark, Pig, Hive, HBase, and MongoDB, to perform various computations on the YouTube dataset.
5. **SQL-Like Querying with Hive**: Utilize Apache Hive for SQL-like queries on structured YouTube data, such as calculating average ratings and views across categories.
6. **Data Transformation with Apache Pig**: Use Apache Pig to transform and process video data, focusing on insights such as total comments and average video length by category.
7. **NoSQL Data Storage with HBase**: Leverage HBase for storing and retrieving unstructured data like video metrics and metadata, ensuring fast access for querying and real-time analytics.
8. **Query Performance Comparison**: Conduct a performance comparison of query execution times across Apache Spark, Pig, Hive, HBase, and MongoDB to identify the most efficient tools for specific data analysis tasks.

9. **Insight Generation**: Derive insights, such as top-rated videos, most-viewed categories, and video length analysis, to understand trends in content performance.
10. **Scalability and Flexibility**: Design the system to be scalable and adaptable, capable of handling future data growth on platforms like YouTube, ensuring continued efficiency as data volume increases.
11. **Data-Driven Decision Making**: Provide actionable insights to stakeholders, including content creators and advertisers, helping them optimize strategies based on viewer engagement and performance metrics.
12. **Real-Time Data Analytics**: Enable near real-time analysis of video data using HBase and Spark, allowing for timely trend reporting and decision-making in video consumption.

OBJECTIVE

1. **Efficient Data Processing**: Leverage the Hadoop ecosystem—including Apache Spark, Pig, Hive, HBase, and MongoDB—to process and analyze large-scale YouTube datasets, ensuring high-speed and accurate analysis.
2. **Performance Optimization**: Compare query execution times across Apache Spark, Pig, Hive, HBase, and MongoDB to determine the most efficient tools for various data processing tasks, optimizing overall performance.
3. **Actionable Insights**: Derive key insights such as top-rated videos, most-viewed categories, and video length distribution, providing valuable information for content creators, advertisers, and platform managers.
4. **Data Cleaning and Transformation**: Implement data cleaning and transformation processes using Apache Spark to ensure the dataset is consistent, accurate, and ready for analysis.
5. **Scalability and Fault Tolerance**: Build a scalable and fault-tolerant system using HDFS and other Hadoop components, capable of handling the increasing volume of video data as YouTube content grows.
6. **Real-Time Data Analysis**: Enable near real-time data retrieval and analysis using HBase, allowing for faster insights and decision-making on YouTube data trends.
7. **Comprehensive Ecosystem Utilization**: Demonstrate the integration of multiple Hadoop technologies—Spark, Hive, Pig, HBase, and MongoDB—showcasing the advantages of using a distributed computing framework for large-scale data analysis.

LITERATURE SURVEY

The increasing volume of data generated by digital platforms has necessitated the development of advanced technologies for effective data management and analysis. In particular, the YouTube platform serves as a prime example of big data challenges, where millions of videos are uploaded and viewed daily, resulting in vast amounts of metadata and user interaction data. Various studies have explored the complexities of managing and analyzing such large datasets, highlighting the limitations of traditional database systems in handling big data effectively. As a result, researchers have increasingly turned to distributed computing frameworks, particularly the Hadoop ecosystem, which allows for scalable data processing and storage.

Apache Hadoop, as introduced by Dean and Ghemawat (2004), revolutionized big data processing by utilizing a distributed architecture that enables data to be stored and processed across multiple nodes. HDFS (Hadoop Distributed File System) plays a critical role in this architecture, providing a robust storage solution that is fault-tolerant and capable of handling large datasets. The scalability of HDFS allows organizations to expand their data storage capacity seamlessly as data volumes grow. Additionally, Hadoop's MapReduce programming model facilitates efficient data processing by distributing tasks across a cluster, significantly improving performance and throughput in big data scenarios.

The integration of Apache Spark into the Hadoop ecosystem has further enhanced data processing capabilities. Spark's in-memory computing architecture allows for faster data processing compared to traditional MapReduce, making it ideal for iterative algorithms and interactive data analysis (Zaharia et al., 2010). Studies have shown that Spark can outperform MapReduce by an order of magnitude for certain workloads, particularly those requiring multiple

passes over the data. This performance boost is especially valuable for analyzing large datasets, such as those found on YouTube, where real-time insights are essential for content optimization and user engagement.

In addition to data processing, querying capabilities are a crucial aspect of big data analysis. Apache Hive and Apache Pig offer distinct approaches to querying data stored in HDFS. Hive provides a SQL-like interface for data analysis, making it accessible to users familiar with traditional database systems. It abstracts the complexities of Hadoop's underlying architecture, allowing analysts to focus on extracting insights without needing extensive programming knowledge (Thompson et al., 2016). On the other hand, Pig's procedural language offers greater flexibility for data transformation tasks, enabling users to express complex data workflows more succinctly. Both tools have proven effective in enabling organizations to conduct analyses on large datasets, such as YouTube's extensive video catalog.

Moreover, the incorporation of NoSQL databases like HBase into the Hadoop ecosystem addresses the need for real-time data access and storage of unstructured data. HBase enables efficient read and write operations, making it suitable for scenarios requiring fast access to large volumes of data (George, 2011). This capability is particularly valuable for managing YouTube metadata and user interaction logs, which are inherently unstructured and continuously growing. Research has shown that the combination of HDFS for storage and HBase for real-time access creates a powerful architecture for handling big data, facilitating seamless querying and analysis across various dimensions of the dataset.

Beyond Hadoop, MongoDB, a widely adopted NoSQL database, offers schema flexibility and high performance for managing semi-structured data. MongoDB's document-oriented storage

model and ease of use make it suitable for applications requiring rapid development and flexible schema management, particularly with data that does not fit neatly into a structured schema. In recent studies, MongoDB has demonstrated competitive performance in data retrieval and aggregation tasks, providing a valuable point of comparison against Hadoop-based tools for handling semi-structured YouTube data.

In conclusion, the literature indicates a significant shift towards utilizing the Hadoop ecosystem for big data analysis, particularly in dynamic environments like YouTube. The combination of HDFS, Apache Spark, Hive, Pig, HBase, and MongoDB offers a comprehensive solution for managing, processing, and analyzing large datasets effectively. As organizations continue to grapple with the challenges posed by big data, the adoption of these technologies will likely play a crucial role in enabling data-driven decision-making and optimizing content strategies in the rapidly evolving digital landscape. This project will build on these foundational concepts to explore the practical application of the Hadoop ecosystem in analyzing YouTube data, including performance comparisons across various technologies.

SYSTEM ARCHITECTURE

System architecture is a conceptual model that defines the structure, behavior, and views of a system. It serves as a blueprint for the system and guides the development process. System architecture is essential in designing and implementing systems, providing a comprehensive framework that allows various components to work together effectively.

This architecture outlines the system components and data flow for a YouTube data analysis project leveraging the Hadoop ecosystem. The project aims to efficiently handle large datasets with scalable data storage, processing, and visualization capabilities.

1. Data Ingestion Layer

- **Data Source**: The dataset is sourced from YouTube Data.
- **Storage in HDFS**: The dataset is ingested into the Hadoop Distributed File System (HDFS) for distributed, fault-tolerant storage.
- **Storage in HBase**: HBase serves as a NoSQL database layer on top of HDFS, optimized for real-time read/write access to large datasets. It is ideal for storing structured information required for random access.

2. Processing Layer

- **Apache Spark**
    - **Dataset Cleaning and Transformation**: Spark performs data preprocessing tasks such as filtering, removing duplicates, and transforming data as needed for analysis.

- **Data Analysis**: Spark executes complex queries (e.g., identifying top viewed videos, calculating average ratings per category) in parallel, enabling faster data analysis across large data volumes.

3. Query and Analysis Layer

- **Hive**: SQL-like queries are executed on structured data within HDFS. Hive enables easy querying of large datasets without complex coding, allowing users to perform operations like calculating average ratings and view counts across categories.
- **Pig**: Pig provides a high-level scripting language suitable for complex, multi-step transformations. It is particularly useful for ETL (Extract, Transform, Load) operations and data aggregation tasks, such as identifying the top 5 longest videos or the most active uploaders.
- **Apache Spark**: Spark is used for high-speed data processing, supporting both batch and real-time analysis. Spark allows for the execution of complex queries and transformations, with tasks like filtering, data transformation, and iterative computations for identifying trends and insights within large datasets.
- **HBase**: HBase is employed as a NoSQL database for storing and retrieving unstructured or semi-structured data, allowing for efficient, real-time querying. This makes it ideal for scenarios that require quick access to specific data points, such as metadata and user interactions in large datasets.
- **MongoDB**: Used as an additional NoSQL database, MongoDB allows for flexible schema management, making it particularly useful for semi-structured data. MongoDB's performance is compared against Hadoop ecosystem components to evaluate its

efficiency in handling specific types of queries, especially on datasets requiring flexible schema structures.

4. Coordination Layer

- **Zookeeper**: Zookeeper ensures coordination and synchronization across distributed components, helping maintain consistency and manage configurations for components like HBase and Spark.

5. Workflow Automation Layer (Future Scope)

- **Oozie**: In future scope, Oozie can be incorporated to manage and automate workflows between Spark, Hive, and Pig jobs, supporting periodic data analysis and scheduled data ingestion tasks.

6. Visualization Layer

- **Data Export**: Spark outputs processed data (e.g., query results) in formats like Parquet or CSV for easy access in visualization tools.
- **Jupyter Notebook**: Jupyter Notebook is utilized for data visualization with Spark in Python. Various Python libraries (e.g., Matplotlib, Seaborn, Plotly) are employed to create visual representations of analysis results, such as bar charts, line graphs, and correlation matrices.
- **Zeppelin (Future Scope)**: As a future enhancement, Zeppelin can be integrated for interactive visualizations directly on Hadoop components, allowing dynamic data exploration and presentation of insights.

Data Flow Summary

1. **Raw Data** is ingested and stored in HDFS and HBase.
2. **Data Cleaning and Transformation** are conducted in Spark, with processed data stored back in HDFS and HBase.
3. **Query Execution** occurs through Hive and Pig, with additional processing handled by Spark.
4. **Visualization** is performed in Jupyter Notebook with Spark and Python, with Zeppelin as a potential future addition for direct Hadoop-based visualization.
5. **Comparative Analysis**: MongoDB is used alongside other Hadoop tools to compare query execution times, providing insights into optimal tools for different query types.

System architecture is critical for designing complex systems, ensuring that all components work together seamlessly. This high-level overview guides the development process, helps manage complexity, and aligns the system with project goals.

This architecture provides a robust, scalable setup leveraging Hadoop's distributed storage and Spark's efficient data processing, with flexibility for future enhancements in workflow automation and visualization capabilities.

DATASET PREPARATION

The dataset preparation process is essential to ensure that the YouTube dataset is organized, cleaned, and formatted for efficient analysis across different platforms (MongoDB, Hive, Pig, and HBase). This project involves using a comprehensive YouTube dataset from 2007, consisting of 35 zip files, each containing multiple .txt files. The data includes various attributes such as video ID, uploader, video age, category, length, views, ratings, comments, and related videos. The goal is to preprocess this dataset for streamlined querying and analysis.

1. Dataset Overview

- **Source**: The dataset comprises 35 zip files, each representing a specific portion of the YouTube data for 2007.
- **File Types**: Each zip file contains .txt files with the following structure:
    - 0.txt, 1.txt, 2.txt, etc., containing actual video data with columns such as video ID, uploader, age, category, length, views, ratings, comments, and related videos.
    - log.txt, containing metadata about the data extraction process (e.g., timestamps, number of videos processed).
- **Attributes**:
    - **Video ID**: Unique identifier for each video.
    - **Uploader**: Username of the uploader.
    - **Age**: Days since video upload relative to YouTube's start date (Feb 15, 2007).
    - **Category**: Video category as chosen by the uploader.
    - **Length**: Video length in seconds.

- **Views, Rating, Ratings Count, Comments Count**: Metrics for viewer engagement.
- **Related Videos**: List of related video IDs (up to 20 per video).

2. Data Extraction and Initial Processing

I. Unzipping and Organizing Files

1. **Unzipping**: Each zip file was extracted to a specified directory. After extraction, the zip files were removed to save storage.
2. **Folder Structure**: The unzipped contents were stored in a temporary directory, ensuring all .txt files were accessible in one location for further processing.
3. **File Consolidation**: Since each folder contained multiple files with identical names (e.g., 0.txt, 1.txt), these files were renamed with their folder prefix (e.g., 0001_0.txt, 0002_1.txt) to avoid conflicts during merging.

II. Merging Data Files

1. **Combining Files**: All .txt files (excluding log.txt) were merged into a single file for each .txt group across folders. For example, all 0.txt files across the 35 folders were combined into one, followed by all 1.txt files, and so on.
2. **Creating a Unified Dataset**: Each combined .txt file was then appended into one consolidated dataset, saved as a .csv file.
3. **Delimiter Handling**: The data columns were separated by tabs, so the final .csv was created with a tab delimiter for compatibility.

3. Data Cleaning and Transformation

I. Data Formatting

1. **Schema Definition**: To facilitate consistency across platforms, the following schema was adopted:
    - Video ID: String
    - Uploader: String
    - Age: Integer
    - Category: String
    - Length: Integer
    - Views: Integer
    - Rating: Float
    - Ratings Count: Integer
    - Comments Count: Integer
    - Related Videos: Array of Strings
2. **Column Adjustments**:
    - **Related Videos**: Initially stored as a comma-separated string, this field was transformed into an array format compatible with MongoDB and HBase. For compatibility with CSV (for Hive and Pig), Related Videos was converted to a single comma-separated string within the .csv.

II. Handling Missing Values and Data Types

1. **Missing Data**: Missing or empty values in Related Videos were filled with an empty list ([]) for MongoDB and HBase compatibility, and an empty string for Hive and Pig.

2. **Data Type Consistency**: Ensured integer and float data types for metrics (e.g., Views, Rating, Comments Count) to facilitate numeric operations in analytical queries.

4. Loading and Integration

I. MongoDB Integration

The prepared .csv file was imported into MongoDB using a Python script, where the following transformations were applied:

- Related Videos were stored as an array of strings to allow for efficient querying.
- Document structure was designed to be easily accessible for predefined queries (e.g., Top 10 most viewed videos, average ratings by category).

II. Hive Integration

1. **Loading to HDFS**: The final .csv file was uploaded to HDFS to serve as the input source for Hive.
2. **Table Creation**: A Hive table with the specified schema was created, with Related Videos stored as a single string.
3. **Query Preparation**: The dataset was structured for aggregation queries like category-based analysis and view-count distribution.

III. Pig Integration

1. **Data Loading**: The .csv file in HDFS was loaded into Pig, where columns were accessed via the specified schema.

2. **Transformations**: Custom transformations were applied to parse Related Videos as a string for text-based analyses and group aggregations.

IV. HBase Integration

1. **Row Key**: The Video ID was set as the row key for efficient lookups.
2. **Column Families**: The remaining columns were organized under column families (metrics for numeric data and details for categorical attributes).
3. **Data Loading**: A bulk loader was used to ingest the data into HBase, with appropriate mappings for each column family.

Summary of Dataset Preparation

The dataset preparation involved multiple steps to consolidate, clean, and format the YouTube data for effective multi-platform analysis. Key takeaways include:

- **Consistency**: Ensuring uniform data types and handling missing values for compatibility.
- **Storage Optimization**: Organizing data in HDFS for access by Hive, Pig, and HBase while retaining flexibility for MongoDB.
- **Transformation**: Modifying the Related Videos column based on platform requirements, enhancing query performance and data accessibility.

METHODOLOGY

1. Environment Setup and Dependencies

To analyze and process large-scale data on the Hadoop ecosystem, multiple libraries were installed, including pyspark for Spark functionalities, dask for data manipulation, and happybase for HBase integration. These libraries enable efficient handling, storage, and querying of massive datasets within the Hadoop ecosystem.

2. Data Ingestion and Preprocessing

- **Unzipping Data Files**: Automated extraction of 35 zip files containing YouTube data was implemented to streamline access. Each folder within the dataset was unzipped and verified for completeness to avoid data loss or redundancy.
- **File Renaming and Organization**: Following extraction, .txt files in each folder were renamed based on their parent folder's name to maintain a structured format. This process helped in organizing and storing files systematically in a temporary directory, omitting unwanted files such as log.txt.

3. Data Merging

A script was developed to iterate through all text files in the temporary directory, loading them into a Spark DataFrame. This DataFrame consolidated the entire dataset, providing a unified structure with all records, enabling further analysis and transformation within a single environment.

4. Data Transformation

- **Column Definition and Parsing**: Using Spark SQL, each column within the DataFrame was explicitly defined based on the data structure: Video ID, Uploader, Age, Category, Length, Views, Rating, Ratings Count, Comments Count, and Related Videos.
- **Handling Related Videos**: The Related Videos column, which contained multiple IDs, was transformed into a list format to enable easy querying and manipulation.
- **Schema Validation**: Before proceeding with analysis, the schema was validated to ensure column types matched expected formats (e.g., int, string, double). This validation step guaranteed that data inconsistencies were handled prior to analysis.

5. Data Storage on HDFS

After transformation, the dataset was saved onto the Hadoop Distributed File System (HDFS) in two formats: as a tab-separated .txt file and as a standard .csv file, facilitating access for various Hadoop components like Hive, Pig, and HBase.

6. Query Execution Using MongoDB (for additional insights)

Although MongoDB is not central to the primary Hadoop ecosystem analysis, it was used for running specific queries, leveraging its capabilities to store and query document-based data structures. A Python script facilitated direct MongoDB queries to analyze trends and store intermediate results, which complemented the primary Hadoop analysis.

7. Query Analysis on Hadoop Components

Leveraging the Hadoop ecosystem, various queries were performed using different components to maximize efficiency for specific tasks:

- **HDFS**: Used as the foundational storage layer, enabling scalable and fault-tolerant storage for the entire YouTube dataset across distributed nodes.
- **Apache Spark**: Utilized for high-speed data processing, including data cleaning, transformation, and complex analyses. Spark enabled faster execution of iterative queries, such as identifying top-rated videos and analyzing viewership patterns.
- **Hive**: Applied for structured queries and batch processing, including analyses such as calculating average ratings and total comment counts by category.
- **Pig**: Employed for ETL operations and data aggregation, extracting insights such as the top 5 longest videos and the uploaders with the highest video counts.
- **HBase**: Leveraged for efficient storage and quick access, particularly for real-time queries on large datasets requiring fast retrieval of specific data points.

8. Performance Analysis and Evaluation

To assess the efficiency of each Hadoop component, query execution times were recorded, offering a comparative analysis of the ecosystem's performance under different conditions and storage architectures. This analysis is presented in the evaluation section to underscore the strengths and limitations of each tool.

9. Data Visualization and Insights

For visual representation, additional visualizations were generated using Spark and Python. These visualizations illustrated key metrics like average views by category,

relationships between video length and view count, and the distribution of ratings across various categories, supporting a comprehensive understanding of the dataset's trends.

FUTURE WORK

While this project provides a foundational analysis of early YouTube data within the Hadoop ecosystem, several avenues for future work could enhance its scope and depth:

1. **Expanding the Dataset**: Including data from more recent years would allow for a longitudinal analysis, tracking how YouTube trends evolved over time. This could involve comparing 2007 metrics with present-day metrics, highlighting shifts in viewer preferences and content strategies.
2. **Automating Data Pipelines**: Implementing workflow automation tools like Apache Oozie could streamline the data pipeline, enabling scheduled data ingestion, transformation, and loading processes. This would make the analysis scalable for larger datasets or frequent updates.
3. **Machine Learning Integration**: Applying machine learning algorithms, such as clustering or classification, could yield insights into viewer behavior patterns or predict video popularity based on historical data. Spark MLlib could be used within the Hadoop ecosystem for scalable machine learning tasks.
4. **Real-Time Analysis**: Using tools like Apache Kafka in combination with HBase could enable real-time data streaming and analysis, allowing for dynamic insights into ongoing viewer engagement and content trends.

LIMITATIONS

Despite its extensive scope, this project faced certain limitations that impacted the analysis and its potential outcomes:

1. **Data Scope**: The dataset is limited to 2007, which constrains the analysis to early YouTube trends without capturing changes over time. As a result, insights are historical and cannot fully account for the evolution of user behavior and platform algorithms.
2. **Platform Constraints**: Each tool in the Hadoop ecosystem has specific strengths but also limitations. For instance, HBase is highly efficient for real-time data access but is not optimized for complex aggregations. Similarly, while Hive and Pig handle large datasets effectively, their performance decreases with more complex data transformations and nested queries.
3. **Limited Real-Time Capabilities**: The Hadoop ecosystem, while efficient for batch processing, is not designed for real-time analysis. Any insights generated are retrospective rather than live, which restricts applications that require immediate data insights.
4. **Resource Intensive**: The project required substantial computational resources for handling the dataset within Hadoop, particularly for memory-intensive tasks in Pig. This constraint could be alleviated with a more optimized or cloud-based setup but is a factor to consider for similar large-scale analyses.

EVALUATION AND PERFORMANCE ANALYSIS

Note: *Execution times (in seconds) for each query may vary across different platforms depending on system specifications, and some queries may not execute in certain environments. The results provided are based on a system with 8GB of physical memory. Please interpret results accordingly, considering your system's architecture and resources.*

Query	**ApachSpark**	**Hive**	**Pig**	**HBase**	**MongoDB**
Top 10 Most Viewed Videos	0.02	5.42	-	86.58	9.54
Average Rating by Category	0.04	4.26	22	91.15	10.02
Videos with Ratings Greater Than 4.5	0.013	6.45	18	84.53	291.56
Average Views per Category	0.04	4.65	18	80.96	15.21
Top 5 Uploaders by Number of Videos	0.05	13.14	-	89.64	26.89
Top 5 Longest Videos	0.01	4.43	-	86.74	9.67
Total Comments Count by Category	0.03	4.98	22	88.84	8.86
Total Number of Videos by Each Uploader	0.04	11.36	22	79.06	21.30
Average Video Length by Category	0.24	4.09	12	79.73	19.07

Query	ApachSpark	Hive	Pig	HBase	MongoDB
Top 10 Most Viewed Videos	0.02	5.42	-	86.58	9.54
Videos with High Views but Low Ratings	0.15	4.85	12	83.96	15.51
Relationship Between Video Length and Views	0.02	4.12	8	82.77	21.77

Visualizations

Fig 1. Heatmap of Execution Time Comparison across Different Platforms.

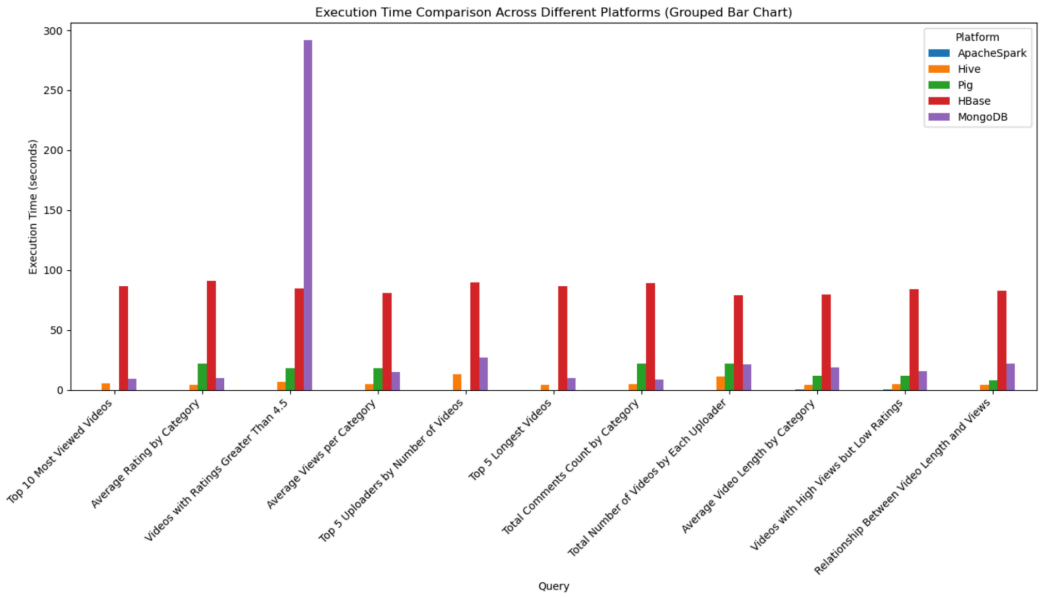

Fig 2. Bar Chart of Execution Time Comparison across Different Platforms.

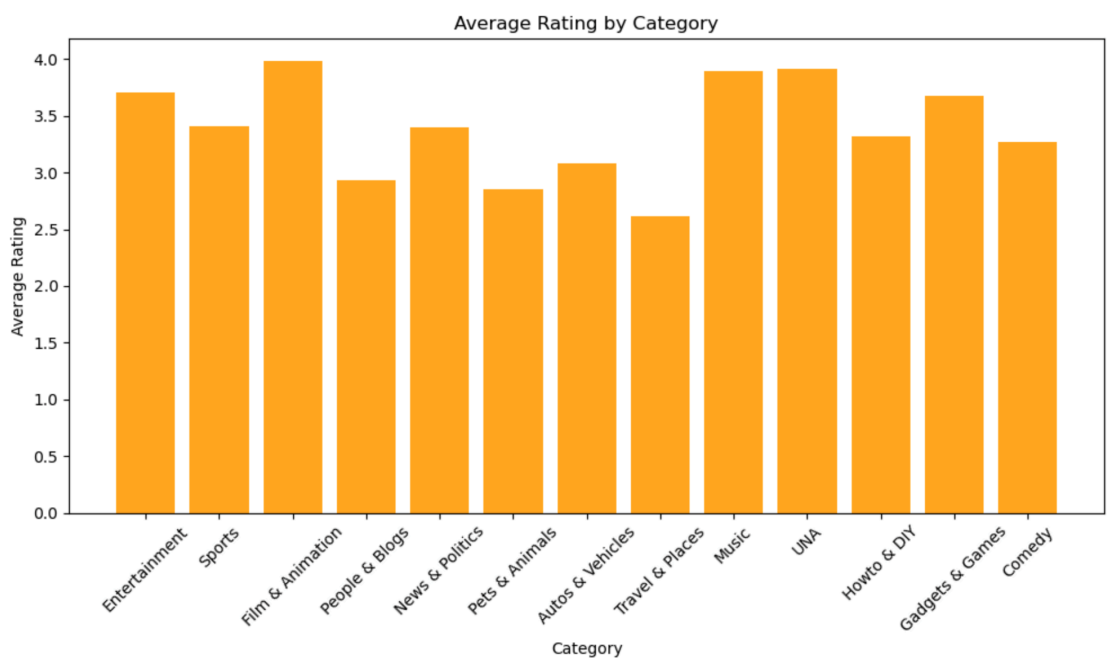

Fig 3. Bar Chart of Average Rating by Category.

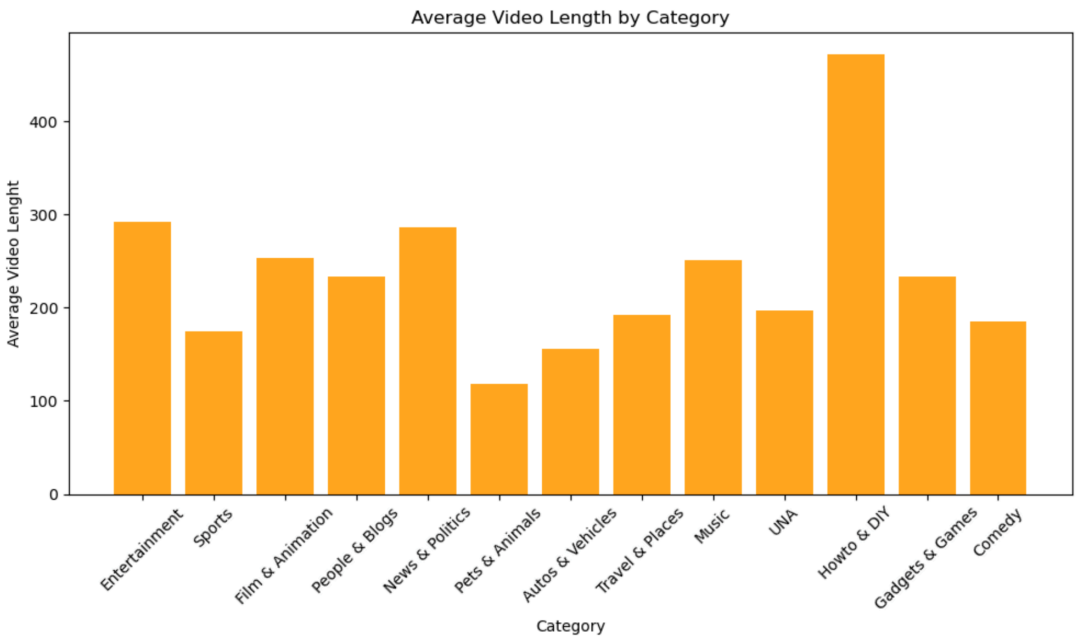

Fig 4. Bar Chart of Average Video Length by Category.

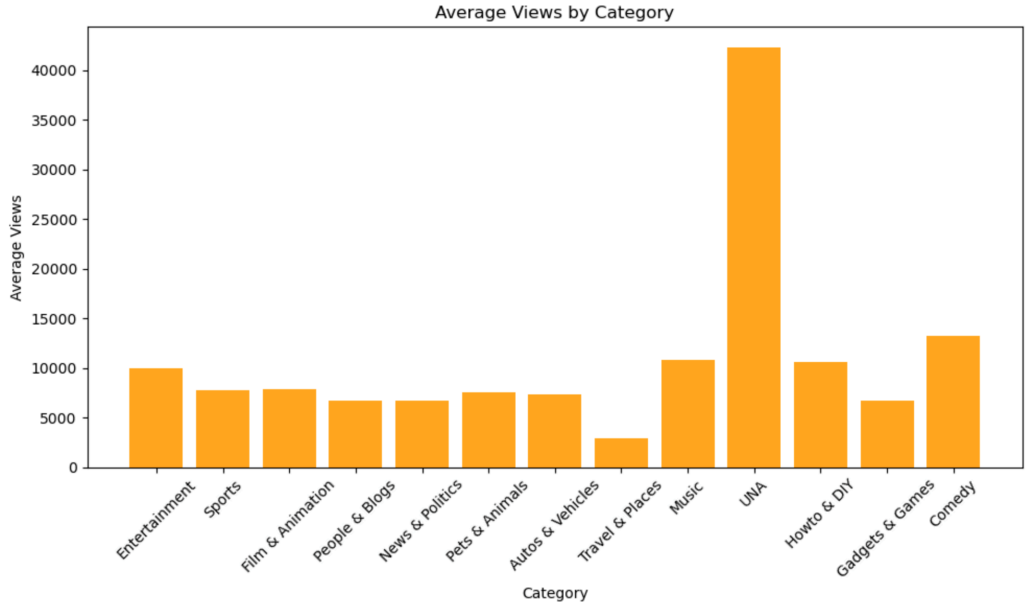

Fig 5. Bar Chart of Average Views by Category.

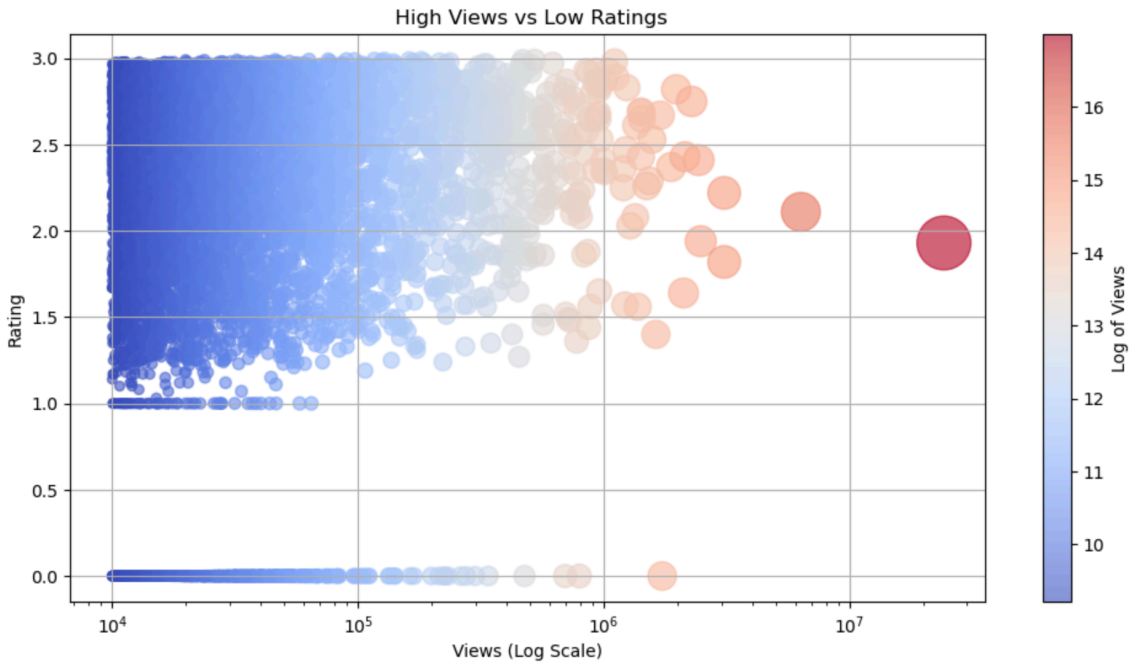

Fig 6. Scatter Plot of High Views vs Low Ratings

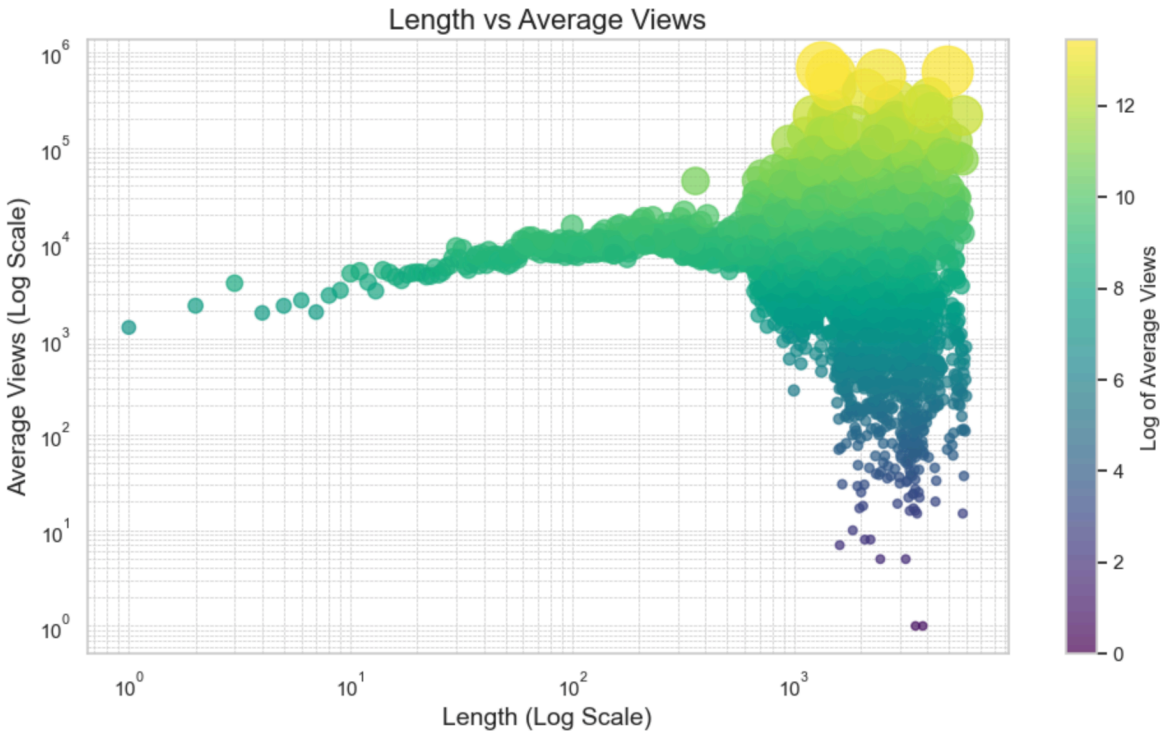

Fig 7. Scatter Plot of Video Length VS Average Views

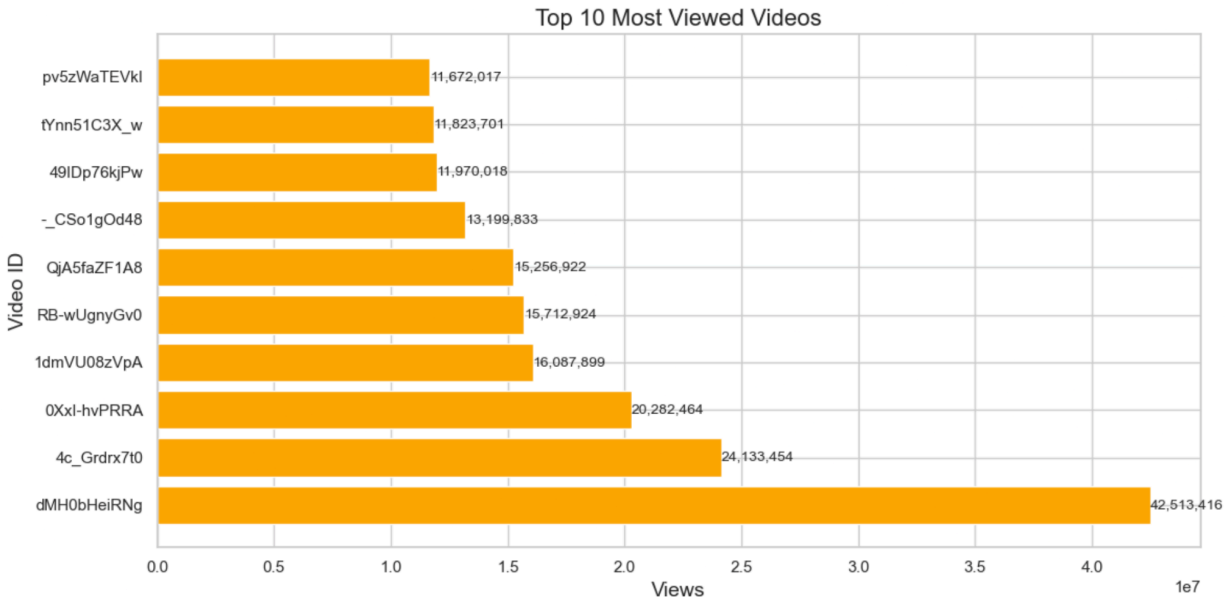

Fig 8. Bar Chart of Top 10 Most Viewed Videos

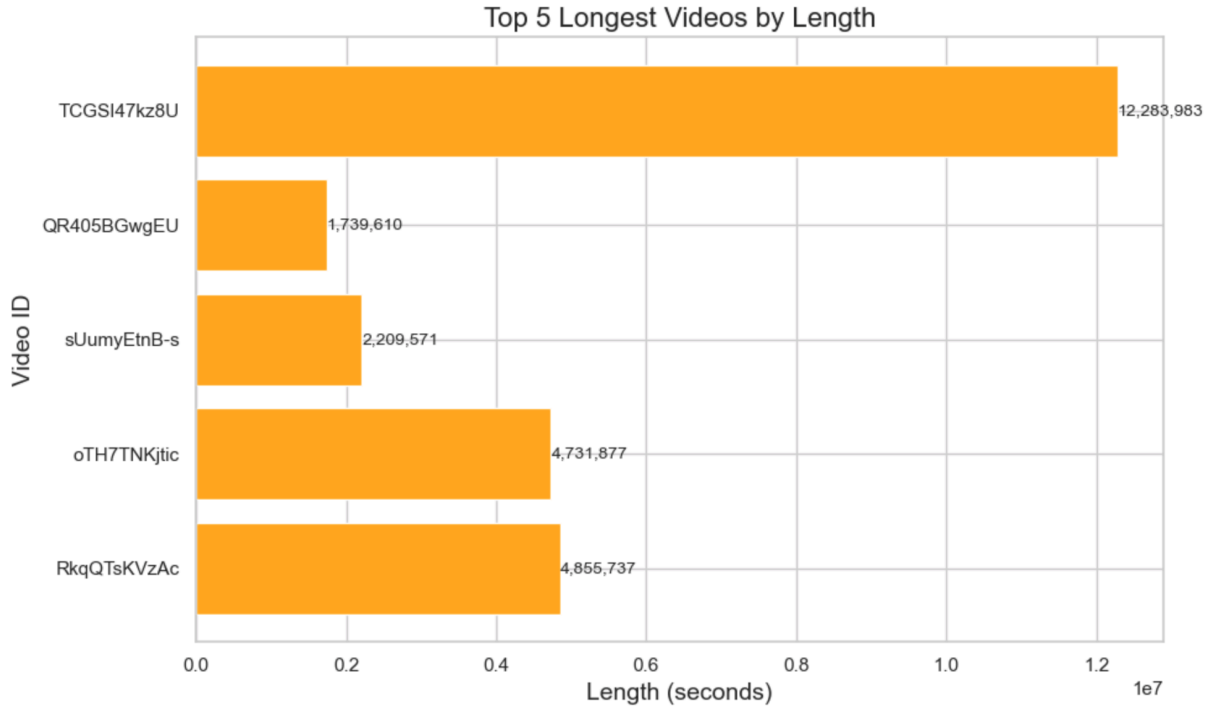

Fig 9. Bar Chart of Top 10 Longest Videos

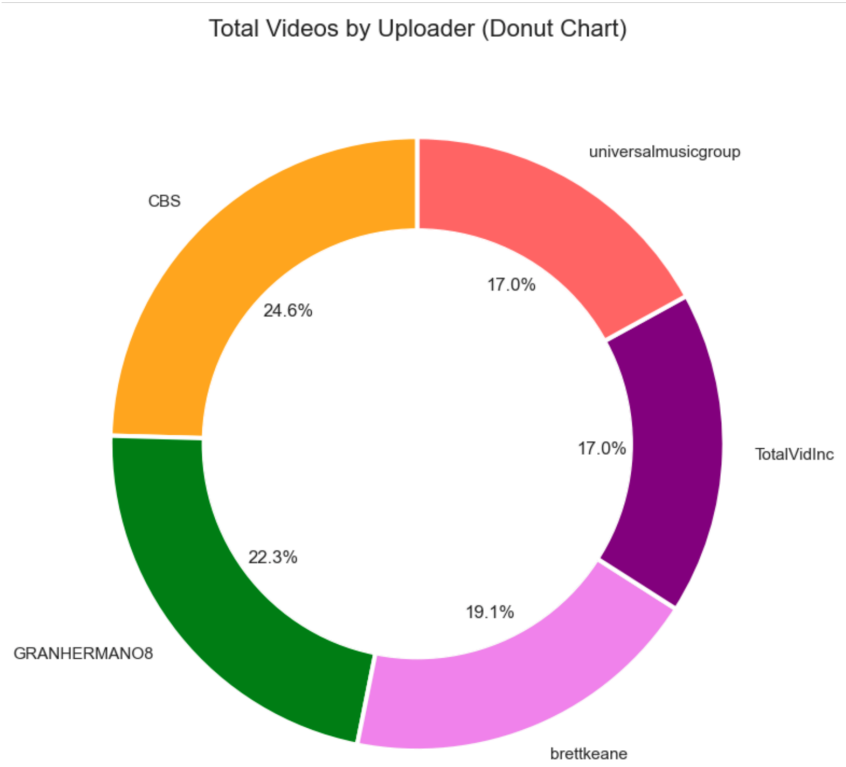

Fig 10. Donut Chart of Top 5 Uploaders by Number of Videos

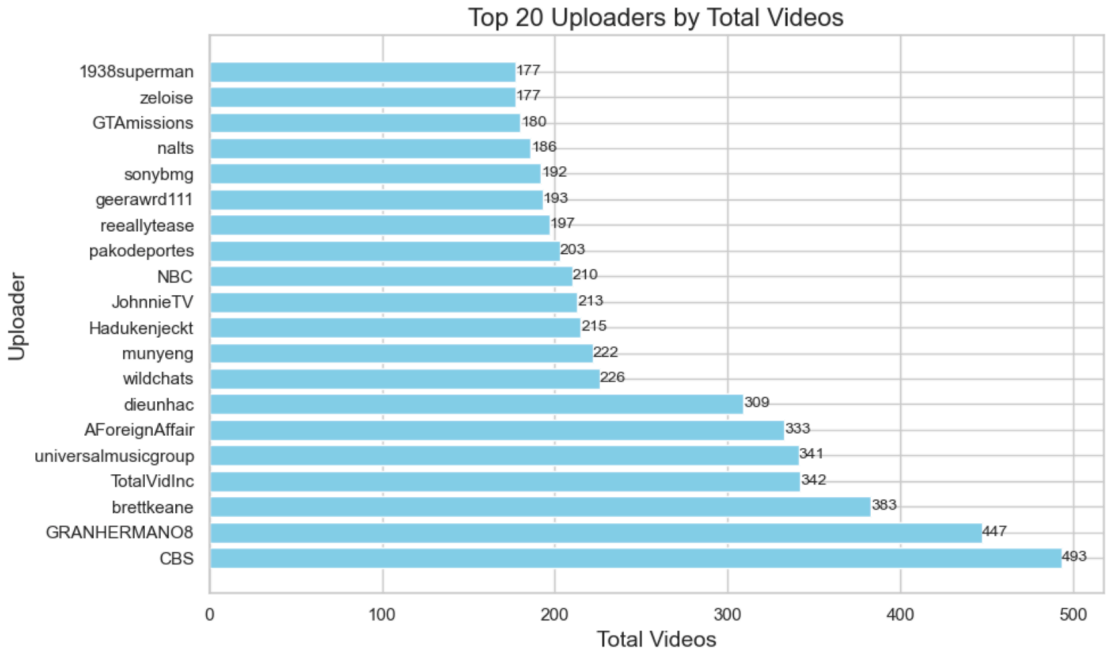

Fig 11. Bar Chart of Top 20 Uploaders by Number of Videos

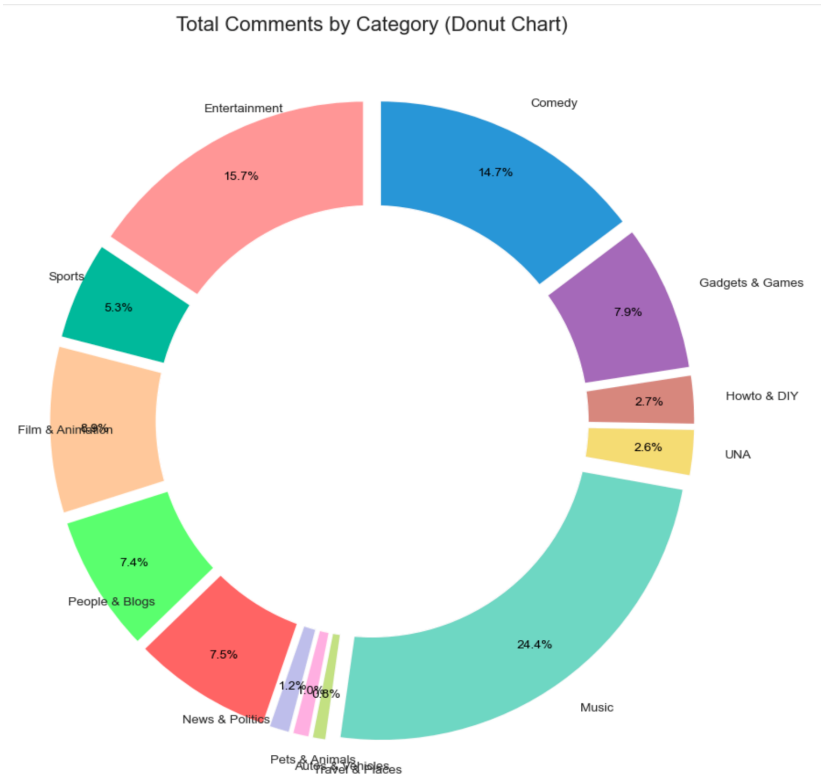

Fig 12. Donut Chart of Total Comments By Category

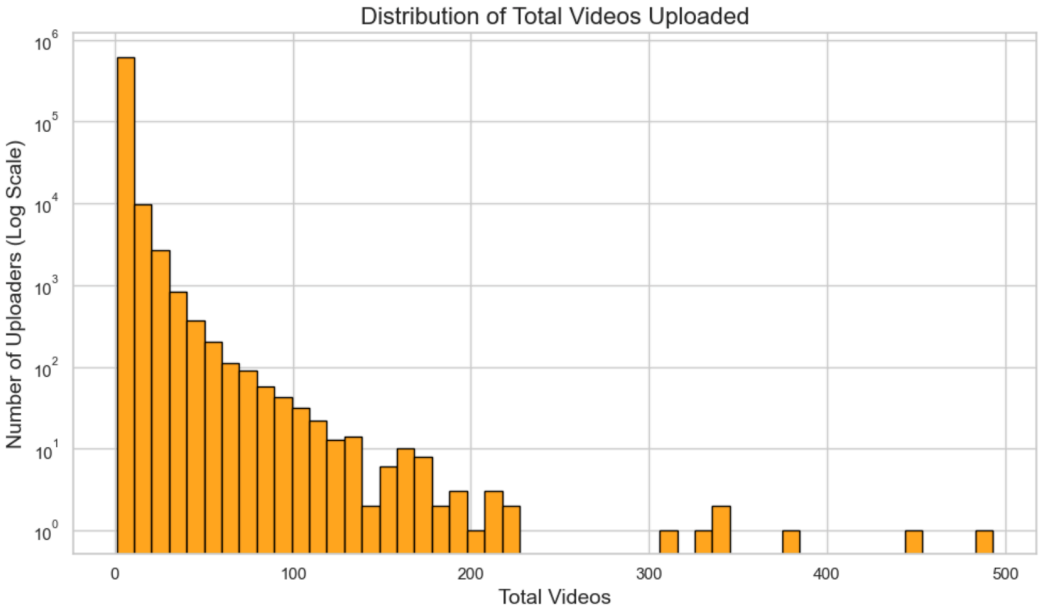

Fig 13. Bar Chart of Total Videos Uploaded

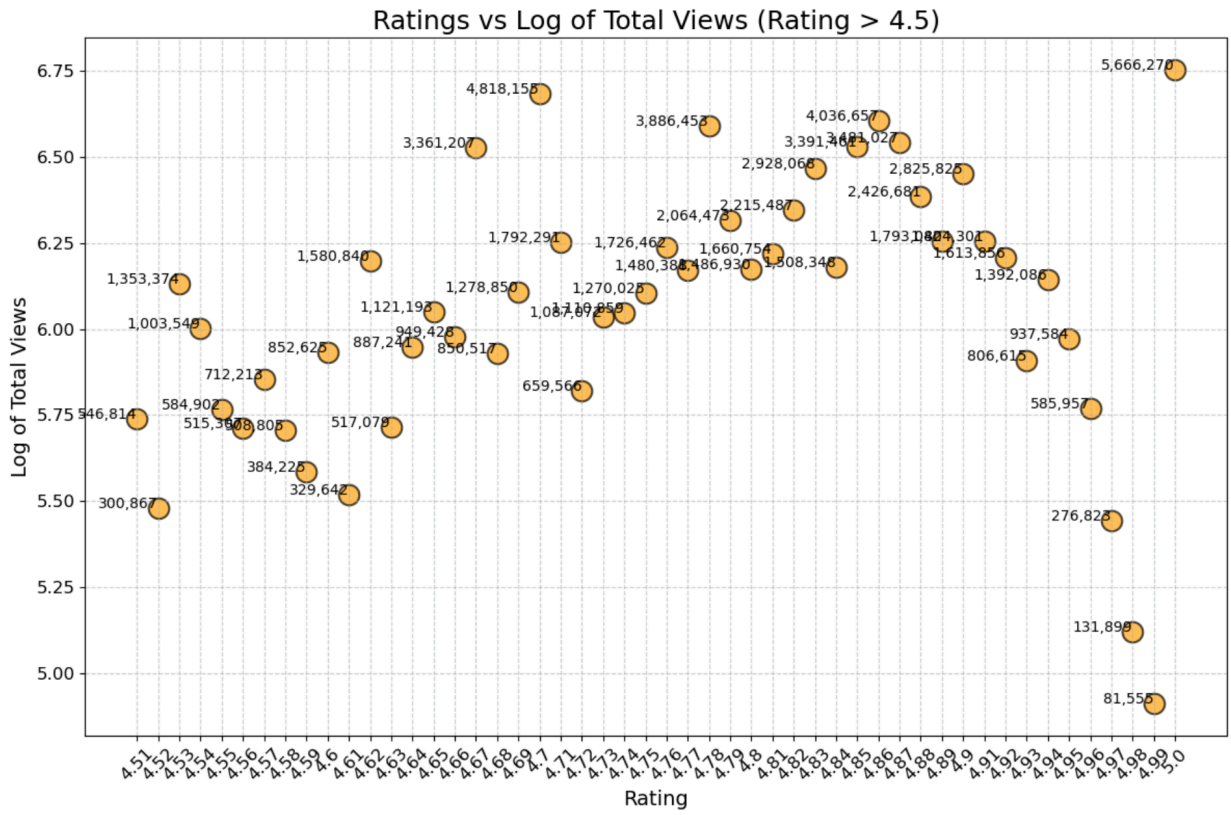

Fig 14. Scatter Plot of Rating Vs Total Views

NOVELTY

This YouTube analysis project is unique in its approach, leveraging the Hadoop ecosystem to perform a comprehensive analysis of early YouTube video metrics from 2007. Using Hive, Pig, and HBase within Hadoop, this project explores viewer engagement, content categories, and video popularity in a time period when YouTube was still in its infancy. By examining this historical data, the project offers a rare glimpse into foundational trends that have influenced how online video has evolved into a major part of social media.

1. Historical Data from Early YouTube (2007)

The exclusive focus on 2007 data makes this project particularly novel. Given YouTube's transformation over the years, most research and analysis focus on modern data; however, this project examines data from one of the platform's earliest years. This dataset captures the engagement patterns, content categories, and trends of 2007, providing a baseline for understanding YouTube's evolution. Analyzing early data can reveal unique insights into foundational video viewing behaviors and popular content types, offering a fresh perspective on user engagement compared to today.

2. Multi-Platform Analysis within the Hadoop Ecosystem

This project's analysis utilizes multiple components within the Hadoop ecosystem to maximize the potential of this dataset:

- **Hive**: The SQL-like interface in Hive enables efficient data aggregation and complex queries over data stored in HDFS. Using Hive, this project performs category-based

analysis, video view trends, and other aggregated metrics, helping to uncover key insights such as average ratings by category and top-viewed videos.

- **Pig**: Known for its powerful scripting capabilities, Pig is used to preprocess, cleanse, and transform the data, which allows for customized data flows. Pig's flexibility makes it ideal for preparing the dataset for other tools and performing text-based analysis and processing, enhancing the quality and consistency of the data before running final analyses.
- **HBase**: A NoSQL database within the Hadoop ecosystem, HBase supports high-speed access to records with minimal latency. Using Video ID as the row key allows for quick lookups, making HBase suitable for querying specific video metrics and time-sensitive data retrieval, such as time-based trends in video engagement. HBase's schema flexibility supports large datasets, making it a valuable tool for storing the extensive YouTube dataset efficiently.

This Hadoop ecosystem architecture enables a robust and flexible analytical approach, allowing each tool to contribute its unique strengths in data storage, processing, and querying. This hybrid model highlights the novelty of using a distributed framework, which maximizes Hadoop's potential for storing and analyzing large-scale datasets effectively.

3. Extensive Query Set for Multi-Dimensional Analysis

The project performs a comprehensive set of queries designed to provide a multi-dimensional analysis of the YouTube data. Each query is tailored to answer specific questions about viewer behavior, content categories, and engagement metrics, enabling a detailed view of early YouTube dynamics. Some of the key queries performed include:

1. **Top 10 Most Viewed Videos**: Identifies the most popular videos, helping understand the content that attracted the highest viewership.
2. **Average Rating by Category**: Provides insights into viewer satisfaction within each category.
3. **Videos with Ratings Greater Than 4.5**: Highlights highly-rated videos, possibly indicating high-quality or niche content.
4. **Average Views per Category**: Shows engagement patterns across categories, identifying popular content types.
5. **Top 5 Uploaders by Number of Videos**: Highlights prolific content creators, showing who contributed most to the platform.
6. **Top 5 Longest Videos**: Examines longer content to see how video length influenced viewer interest.
7. **Total Comments Count by Category**: Measures interaction levels, providing insights into how engaging each category was.
8. **Total Number of Videos by Each Uploader**: Examines uploader activity, identifying the most active content creators.
9. **Average Video Length by Category**: Investigates video length trends across categories to understand genre-specific content styles.
10. **Videos with High Views but Low Ratings**: Identifies videos with high viewership but low ratings, suggesting potentially misleading content.
11. **Relationship Between Video Length and Views**: Explores if there is a correlation between video length and viewership, highlighting potential optimal content lengths.

These queries go beyond basic metrics to uncover nuanced insights into content strategies, viewer engagement, and viewing behavior, creating a comprehensive view of early YouTube dynamics. This diverse set of queries exemplifies the novelty of this project by addressing both high-level trends and specific behavioral insights from multiple perspectives within the Hadoop framework.

4. Historical Relevance and Comparative Insights

Analyzing this 2007 dataset provides unique historical insights and allows for comparative studies with today's YouTube trends. For instance, current trends in video duration, content categorization, and engagement levels can be compared to 2007 data to show how viewer preferences and content strategies have shifted over time. This historical perspective can inform modern content creators and analysts about the evolution of online video, giving insights into the factors that have shaped digital media consumption.

Such comparative insights are valuable not only for academic research but also for content strategy and media analysis, as they offer a window into the changing dynamics of social media. By contextualizing how user engagement and content popularity evolved on YouTube, this project contributes to a broader understanding of digital video trends.

5. Additional Implementation with MongoDB

As an additional implementation, MongoDB is utilized to explore the dataset using a modern NoSQL database structure, providing flexibility for document-based queries. While MongoDB is not the primary focus, this implementation demonstrates how the dataset can be managed and queried within a non-relational database for flexible analysis of attributes such as Related Videos

and uploader-specific metrics. This supplementary MongoDB implementation offers a contemporary perspective, showcasing how modern databases can complement Hadoop-based analytics for specific use cases.

CONCLUSION

This project's novelty lies in its unique combination of historical YouTube data from 2007 and a robust, multi-component Hadoop ecosystem framework, leveraging Hive, Pig, HBase, and MongoDB to deliver in-depth insights into early video engagement, viewer preferences, and content strategies. By performing diverse queries on metrics like video popularity, content category ratings, and uploader activity, the project provides a comprehensive view of YouTube's early dynamics.

The results of the comparative analysis highlight the strengths and weaknesses of each component for different query types:

- **Apache Spark** demonstrated exceptional performance with the fastest execution times across most queries, making it well-suited for real-time analysis and iterative processing tasks.
- **Hive** effectively handled batch processing for structured data but showed longer execution times, particularly for complex aggregations.
- **Pig** was efficient for ETL operations and multi-step transformations, although it struggled with real-time requirements and had slower execution on complex queries.
- **HBase** provided rapid read/write access for real-time querying of unstructured data, ideal for fast data retrieval but less suited for in-depth analytical queries.
- **MongoDB**, as a modern NoSQL solution, exhibited flexibility in handling semi-structured data but displayed high execution times for certain queries, particularly on larger datasets, emphasizing its strengths in schema flexibility over query speed.

This dual approach—utilizing both a distributed Hadoop environment and MongoDB—offers valuable perspectives for historical comparison and contemporary media research, demonstrating how viewer behaviors and content trends have evolved over time. The analysis provides a foundational understanding of early online video trends and highlights the effectiveness of different technologies for specific big data tasks, contributing to ongoing research in social media and digital content evolution.

REFERENCES

Reference No.	Author(s)	Title	Publication/Source	Year	URL/DOI
1	Apache Software Foundation	Apache Hadoop Documentation	Apache Hadoop Project	2023	https://hadoop.apache.org/docs/
2	Apache Software Foundation	Apache Hive Documentation	Apache Hive Project	2023	https://cwiki.apache.org/confluence/display/Hive
3	Apache Software Foundation	Apache Pig Documentation	Apache Pig Project	2023	https://pig.apache.org/docs/
4	Apache Software Foundation	Apache HBase Documentation	Apache HBase Project	2023	https://hbase.apache.org/book.html
5	YouTube Data API Team	YouTube Data API v3 Overview	YouTube Developers	2023	https://developers.google.com/youtube/v3
6	Spark Team	PySpark Documentation	Apache Spark Project	2023	https://spark.apache.org/docs/latest/api/python/
7	YouTube Dataset Team	YouTube Dataset for Research	Simon Fraser University	2023	https://netsg.cs.sfu.ca/youtubedata/