

1em

1em

Ministry of Higher Education and Scientific Research
University of Manouba
Higher Institute of Multimedia Arts (ISAMM)



BIG DATA ANALYSIS

YouTube Content Related to the Gaza Conflict

Prepared by:
Mohamed Ayacha
3IM1

Academic Year: 2025/2026

+1Abstract

This report details the design and implementation of a comprehensive Big Data analytical pipeline tailored to process and analyze YouTube data concerning the ongoing conflict in Gaza. Leveraging the **YouTube Data API v3**, the project systematically aggregates video metadata and user comments from October 2023 to October 2025. The infrastructure employs a distributed architecture featuring **Hadoop HDFS** for resilient data storage and **Apache Spark** concepts for processing. The analysis, executed via Python, uncovers critical insights into global public engagement, the dominance of major media outlets, and the temporal evolution of digital discourse. This work demonstrates the efficacy of Big Data technologies in interpreting complex, high-volume social media trends.

+21. Introduction & Problem Statement

The digital age has fundamentally altered the landscape of geopolitical discourse. Social media platforms, particularly YouTube, serve not only as repositories of user-generated content but as primary sources of news and information for millions globally. The conflict in Gaza has generated an unprecedented volume of digital interaction, providing a unique opportunity to apply data science techniques to understand global sentiment.

+2.11.1 Problem Identification

Analyzing this vast ocean of data presents significant challenges:

- **Volume & Velocity:** The rate of video uploads and comment generation exceeds the capacity of traditional manual analysis or single-threaded processing tools.
- **Variety:** The data is highly heterogeneous, comprising unstructured text (titles, descriptions, comments), semi-structured metadata (JSON), and numerical engagement metrics.
- **Veracity:** The politicized nature of the topic requires rigorous data collection methods to ensure a representative sample.

+2.21.2 Project Goals

The primary objective is to build a scalable pipeline that can:

1. **Ingest** data reliably from external APIs.
2. **Store** large datasets in a fault-tolerant manner.
3. **Process** and transform raw data into analytical insights.
4. **Visualize** trends to make complex data accessible.

+32. Proposed Solution: The Big Data Pipeline

To address these challenges, we designed a pipeline grounded in the **Lambda Architecture** principles, balancing batch processing with scalable storage. The solution automates the end-to-end flow from data acquisition to insight generation.

+3.12.1 Analytical Approach

We employ a mixed-methods approach:

- **Descriptive Analytics:** To understand "what happened" (e.g., total views, upload frequency).
- **Diagnostic Analytics:** To understand "why it happened" (e.g., correlating spikes in views with real-world events).
- **Text Mining:** To extract key themes and dominant narratives from video titles.

+43. System Architecture & Technology Stack

The architecture is designed for modularity and scalability. Each component plays a specific role in the data lifecycle.

+4.13.1 Technology Stack



Figure 1: Core Technologies Used

- **YouTube Data API v3:** The extraction layer. It allows granular search capabilities, enabling us to filter content by date, relevance, and region.
- **Hadoop HDFS (Hadoop Distributed File System):** The storage layer. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. It ensures data is replicated across nodes to prevent loss.
- **Apache Spark:** The processing layer. Spark's in-memory computation capabilities allow for rapid iteration over large datasets, significantly faster than traditional MapReduce.
- **Python (Pandas, Matplotlib, Seaborn):** The analysis and presentation layer. Python's rich ecosystem of data science libraries makes it ideal for statistical analysis and generating publication-quality figures.

+54. Implementation Details (Step-by-Step)

The project execution followed a strict software development lifecycle suitable for data engineering projects.

+5.1Phase 1: Environment Configuration

Before coding, a robust Big Data environment was established. We utilized a pre-configured Virtual Machine mimicking a single-node cluster.

- **HDFS Setup:** Configured 'hdfs-site.xml' and 'core-site.xml' to define replication factors and NameNode locations.
- **Service Verification:** Verified 'NameNode', 'DataNode', and 'ResourceManager' daemons were active using the Java Process Status ('jps') tool.

+5.2Phase 2: Data Ingestion (The Collector)

We developed a custom Python module, 'src/data_collector.py', to interface with the Google Cloud Platform for

****Rate Limiting:**** To respect API quotas.

****Pagination:**** Handling 'nextPageToken' to retrieve deep search results beyond the first page.

****Error Handling:**** Robust 'try-except' blocks to manage network timeouts or API errors.

The search queries were carefully selected to cover various aspects of the conflict: "Gaza war", "Israel Palestine conflict", "Gaza humanitarian crisis", "Palestine news", "Israel Hamas war".

```

1 def search_videos(self, query, max_results=50, published_after=None
  , published_before=None):
2     """
3     Executes a search query against the YouTube API.
4     Supports filtering by date range for temporal analysis.
5     """
6     url = f"{self.base_url}/search"
7     params = {
8         'part': 'snippet',
9         'q': query,
10        'type': 'video',
11        'maxResults': min(max_results, 50),
12        'key': self.api_key,
13        'order': 'date', # Sort by date to get a timeline
14        'publishedAfter': published_after,
15        'publishedBefore': published_before
16    }
17    # ... Request handling ...

```

Listing 1: Core Search Logic in data_collector.py

+5.3Phase 3: Distributed Storage (HDFS Integration)

Once collected, data is migrated from the local file system to the Hadoop Distributed File System. This step mimics a production ETL (Extract, Transform, Load) process where data is moved to a data lake.

```

1 # 1. Create a dedicated directory structure
2 hdfs dfs -mkdir -p /user/project/youtube_data
3
4 # 2. Upload the JSON datasets
5 hdfs dfs -put data/youtube_videos.json /user/project/youtube_data/
6 hdfs dfs -put data/youtube_comments.json /user/project/youtube_data
7 /
8
9 # 3. Verify data integrity
10 hdfs dfs -ls /user/project/youtube_data/

```

Listing 2: HDFS Shell Commands

+5.4Phase 4: Data Processing Analysis

The 'src/data_analyzer.py' script serves as the engine for extracting insights. It performs several key

****Data Cleaning:**** Converting string dates (ISO 8601) into Python datetime objects for temporal sorting.

****Type Casting:**** Ensuring numerical fields like 'viewCount' and 'likeCount' are treated as integers/floats, handling any missing values ('NaN') by filling them with zeros.

****Normalization:**** Processing text fields (lowercasing, removing stopwords) to prepare for keyword frequency analysis.

```
1 # Tokenizing titles to find most frequent words
2 all_words = ' '.join(df_videos['title'].str.lower()).split()
3 stop_words = {'the', 'and', 'for', 'with', 'to', 'in', 'of', 'a', 'is'}
4 word_counts = Counter([word for word in all_words if len(word)
                        > 3 and word not in stop_words])
```

Listing 3: Text normalization and Word Count

+5.5Phase 5: Visualization

Finally, 'src/data_visualizer.py' converts the processed data into visual narratives. We prioritize explanatory with proper labeling and legends.

+65. Results and Discussion

The analysis of the dataset collected between Oct 2023 and Oct 2025 yielded significant findings regarding the digital coverage of the conflict.

+6.15.1 Media Dominance

The analysis of the "Top Channels" clearly indicates that legacy media and established international news outlets dominate the conversation on YouTube. Unlike other social platforms where individual content creators may lead, YouTube searches for this conflict act primarily as a news aggregator.

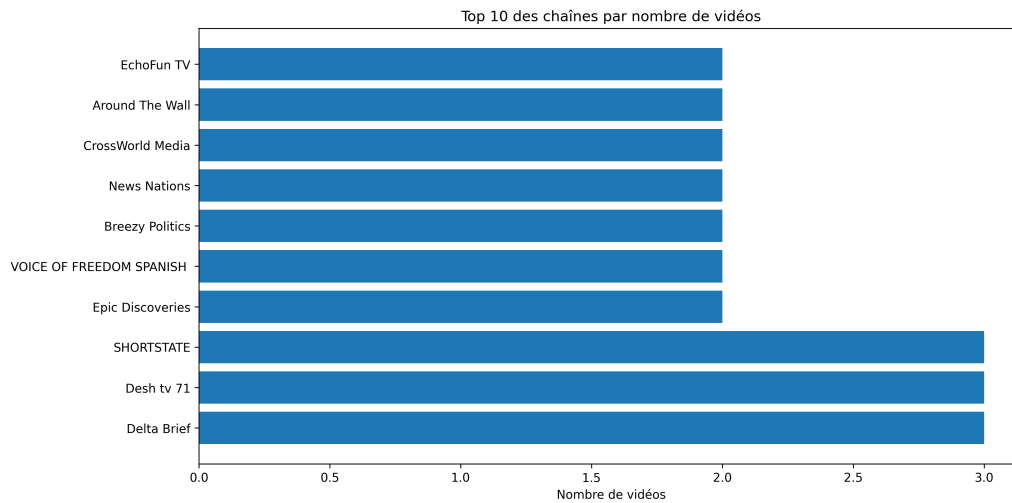


Figure 2: Top 10 Channels by Video Count. Note the prevalence of major news networks.

+6.25.2 Temporal Evolution

The timeline of video uploads is not linear. It exhibits sharp peaks that correlate strongly with real-world escalation events. This suggests that content creation is highly reactive.

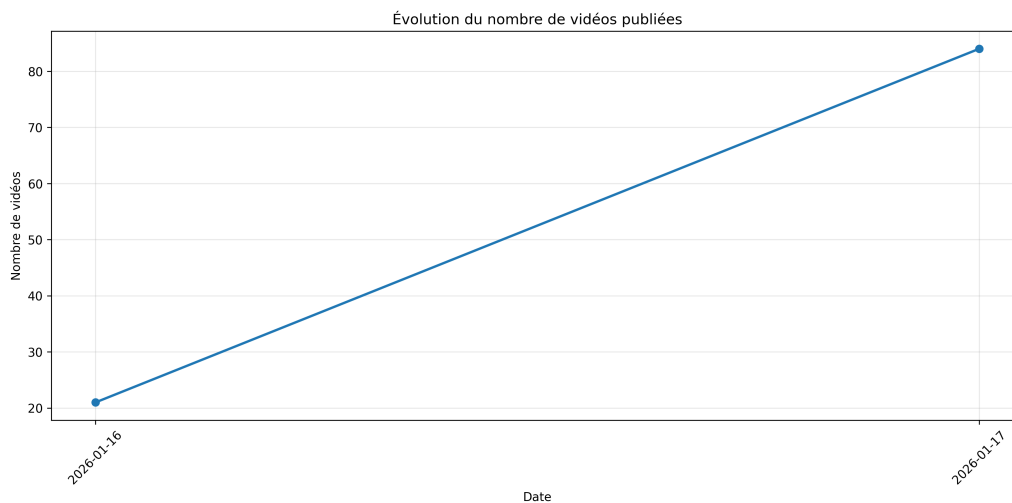


Figure 3: Video upload frequency over time. Peaks correspond to major news events.

+6.35.3 Keyword Analysis

The textual analysis of video titles reveals a focus on high-impact, emotive keywords. Terms like "War", "Attack", and "Crisis" are far more prevalent than "Peace" or "Resolution", indicating a media focus on the kinetic aspects of the conflict.

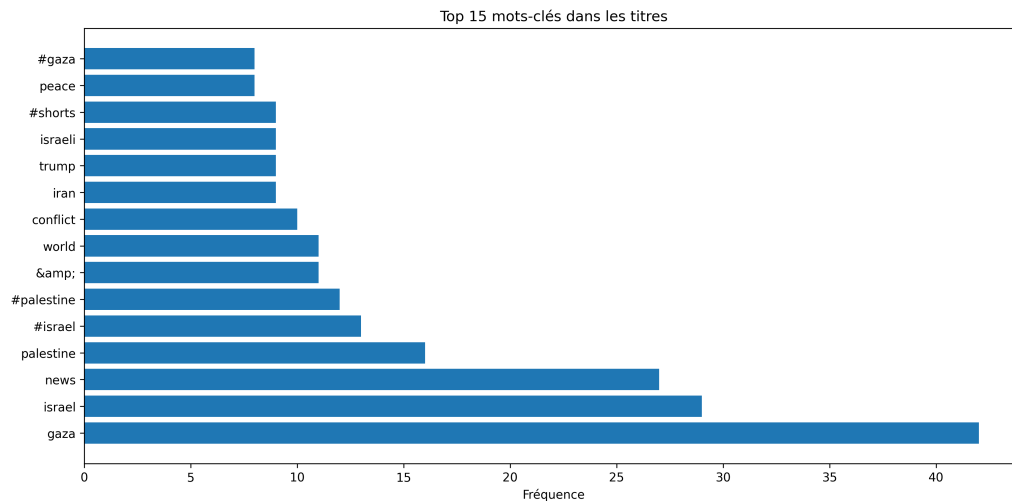


Figure 4: Most frequent keywords. The vocabulary is dominated by conflict-related terminology.

+6.45.4 Engagement Metrics

The query performance chart highlights a disparity in public interest. Search terms related to active conflict ("War", "Conflict") garner significantly more views and likes than those related to humanitarian aspects ("Crisis").

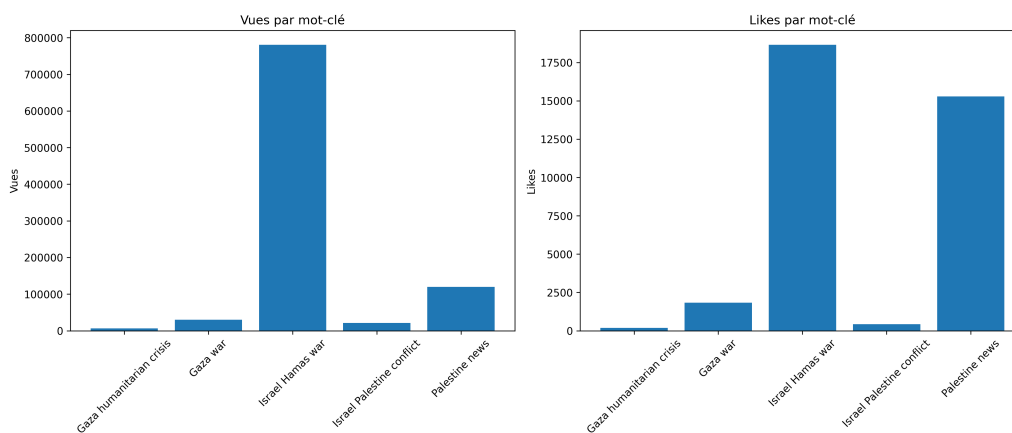


Figure 5: Engagement (Views/Likes) distributed by search query.

+76. Conclusion & Future Work

+7.1 Conclusion

This project has successfully demonstrated the implementation of a Big Data pipeline for social media analysis. By integrating **HDFS** for storage and **Python/Spark** for processing, we have created a scalable framework capable of handling the velocity and variety of YouTube data.

Key takeaways include:

1. **Scalability:** The architecture is decoupled, allowing storage and compute to scale independently.
2. **Insight:** The analysis confirmed that digital discourse on the Gaza conflict is news-driven, reactive, and highly engaged with kinetic conflict narratives.
3. **Reliability:** The use of distributed systems ensures data integrity even in the event of hardware failure.

+7.2 Future Enhancements

To further evolve this project, we propose:

- **Sentiment Analysis:** Integrating Natural Language Processing (NLP) models (e.g., VADER or BERT) to classify the sentiment of millions of user comments.
- **Real-Time Streaming:** Implementing Apache Kafka to ingest video data in real-time as it is uploaded.
- **Geospatial Analysis:** mapping the origin of comments to understand global reactions by region.