Multilinguistic Sentimental Analysis using AWS

Major Project Report

***Submitted in partial fulfillment of the requirements for the award of the Degree* of**

Bachelor of Technology (B. Tech)

in

COMPUTER SCIENCE AND ENGINEERING

by

MADHAGONI SRAVANI	**21AG1A0597**
PAVANI K M	**21AG1A05B3**
B SHIVA KUMAR	**22AG1A0509**
R SANDHYA	**21AG1A05B5**

Under the Esteemed Guidance of
Dr. Ch. Vijaya Kumar
Assistant Professor & HOS

Department of Computer Science and Engineering
ACE ENGINEERING COLLEGE
An AUTONOMOUS Institution
NBA Accredited B. Tech Courses, Accorded NAAC 'A' Grade
(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telangana)
Ankushapur (V), Ghatkesar (M), Medchal – Malkajgiri Dist - 501 301.

JUNE - 2025

ACE

Engineering College

An AUTONOMOUS Institution
NBA Accredited B. Tech Courses, Accorded NAAC 'A' Grade
Website: www.aceec.ac.in E-mail: info@aceec.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Mini project work entitled **" Multilinguistic sentimental analysis using AWS"** is being submitted by **MADHAGONI SRAVANI (21AG1A0597), PAVANI K M (21AG1A05B3), B SHIVA KUMAR (22G1A0509), RAVULA SANDHYA (21AG1A05B5)** in partial fulfillment for the award of Degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** to the Jawaharlal Nehru Technological University, Hyderabad during the academic year 2024-25 is a record of bonafide work carried out by them under our guidance and supervision.

The results embodied in this report have not been submitted by the student to any other University or Institution for the award of any degree or diploma.

Internal Guide	**HoS**	**Dean-CSE**
Dr.Ch.Vijaya Kumar	**Dr.V.Ravi Kumar**	**Dr.M.V.Vijaya Saradhi**
Assistant Professor & HOS	**Professor & HOS**	**Professor & Dean of Dept**
CSE		

Project Coordinator
Ms. SHUBHANGI MAHULE
Assistant Professor

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We would like to express our gratitude to all the people behind the screen who have helped us transform an idea into a real-time application.

We would like to express our heartfelt gratitude to our parents without whom we would not have been privileged to achieve and fulfill our dreams.

A special thanks to our General Secretary, **Prof. Y. V. Gopala Krishna Murthy,** for having founded such an esteemed institution. Sincere thanks to our COO **Mr. Y.V. Raghu Vamshi**, for support in doing project work. I am also grateful to our beloved principal, **Dr. Dr. K. S. Rao** for permitting us to carry out this project. I am very much thankful to our beloved Vice Principal & Dean-Skill Development **Dr. M. Murali** for his continuous encouragement to fulfill our project.

We profoundly thank **Dr. M. V. Vijaya Saradhi,** Professor & Dean of the Department of Computer Science and Engineering, who has been an excellent guide and also a great source of inspiration for our work.

We extremely thank **Ms. Shubhangi Mahule**, Associate Professor, and Project Coordinator, who helped us in fulfilling all aspects of the completion of our Major-Project. We are very thankful to our internal guide **Dr. Ch. Vijaya Kumar**, Assistant Professor & HOS who has been excellent and also given continuous support for the Completion of our project work.

The satisfaction and euphoria that accompany the successful completion of the task would be great, but incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success. In this context, we would like to thank all the other staff members, both teaching and non-teaching, who have extended their timely help and eased our task.

MADHAGONI SRAVANI (21AG1A0597)
PAVANI K M (21AG1A05B3)
B SHIVA KUMAR (22AG1A0509)
R SANDHYA (21AG1A05B5)

DECLARATION

We hereby declare that this mini project entitled "**Multilinguistic sentimental analysis using AWS**" submitted to the ACE Engineering College, is a record of an original work done by us under the guidance of **Dr. Ch. VIJAYA KUMAR**, Professor and HOS of the Department of Computer Science and Engineering, **ACE Engineering College**, and this project work submitted in the partial fulfillment of the requirements for the mini project; the results embodied in this thesis have not been submitted to any other university or institute for award of any degree or diploma.

MADHAGONI SRAVANI (21AG1A0597)
PAVANI K M (21AG1A05B3)
B SHIVA KUMAR (22AG1A0509)
R SANDHYA (21AG1A05B5)

INDEX

LIST OF FIGURES

ABSTRACT

Multilingual sentiment analysis is a technique that cracks the code of emotions across languages. It analyses written text, like social media posts or customer reviews, to understand if the overall tone is positive, negative, or neutral. This is particularly useful for businesses that operate internationally, as it allows them to gather insights from a wider audience that speaks various languages. Multilingual sentiment analysis is a technique that cracks the code of emotions across languages. It analyses written text, like social media posts or customer reviews, to understand if the overall tone is positive, negative, or neutral. This is particularly useful for businesses that operate internationally, as it allows them to gather insights from a wider audience that speaks various languages.

The project integrates AWS services such as **Amazon Comprehend** for multilingual sentiment analysis, **AWS Lambda** for serverless processing, **Amazon S3** for data storage, and **Amazon RDS** for structured data management. It also employs **Amazon Translate** to bridge language barriers and ensure accurate sentiment detection across diverse linguistic inputs.

By utilizing a cloud-based architecture, the system ensures scalability, reliability, and real-time analysis of textual data. The project aims to assist organizations in making data-driven decisions by extracting actionable insights from user feedback, reviews, and social media content in various languages.

CHAPTER 1
INTRODUCTION

In today's globalized digital environment, individuals across the world are constantly expressing their thoughts, opinions, and feedback through various online platforms, particularly on widely used platforms like YouTube. These comments offer valuable insights into public opinion, customer satisfaction, and market trends. However, the diversity of languages and cultural expressions presents a significant challenge when trying to analyse these sentiments at scale. Traditional sentiment analysis tools often struggle with grammatical differences, contextual nuances, and the scarcity of labelled data in low-resource languages, leading to inaccurate or incomplete insights.

This project, titled **"Multilingual Sentiment Analysis of YouTube Comments Using AWS"**, aims to build a comprehensive and automated system capable of accurately analysing sentiment in comments written in various languages. The primary goal is to address the linguistic and technical challenges associated with multilingual sentiment analysis by leveraging cloud-based AI services provided by Amazon Web Services (AWS). Specifically, the system uses **Amazon Comprehend** for detecting sentiment (positive, negative, neutral, or mixed) from text data, and **Amazon Translate** to translate non-English comments into English for more consistent analysis. These services are integrated to work seamlessly in a batch processing mode, which allows the system to process large volumes of data effectively.

The comment data is extracted from YouTube using its public API, and then securely stored in **Amazon S3 (Simple Storage Service)**, which serves as the central data repository. To manage permissions and ensure secure usage of AWS services, **AWS Identity and Access Management (IAM)** is employed. The analysis process is designed around the **asynchronous job-based architecture** of Amazon Comprehend, which enables efficient processing of bulk data without the need for continuous compute resources, thereby reducing costs and improving system performance. This serverless approach ensures that the system can automatically scale based on the volume of input data, making it suitable for both small-scale and large-scale deployments.

One of the key aspects of this project is its **cost-effectiveness**. By operating within the AWS Free Tier wherever possible, and using pre-built AI services instead of training custom models, the solution avoids unnecessary infrastructure and licensing expenses. This makes it especially

suitable for startups, academic research, and organizations that want to implement multilingual sentiment analysis without investing heavily in machine learning infrastructure. The decision to use asynchronous batch processing further contributes to lower compute costs, while still maintaining high performance for processing large datasets.

The system developed in this project offers practical applications in areas such as **social media monitoring**, **customer feedback evaluation**, **brand reputation management**, and **market research**. By understanding the sentiments expressed in YouTube comments across various languages, organizations can respond more proactively to public opinion, improve customer experience, and make better-informed decisions. Furthermore, the automated and scalable nature of the solution significantly reduces the manual effort involved in multilingual analysis. In conclusion, this project demonstrates the feasibility and effectiveness of a cloud-based solution for real-time and large-scale multilingual sentiment analysis. It highlights the strengths and limitations of AWS services like Comprehend and Translate when dealing with linguistic diversity, and proposes a secure, scalable, and low-cost architecture that can be extended to other platforms beyond YouTube. Future improvements could include refining the analysis for specific languages, adding domain-specific tuning, and incorporating advanced visualizations for better insight delivery.

CHAPTER 2
LITERATURE SURVEY

The review explores multilingual sentiment analysis, incorporating YouTube comment text to gain deeper emotional insights. The following papers were analysed:

[1] Zero-Shot Multilingual Sentiment Analysis Using Transformer-Based Models (2019)
The paper delves into zero-shot approaches to multilingual sentiment analysis, emphasizing their effectiveness across diverse languages. It highlights the adaptability of transformer models in understanding sentiment without direct training on specific language datasets. Challenges such as domain-specific nuances and limited training resources are addressed. The authors propose solutions leveraging large language models (LLMs) for better performance. The study aims to bridge language barriers in sentiment analysis using innovative methodologies.

[2] Leveraging Multilingual Resources for Language Invariant Sentiment Analysis (2020)
This research investigates the role of machine translation in sentiment analysis to achieve language invariance. The authors use multilingual resources to train models capable of consistent performance across languages. A detailed methodology incorporates translation pipelines integrated with sentiment detection tools. Key findings suggest that while translation aids in cross-language sentiment detection, subtle cultural nuances often pose challenges. This work underscores the importance of high-quality translations in multilingual sentiment tasks.

[3] Audio and Text Sentiment Analysis of Radio Broadcasts (2023)
This work explores sentiment extraction from both audio and text, using radio broadcasts as a case study. It integrates tools like Vokaturi and VADER to analyze emotions in news audio. The hybrid "bifurcate and mix" model enhances accuracy in detecting emotional tone. It reveals useful trends in media sentiment. The findings aid future applications in journalism and communication.

[4] The Multimodal Sentiment Analysis in Car Reviews (MuSe-CaR): Dataset Collection, Insights and Improvements (2023)
This paper introduces the MuSe-CaR dataset, a rich multimodal resource combining text, audio, and visual reviews of cars. It supports sentiment, emotion, and trustworthiness prediction. The dataset is designed for real-world robustness, with diverse annotations. A multi-

head attention model is proposed for better performance. This work enhances affective computing research.

[5] A Survey of Sentiment Analysis: Approaches, Datasets, and Future Research (2023)
This recent survey reviews sentiment analysis techniques categorized into machine learning, deep learning, and ensemble methods. It discusses preprocessing, feature extraction (e.g., TF-IDF, Word2Vec), and classification models. The authors highlight practical applications and future research directions. It provides comparative evaluations of datasets and models. The paper is a valuable reference for researchers.

[6] A Review on Text Sentiment Analysis With Machine Learning and Deep Learning Techniques (2024)
This paper comprehensively reviews advancements in machine learning and deep learning for sentiment analysis. It categorizes models by complexity, input representation, and learning architecture. Deep models like RNNs, CNNs, and attention-based networks are emphasized. The paper introduces a novel taxonomy for analysis. It's a helpful guide for implementing modern NLP approaches.

[7] Transformer-Based Deep Learning Models for Sentiment Analysis (2024)
This article investigates the application of transformer models such as BERT and RoBERTa in sentiment classification. It reviews their performance, scalability, and adaptability across domains. The paper emphasizes contextual embeddings and transfer learning. It shows how transformers outperform earlier ML models. Insights are drawn from various sentiment datasets.

[8] A Survey on Sentimental Analysis Approaches, Datasets, Challenges and Applications (2024)
This document (title inferred from filename) likely provides a comprehensive review of sentiment analysis techniques and datasets. It probably discusses limitations in current models and highlights key challenges like domain adaptation and resource constraints. The survey may also propose future directions. Exact details were limited, but the paper fits well within modern sentiment research surveys.

[9] Advancing Aspect-Based Sentiment Analysis in Course Evaluation: A Multi-Task Learning Framework with Selective Paraphrasing, (2025)

This study proposes a multi-task learning model for aspect-based sentiment analysis (ABSA) in student feedback. It combines aspect extraction and sentiment classification using models like BERT and RoBERTa. Selective paraphrasing enhances training data diversity without distorting sentiment. The SP-MTL-BERT model showed superior accuracy. It's a promising approach for educational feedback analysis.

[10] Multilingual Sentiment Analysis with LLMs (2024)

This study focuses on using ensemble language models for multilingual sentiment analysis. It discusses the integration of large language models (LLMs) to handle diverse linguistic datasets effectively. The authors present a novel framework combining several LLMs for enhanced prediction accuracy. Special attention is given to overcoming linguistic complexities and data sparsity. This approach showcases the potential of LLMs in advancing multilingual sentiment analysis.

[11] M2SA: Multimodal and Multilingual Model for Sentiment Analysis of Tweets (2024)

This paper introduces a multimodal and multilingual model for cross-lingual sentiment analysis, particularly focusing on tweets. It employs transfer learning techniques to adapt models trained on one language to others. The research highlights the importance of multimodal inputs (text, images) for comprehensive sentiment understanding. Innovative methods for addressing challenges like domain adaptation and data diversity are discussed. The results demonstrate the model's efficiency in cross-lingual and multimodal sentiment analysis.

[12] Sentiment Analysis Using Amazon Web Services and Microsoft Azure (2024)

This paper explores the application of Amazon Web Services (AWS) and Microsoft Azure for sentiment analysis tasks. It compares the capabilities of these platforms in handling diverse datasets, focusing on their ease of use, accuracy, and scalability. The authors discuss the integration of cloud services for real-time sentiment extraction. They also address challenges, including cost-effectiveness and latency. This study provides practical insights for businesses aiming to leverage cloud technologies for sentiment analysis.

CHAPTER 3
SYSTEM ANALYSIS

3.1 PROBLEM STATEMENT

Businesses and organizations process vast amounts of multilingual user-generated content from platforms like social media, customer reviews, and feedback systems, making accurate sentiment analysis across diverse languages a significant challenge. Traditional sentiment analysis tools often fall short due to linguistic diversity, contextual nuances, and scalability issues, while manual analysis is both time-consuming and impractical. This project proposes a cloud-based Multilingual Sentiment Analysis System, initially focusing on YouTube comments, by extracting and analyzing their content using AWS services like Amazon Comprehend for sentiment detection and Amazon Translate for language translation. By automating and scaling the sentiment analysis process, the system will deliver accurate, real-time insights, enabling organizations to overcome language barriers, monitor global customer sentiments, and make informed, data-driven decisions efficiently.

3.2 MODULES AND THEIR FUNCTIONALITIES:

3. 2. 1. SYSTEM:

1. **Data Collection:** Text data is extracted using the YouTube API, capturing multilingual comments and metadata for analysis.
2. **Data Storage:** Extracted data is stored securely in Amazon S3 buckets for further processing.
3. **Sentiment Analysis:** Amazon Comprehend processes the stored data directly from S3, analyzing text in its original language and assigning sentiment scores.
4. **Output Generation:** The analysis results, including sentiment scores and categorizations, are packaged into .tar.gz files containing JSON outputs.
5. **Categorization:** The JSON output is parsed and categorized into sentiment classifications (Positive, Negative, Neutral) for easier interpretation.
6. **Displaying Results:** The categorized results are uploaded, enabling users to check the sentiment scores and gain insights from the processed data.

3. 2. 2 USERS:

1. **Upload Text Data:** Users can upload the extracted .tar.gz files for sentiment analysis or access pre-processed results.
2. **View Results:** Sentiment scores and categorizations are displayed, allowing users to interpret multilingual sentiment trends effectively.
3. **Generate Reports:** Users can generate detailed reports summarizing sentiment classifications and trends for decision-making.

CHAPTER 4
SOFTWARE DESIGN

4.1. ARCHITECTURE DIAGRAM

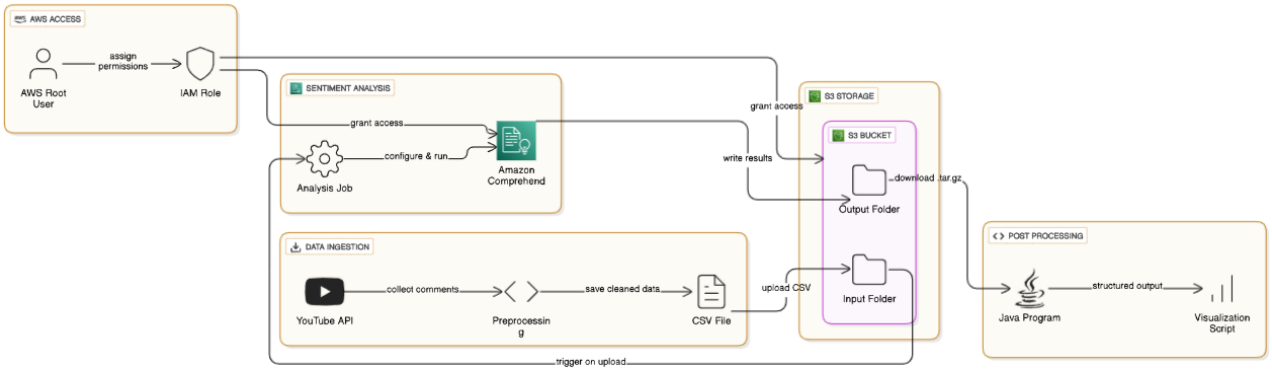


Fig 4. 1. 1. SYSTEM ARCHITECTURE

This architecture illustrates a complete pipeline for sentiment analysis of YouTube comments using AWS services and custom processing. The workflow consists of five primary stages:

1. AWS Access

- The AWS Root User creates an IAM Role to manage permissions for services like Amazon Comprehend and S3.

2. Data Ingestion

- YouTube API collects video comments.
- Comments are preprocessed and saved as a CSV file.
- The file is uploaded to the Input Folder in an S3 Bucket.

3. Sentiment Analysis

- An Analysis Job is configured with Amazon Comprehend.
- Comprehend reads the CSV from S3, analyzes sentiment, and writes results to the Output Folder.

4. S3 Storage

- S3 acts as temporary storage:
- Input Folder: holds cleaned comment data.
- Output Folder: stores sentiment results.

5. Post Processing

- A Java Program downloads the output.
- Data is parsed and passed to a Visualization Script for generating sentiment charts.

4.2. DATA FLOW DIAGRAMS

A Data Flow Diagram (DFD) is a graphical tool used to visualize the flow of information in a system or process. It uses a standardized set of symbols to represent:

- **Processes:** Transformations that convert data from one form to another.
- **Data Flows:** Movement of data between processes, external entities, and data stores.
- **External Entities:** Sources and destinations of data outside the system.
- **Data Stores:** Places where data is kept for future

DFDs are helpful for:

- **Understanding how a system works:** By following the data flow, you can see how data is processed and transformed.
- **Communicating system requirements:** DFDs are a clear and concise way to show what data is needed and how it will be used.
- **Designing new systems:** DFDs can be used to plan out the data flow of a new system before it is built.

4. 2. 1. DFD LEVEL – 0

Level 0 (Context Diagram): A high-level overview of the entire system, showing its main processes and external entities. The below Fig No. 4. 2. 1 shows a high-level overview of the Multilingual sentiment analysis.

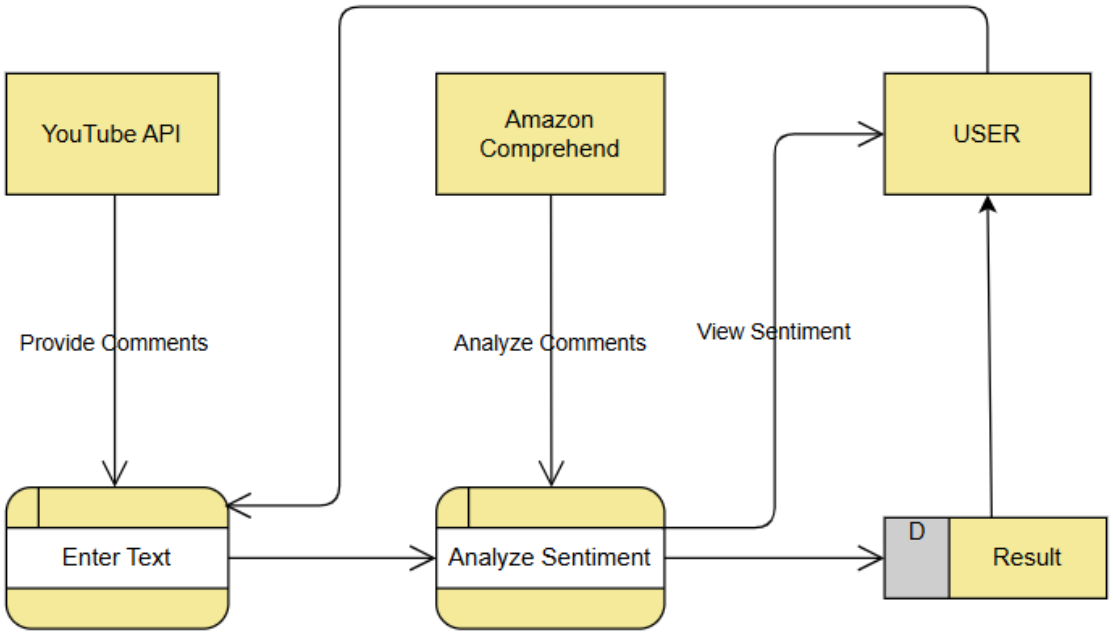


Fig 4. 2. 1. DFD LEVEL – 0

This Level-0 DFD represents the high-level overview of a multilingual sentiment analysis system that processes user-generated comments, analyzes their sentiment using Amazon Comprehend, and presents the results back to the user.

Processes:

1. **Enter Text**: This process takes input from the YouTube API, fetching comments based on user input such as video links or keywords.
2. **Analyze Sentiment**: This process sends the extracted comments to Amazon Comprehend, which performs natural language processing to determine sentiment (positive, negative, neutral, or mixed).

External Entities:

- **User**: Initiates the process by providing video/text input and views the final sentiment analysis results.
- **YouTube API**: Acts as an external data source, providing the raw comments based on user input.
- **Amazon Comprehend**: A cloud-based NLP service that performs sentiment analysis

on the fetched comments.

Data Stores:

- **Result**: Stores the analyzed sentiment results for display or further use.

Data Flows:

- Comments flow from YouTube API to the system.
- Processed sentiment data flows from Amazon Comprehend to the system.
- Final sentiment results are presented to the user.

4. 2. 2. DFD LEVEL – 1

Level 1: A more detailed view of a single process, showing its sub-processes and data flows. The below Fig No. 4. 2. 2 shows a high-level overview of the Multilingual sentiment analysis.

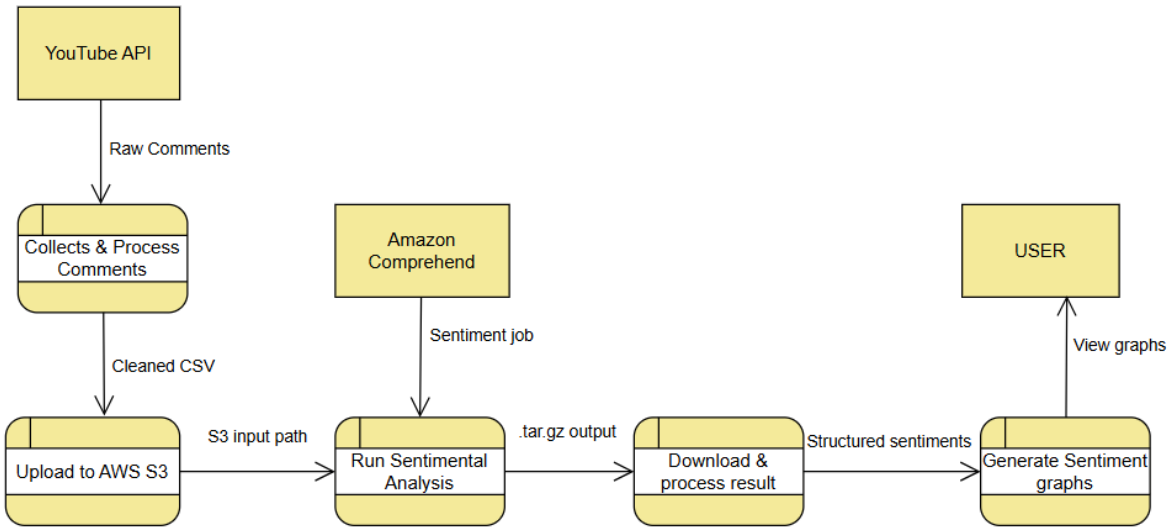


Fig 4. 2. 2. DFD LEVEL – 1

This Level-1 Data Flow Diagram provides a detailed view of the internal data flow and processing steps involved in analyzing sentiments from YouTube comments using Amazon Comprehend and visualizing the results for users.

Processes:

1. **Collects & Process Comments**: Raw comments are retrieved from the **YouTube API**. The system cleans and formats these comments into a structured CSV format.
2. **Upload to AWS S3**: The cleaned CSV is uploaded to **Amazon S3**, enabling further processing by AWS services.
3. **Run Sentiment Analysis**: **Amazon Comprehend** consumes the CSV via a sentiment analysis job. The output is returned as a .tar.gz file containing sentiment-labeled data.

4. **Download & Process Results**: The system downloads the .tar.gz output and extracts structured sentiment data for reporting.
5. **Generate Sentiment Graphs**: Structured sentiments are processed and visualized as graphs or charts. The **User** views the final results as graphical representations of the sentiment analysis.

External Entities:

- **YouTube API**: Supplies the raw user comments based on video IDs or keywords.
- **Amazon Comprehend**: Performs machine learning-based sentiment analysis on the textual data.
- **User**: Initiates the process and views the final sentiment analysis graphs.

Data Stores & Flow:

- **CSV files** are used for data exchange between components.
- **S3 Input Path** and **.tar.gz Output** manage data between AWS services.
- Final output is **Structured Sentiments**, which is visualized and presented to the user.

4. 2. 3. DFD LEVEL – 2

It provides a detailed breakdown of the Multilinguistic Sentiment Analysis System, showing how data flows through different processing stages. The below Fig No. 4. 2. 3 shows a high-level overview of the Multilingual sentiment analysis.

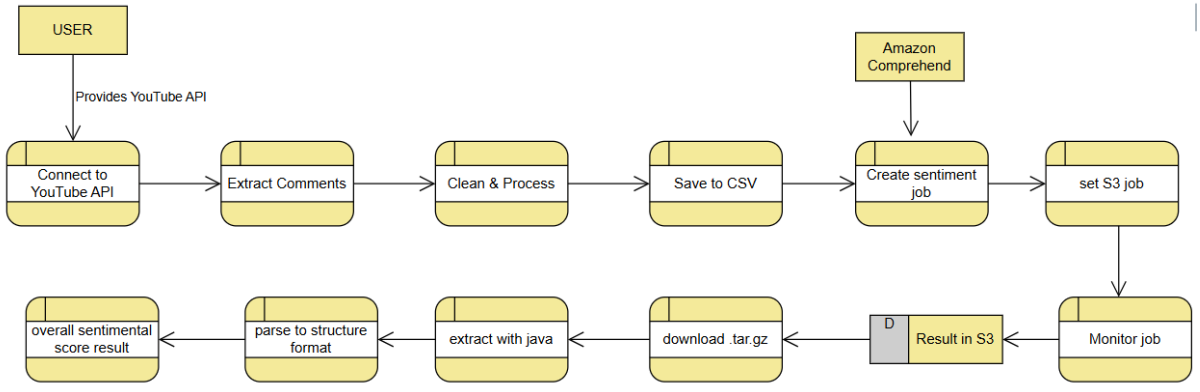


Fig 4. 2. 2. DFD LEVEL – 2

This diagram represents a **Level 2 Data Flow Diagram (DFD)**, providing a more detailed view of the system originally shown as a single process in the Level 1 DFD. Here, we break down the main system into **sub-processes**.

Detailed Process Steps in DFD-2:

1. **Input Text:** The user enters text in any language.
2. **Detect Language:** The system detects the language of the provided text.
3. **Checking Translation (If needed):** If the detected language is not English, the system decides whether translation is required.
4. **Translate to English:** If translation is needed, the system translates the text into English using a translation service (e.g., Amazon Translate).
5. **Analyze Sentiment:** The system processes the translated text to determine the sentiment (Positive, Negative, or Neutral) using NLP techniques.
6. **Classify Sentiment:** Based on the analysis, the sentiment is categorized into **Positive, Negative, or Neutral** for better understanding.
7. **Store in Database:** The system saves **input text, detected language, translation status, and sentiment results** for future reference and analysis.
8. **Show Sentiment:** The analyzed sentiment is displayed to the user.

4. 3. UML DIAGRAMS

4. 3. 1. USECASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. The below Fig No. 4. 3. 1 can portray the different types of users of a system and the various ways that they interact with the system.

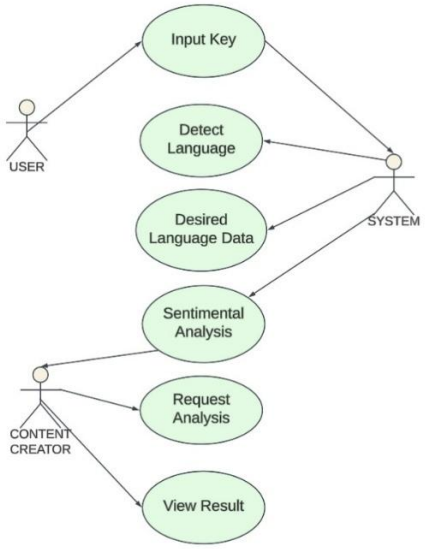


Fig 4. 3. 1. USE CASE DIAGRAM

1. Actors

- **User:** The user interacts with the system by inputting text that needs to be analysed for sentiment.
- **Content Creator:** The content creator initiates the request for sentiment analysis and views the results.
- **System:** The Multilingual Sentiment Analysis System processes the input data and performs various tasks including language detection and sentiment analysis.

2. Use Cases

- **Input Key:** The user provides the initial text input to the system.
- **Detect Language:** The system identifies the language of the entered text.
- **Desired Language Data:** The system obtains necessary language-specific data for further processing.
- **Sentimental Analysis:** The system performs sentiment analysis on the input text to determine the emotional tone.
- **Request Analysis:** The content creator requests the system to perform sentiment analysis on the given data.
- **View Result:** The content creator views the output of the sentiment analysis.

4. 3. 2. CLASS DIAGRAM

The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematics of the application and for detailed modeling translating the models into programming code. In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

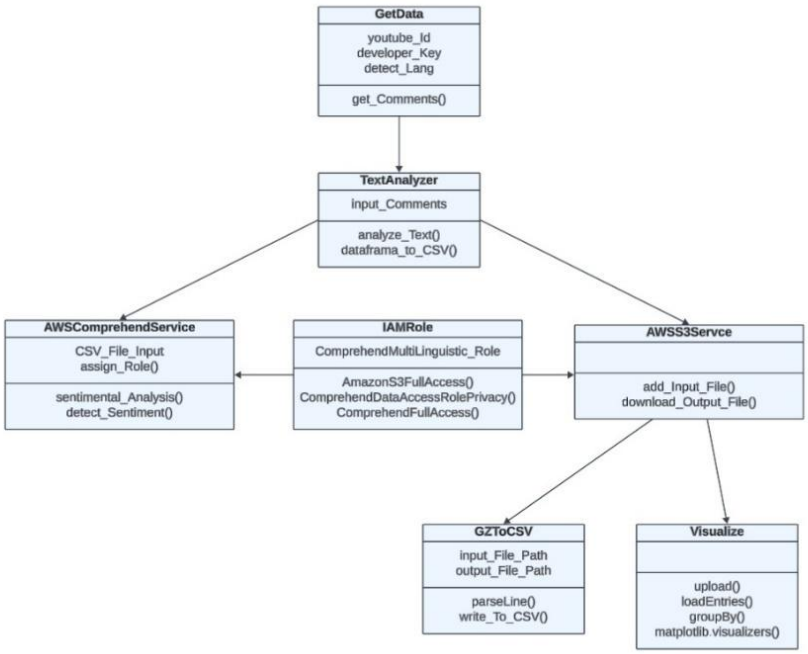


Fig 4. 3. 2. CLASS DIAGRAM

1. **GetData**: Fetches YouTube comments and detects language using get_Comments().
2. **TextAnalyzer**: Central class that processes comments via analyze_Text() and converts data to CSV.
3. **AWSComprehendService**: Performs sentiment analysis on comments using AWS Comprehend.
4. **IAM Role**: Manages AWS roles and permissions for accessing services like S3 and Comprehend.
5. **AWS S3 Service**: Handles file uploads/downloads to/from AWS S3.
6. **Gz To CSV & Visualize**: Converts GZ data to CSV and visualizes results using matplotlib.

4. 3. 3 SEQUENCE DIAGRAM

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in a time sequence. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

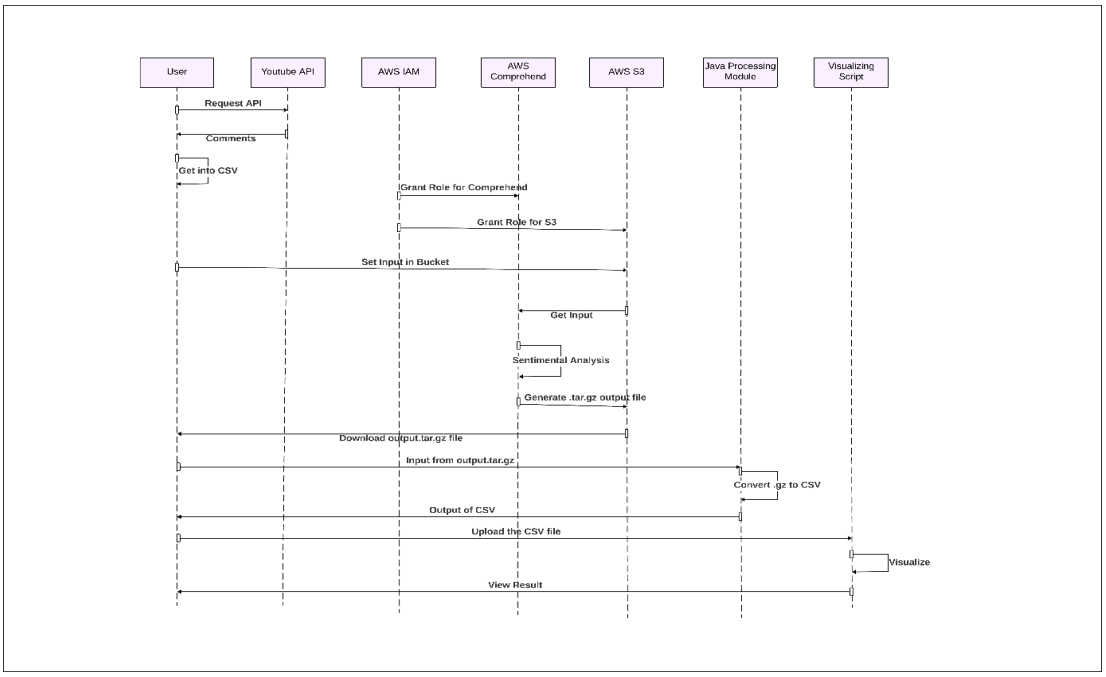


Fig 4. 3. 3. SEQUENCE DIAGRAM

1. **User**: Initiates the entire workflow by requesting data (e.g., YouTube comments). Views the final sentiment analysis results.
2. **YouTube API**: Responds to the user's request by providing video comments. These comments are saved into a CSV file for further processing.
3. **AWS IAM (Identity and Access Management)**: Grants necessary roles for services to access AWS resources. Assigns roles for AWS Comprehend and AWS S3.
4. **AWS Comprehend**: Performs the **sentiment analysis** on the input text. Takes input from the S3 bucket and generates sentiment labelled data.
5. **AWS S3 (Simple Storage Service)**: Stores input files and output results. Receives the input CSV and provides output in a .tar.gz file format.
6. **Java Processing Module**: Extracts and processes the .tar.gz output file. Converts the data into a structured CSV format.
7. **Visualizing Script**: Loads and processes the final CSV. Displays the sentiment analysis visually.

4. 3. 4. COMPONENT DIAGRAM

The below Fig No. 4. 3. 4 depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

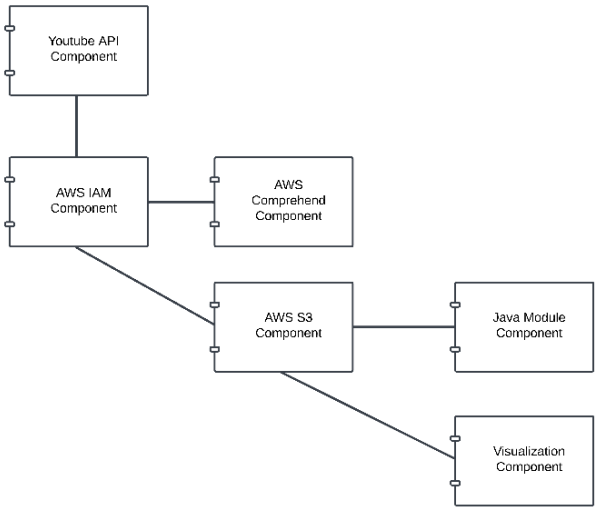


Fig 4. 3. 4. COMPONENT DIAGRAM

Components and Their Roles

1. **YouTube API Component**: Provides access to YouTube video comments. Serves as the data ingestion layer where comments are fetched for sentiment analysis.
2. **AWS IAM Component (Identity and Access Management)**: Manages roles and permissions. Grants secure access rights to AWS Comprehend and AWS S3 services, ensuring that data can flow between components securely.
3. **AWS Comprehend Component**: Performs **sentiment analysis** on textual data. Processes comments (possibly translated and stored in a CSV format). Produces sentiment scores (Positive, Negative, Neutral, Mixed).
4. **AWS S3 Component (Simple Storage Service):** Cloud storage service. Stores input CSVs for processing by AWS Comprehend. Receives output tar.gz or CSV files from sentiment analysis. Shares results with downstream components.
5. **Java Module Component**: Handles post-processing of AWS Comprehend output. Converts output files (e.g., .tar.gz) into usable CSV format, performs cleanup, or enhances formatting.
6. **Visualization Component**: Displays the final sentiment analysis results in a graphical format (charts, graphs, dashboards). Receives cleaned and processed CSV files from the Java Module or directly from AWS S3.

4. 3. 5. ACTIVITY DIAGRAM

The activity diagram is another important diagram in UML to describe the dynamic aspects of the system. It is a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. It shows the process of a user uploading and processing documents in a system as shown in Fig No. 4. 3. 5.

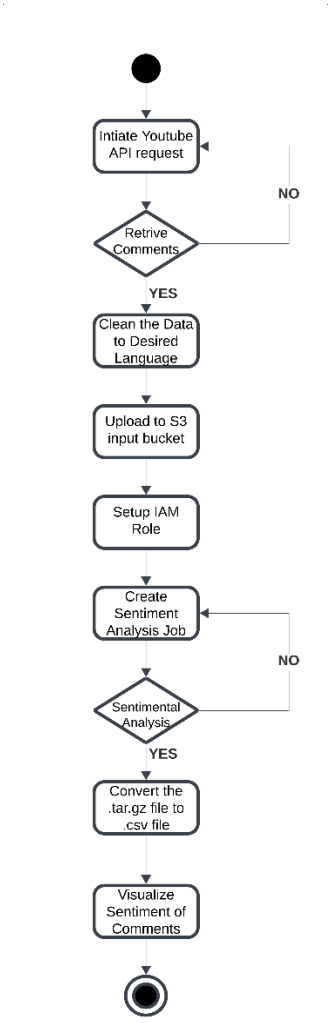


Fig 4. 3. 5. ACTIVITY DIAGRAM

This diagram depicts the automated process of retrieving YouTube comments and analyzing their sentiment using AWS services.

1. **Initiate YouTube API Request**: The process starts by sending a request to the YouTube API to fetch comments from a specific video or channel.
2. **Retrieve Comments**: The system checks whether comments are successfully retrieved.
    - If **NO**, the process loops back to the API request.
    - If **YES**, proceeds to the next step.

3. **Clean the Data to Desired Language**: Irrelevant content is removed. Comments are filtered and possibly translated into a target language (likely English) for compatibility with sentiment analysis tools.
4. **Upload to S3 Input Bucket**: The cleaned and prepared comment data is uploaded to an **AWS S3 bucket**, acting as temporary storage for further processing.
5. **Setup IAM Role**: An **IAM Role** is configured to authorize AWS services (e.g., Comprehend) to access the uploaded data securely.
6. **Create Sentiment Analysis Job**: A job is created using **AWS Comprehend** to start sentiment analysis on the input file stored in S3.
7. **Sentiment Analysis:**
    - If sentiment analysis fails (NO), the system loops back to retry creating the job.
    - If successful (YES), proceeds to data processing.
8. **Convert the .tar.gz File to .csv File**: The output from AWS Comprehend (typically a compressed .tar.gz archive) is extracted and converted into a structured CSV format for easier handling and visualization.
9. **Visualize Sentiment of Comments**: Final output is visualized, likely via graphs, charts, or dashboards, to provide insights into the sentiment distribution of YouTube comments.

4. 3. 6. DEPLOYMENT DIAGRAM

A deployment diagram in the Unified Modeling Language models the physical deployment of artifacts on nodes which is shown in Fig No. 4. 3. 7.

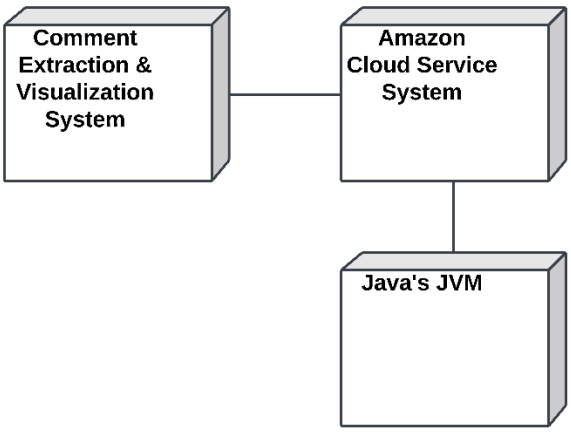


Fig 4. 3. 7. DEPLOYMENT DIAGRAM

This diagram represents a simplified Deployment Diagram in UML (Unified Modeling Language), focusing on the hardware and software components of a bot system and their deployment relationships.

1. **3D Box Notation:**
    - The boxes (nodes) have a 3D appearance, which is typical in UML
    - Deployment Diagrams to represent hardware or execution environments.
2. **Nodes Representing Systems:**
    - Comment Extraction & Visualization System: Represents a client-side or front-end node.
    - Amazon Cloud Service System: A cloud-hosted environment handling services like AWS S3 and Comprehend.
    - Java's JVM: Indicates a runtime environment (on the cloud or backend) for executing Java modules.
3. **Connections (Lines):**
    - The lines show communication or deployment relationships between the nodes.

CHAPTER 5
HARDWARE AND SOFTWARE REQUIREMENTS

5. 1. SOFTWARE REQUIREMENTS:

- **Programming Language:** Python, Java
- **Frameworks & APIs:** Flask or FastAPI
- **AWS Services :**
  - AWS Comprehend
  - IAM
  - Files stored likely via S3 Buckets
- **Libraries Used :** boto3, nltk, transformers, pandas, numpy, scikit-learn, os, io
- **IDE:** VS Code, Google Colab

5. 2. HARWARE REQUIREMENTS

- **Processor:** Intel i5 / AMD Ryzen 5
- **RAM:** 8GB (min)
- **Storage (Hard Disk):** 128 GB
- **Power Supply:** Uninterrupted Power Supply (UPS) for continuous operation
- **Network Connectivity:** High-speed Internet Connection (for accessing AWS Comprehend, and storing results on S3)

CHAPTER 6
MODULES

The Multilingual Sentiment Analysis System consists of various user and system modules that work together to process user input, analyze sentiments, and store results efficiently using AWS services. These are the modules used in our project:

6. 1. System Module:

The **System Module** manages the core logic of data processing, sentiment analysis, and results handling. It consists of the following submodules:

1) **Data Collection & Preprocessing Module**
    - Uses the **YouTube Data API** to collect comments from various videos in different languages.
    - Cleans and preprocesses the text (removal of special characters, emojis, etc.).
    - Stores the processed data in **CSV format** for further use.
2) **AWS Configuration & Setup Module**
    - Creates an **AWS Root Account** and configures the **IAM role** with required permissions:
        - ComprehendFullAccess
        - ComprehendDataAccessRolePolicy
        - AmazonS3FullAccess
    - Sets the **AWS region** to **US East (N. Virginia)**.
    - Creates an **S3 bucket** with structured folders:
        - input/ – For uploading pre-processed CSV files.
        - output/ – For storing analysis results.
3) **Sentiment Analysis Module (AWS Comprehend)**
    - Configures and initiates a **Batch Sentiment Detection Job** in **Amazon Comprehend**.
    - Inputs:
        - Job Name
        - Input and Output S3 folder paths
        - Language specification
    - AWS Comprehend processes the data and writes the results to the output folder as a .tar.gz archive.

4) **Result Categorization & Extraction Module (Java)**
    - A **Java program** is used to:
        - Extract the contents of the .tar.gz file.
        - Parse and categorize the results.
        - Convert them into a structured format suitable for score analysis (e.g., another CSV or JSON).
5) **Sentiment Score Visualization Module (Python)**
    - A **Python script** reads the categorized result file.
    - Calculates **sentiment scores** (count or percentage of positive, negative, neutral, mixed).
    - Generates visualizations using libraries like **Matplotlib**, **Seaborn**, or **Plotly**.

6. 2. User Interaction Module:

The **User Interaction Module** facilitates how users interact with the system:

1) **Comment Input Interface**
    - Not a GUI in this case but handled programmatically via Python scripts using the YouTube API.
2) **Batch Job Monitoring**
    - The user monitors job completion via the **AWS Comprehend Console**.
    - Downloads the result .tar.gz manually from the **S3 output folder**.
3) **Result Display**
    - Final sentiment results are displayed via **charts or printed outputs** after being processed by the Python visualization code.

CHAPTER 7
IMPLEMENTATION

1. Collecting and downloading YouTube comments into a CSV file.

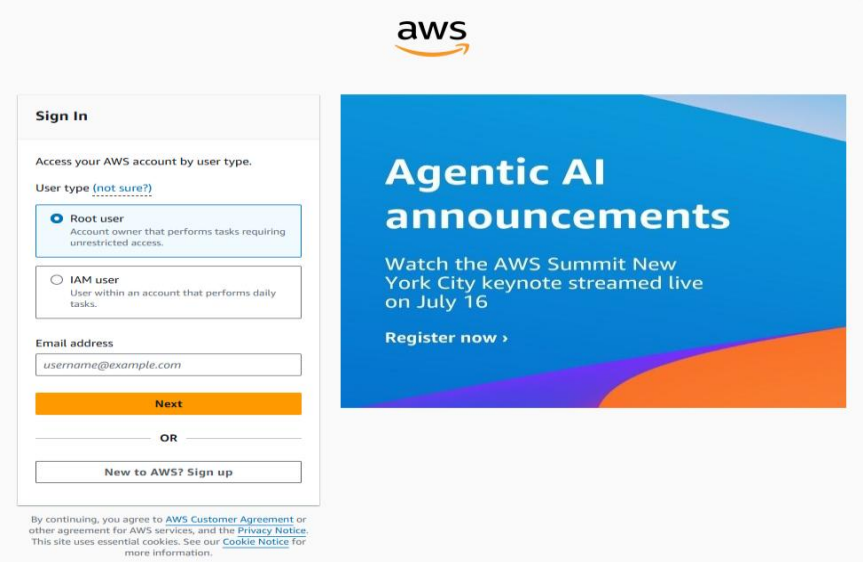
This Python script extracts and filters YouTube video comments using the YouTube Data API. It connects to a specified video using an API key and retrieves up to 1000 top-level comments. Each comment is cleaned to remove HTML tags and excess whitespace, and then processed with langdetect to identify its language. Only comments written in a specified target language (in this case, Japanese) are retained. For each valid comment, the script records its text and like count. Finally, the filtered data is saved to a CSV file (youtube_comments.csv) and downloaded for further use or analysis.

```
!pip install langdetect
!pip install htmls

import googleapiclient.discovery
import pandas as pd
import re
import html
from langdetect import detect, LangDetectException
def clean_text(text):
  text = html.unescape(text)
  text = re.sub(r'<.*?>', '', text)
  text = re.sub(r'\s+', ' ', text).strip()
  return text
api_service_name = "youtube"
api_version = "v3"
DEVELOPER_KEY = "AIzaSyD-A3CKeJTC0mziycvfm6swDdp28qNhX4k"
youtube = googleapiclient.discovery.build(
   api_service_name, api_version, developerKey=DEVELOPER_KEY
)
video_id = "kxs9Su_mbpU"
target_language = "ja"
max_comments = 1000
```

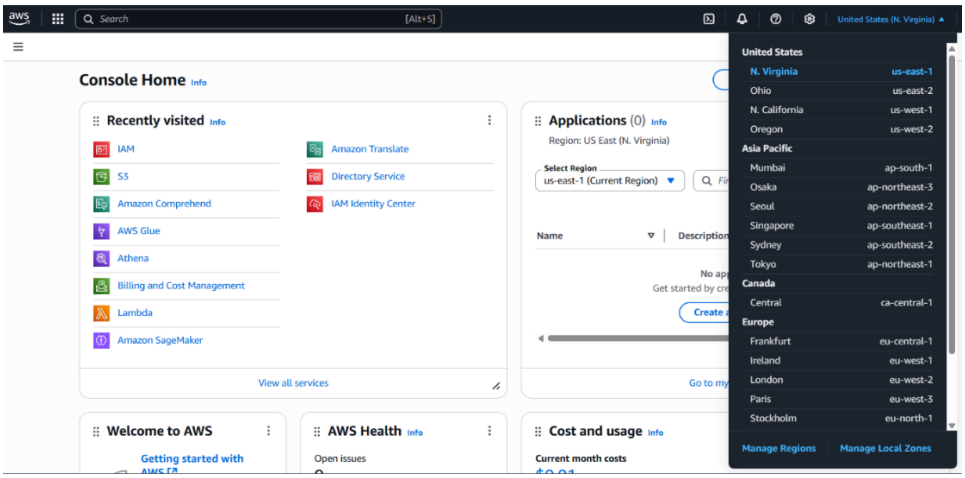
```
comments = []
next_page_token = None
while len(comments) < max_comments:
    request = youtube.commentThreads().list(
        part="snippet",
        videoId=video_id,
        maxResults=1000,
        pageToken=next_page_token
    )
    response = request.execute()
    for item in response['items']:
        snippet = item['snippet']['topLevelComment']['snippet']
        text = clean_text(snippet['textDisplay'])
        like_count = snippet['likeCount']
        if text:
            try:
                language = detect(text)
            except LangDetectException:
                language = "unknown"
            if language == target_language:
                comments.append([like_count, text])
            if len(comments) >= max_comments:
                break
    next_page_token = response.get('nextPageToken')
    if not next_page_token:
        break
df = pd.DataFrame(comments, columns=['like_count', 'text'])
df

df.to_csv("youtube_comments.csv", index=False, encoding='utf-8-sig')
from google.colab import files
files.download("youtube_comments.csv")
```

2. Usage of AWS services and the associated processes

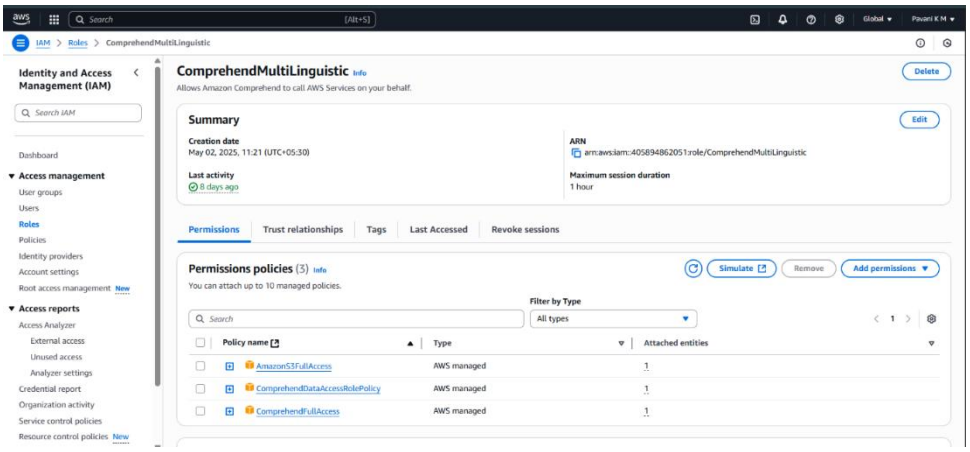
- **Creating an AWS root user account and logging in using the login credentials**



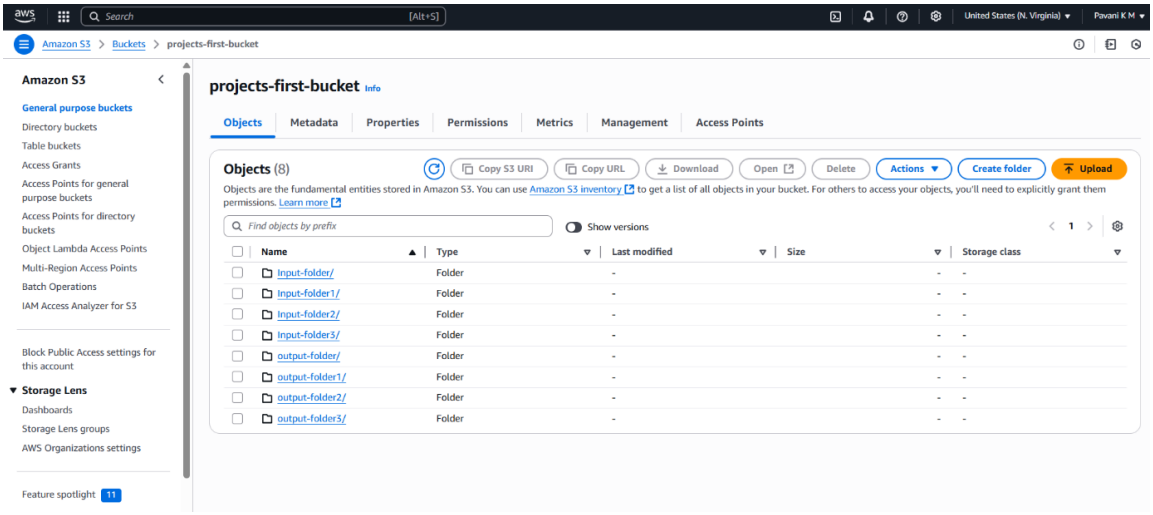
- **Setting the region to N. Virginia to use the services**



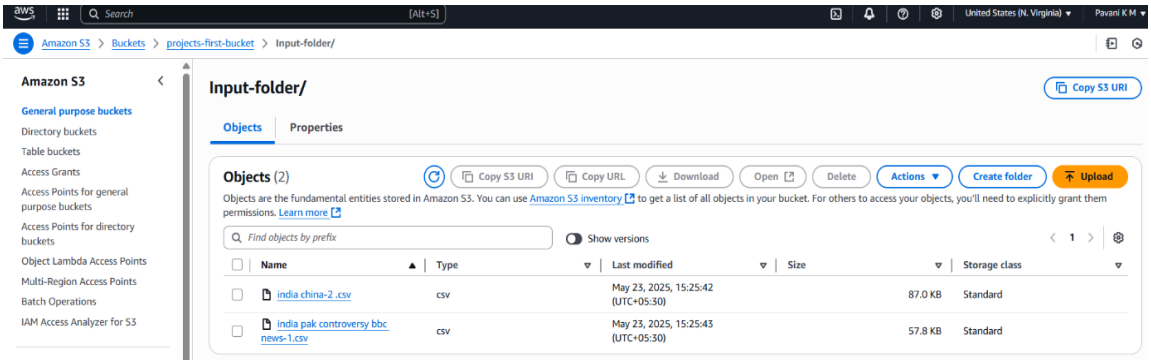
- **Go to Services, open IAM, and create an IAM role to access services like Comprehend, S3 buckets, and more**



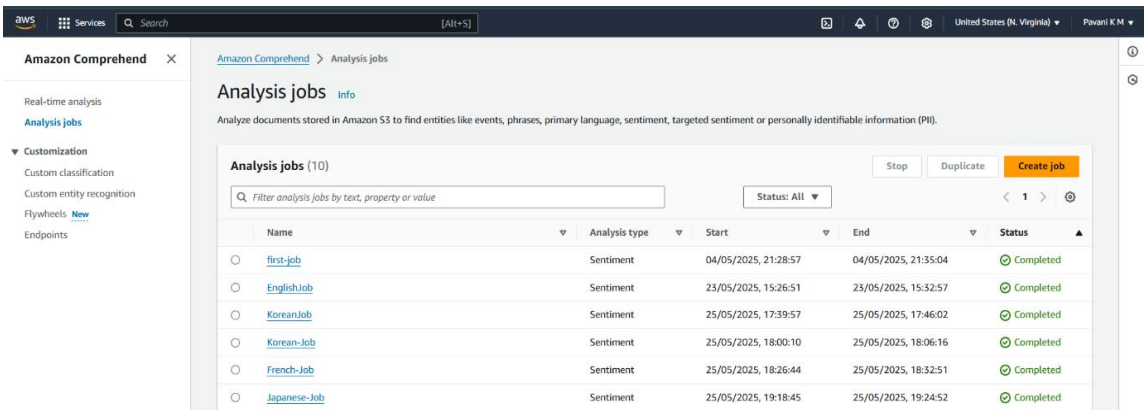
- **Open Amazon S3, create a project bucket, and add folders for input and output storage**



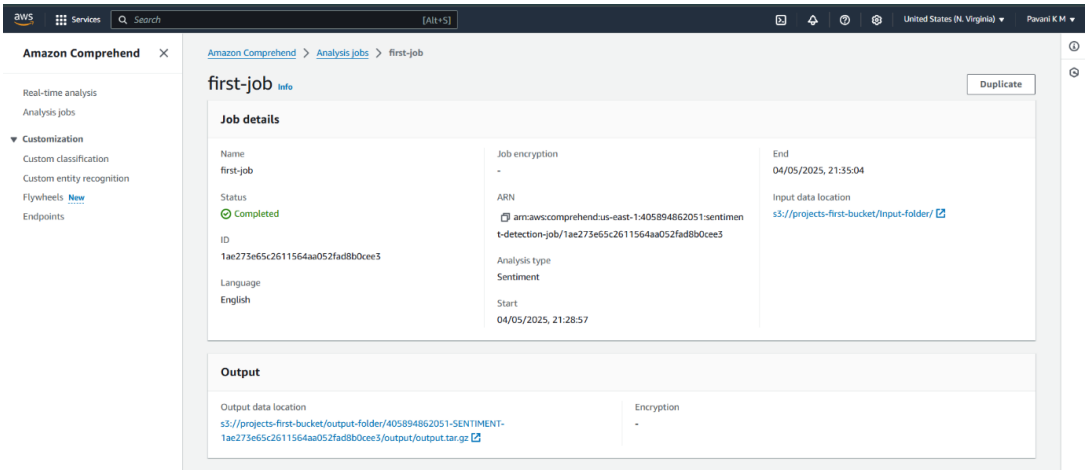
- **Upload the extracted YouTube comments CSV files to the input folder**



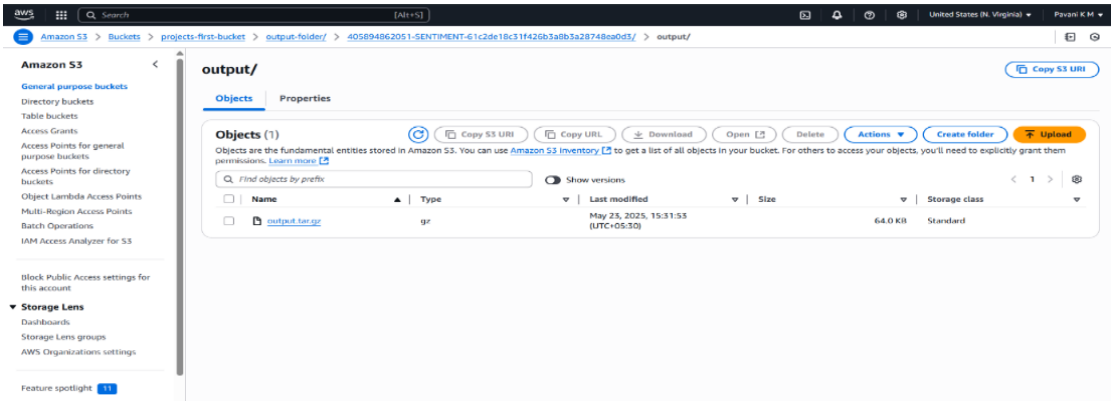
- **Create a job in Amazon Comprehend, selecting sentiment analysis and specifying the input language**



- **The job is created successfully, and the output location is displayed to track the .tar.gz file**



- **After job creation, the output .tar.gz file is generated in the output folder and downloaded**



3. Converting the gz formate file to csv

This Java program reads a text file (output_ja) that contains sentiment analysis results in a structured but uncompressed .gz output format, parses each line to extract relevant fields, and writes the processed data into a structured CSV file (output_ja.csv). Each line is assumed to contain key-value pairs representing attributes such as file name, line number, sentiment category, and confidence scores (Mixed, Negative, Neutral, Positive). The parseLine method handles data cleaning and extraction, converting appropriate values to their respective types. The writeToCSV method then formats and saves the cleaned data into CSV format for easier access and further analysis or visualization.

```
import java.io.*;
import java.util.*;
public class Main1 {
```

```
public static void main(String[] args) {
    String inputFilePath = "output_ja";
    String outputFilePath = "output_ja.csv";
    List<Map<String, Object>> table = new ArrayList<>();
    try (BufferedReader br = new BufferedReader(new FileReader(inputFilePath))) {
        String line;
        while ((line = br.readLine()) != null) {
            Map<String, Object> row = parseLine(line);
            if (row != null) {
                table.add(row);
            }
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
    writeToCSV(outputFilePath, table);
    System.out.println("Data successfully written to " + outputFilePath);
}
private static Map<String, Object> parseLine(String line) {
    Map<String, Object> row = new HashMap<>();
    try {
        line = line.replaceAll("[{}\"]", "");
        String[] parts = line.split(", ");
        for (String part : parts) {
            String[] keyValue = part.split(": ");
            if (keyValue.length == 2) {
                String key = keyValue[0].trim();
                String value = keyValue[1].trim();
                switch (key) {
                    case "File":
                        row.put("File", value);
                        break;
                    case "Line":
                        row.put("Line", Integer.parseInt(value));
```

```
                    break;
                case "Sentiment":
                    row.put("Sentiment", value);
                    break;
                case "Mixed":
                case "Negative":
                case "Neutral":
                case "Positive":
                    row.put(key, Double.parseDouble(value));
                    break;
            }
        }
    }
    } catch (Exception e) {
        System.err.println("Error parsing line: " + line);
        return null;
    }
    return row;
  }
  private static void writeToCSV(String filePath, List<Map<String, Object>> table) {
    try (PrintWriter pw = new PrintWriter(new FileWriter(filePath))) {
        pw.println("File,Line,Sentiment,Mixed,Negative,Neutral,Positive");
        for (Map<String, Object> row : table) {
            pw.printf("%s,%d,%s,%.6f,%.6f,%.6f,%.6f%n",
                    row.get("File"), row.get("Line"), row.get("Sentiment"),
                    row.get("Mixed"), row.get("Negative"), row.get("Neutral"),
row.get("Positive"));
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
}
```

4. Checking the overall sentimental score by uploading the csv downloaded files

This Python code, intended for use in Google Colab, uploads a CSV file (typically containing sentiment analysis results), reads it into a pandas DataFrame, and calculates a summary of sentiment scores. It first uses the files.upload() method to let the user upload a CSV file, which is then read into memory. The code groups the data by the "Sentiment" column and sums the corresponding sentiment confidence scores Negative, Neutral, Positive, and Mixed to give an overall sentiment distribution. It also calculates the percentage of each sentiment category based on the total number of entries, providing a clear picture of the dominant sentiments in the dataset.

```
from google.colab import files
uploaded = files.upload()

import pandas as pd
import io
for fn in uploaded.keys():
  df = pd.read_csv(io.StringIO(uploaded[fn].decode('utf-8')))
df

sentiment_summary = df.groupby('Sentiment')[['Negative', 'Neutral', 'Positive',
'Mixed']].sum()
sentiment_summary

total_rows = len(df)
sentiment_percentages = df['Sentiment'].value_counts() / total_rows * 100
sentiment_percentages
```

CHAPTER 8
TESTING

1. Unit Testing

Test individual components of the project to ensure they work as expected.

- **YouTube API Extraction**: Verify comments are correctly fetched for different video keys. Check language filtering is applied accurately.
- **Data Preprocessing**: Confirm removal of unwanted characters, emojis, and links. Validate the final cleaned output before CSV export.
- **CSV File Generation**: Ensure the CSV contains expected columns (e.g., like_count, comment_text). Check the integrity of data (no missing values or encoding issues).

2. Integration Testing

Test how the components work together in the pipeline.

- **CSV Upload to S3**: Verify the file is correctly uploaded to the right input folder. Confirm file accessibility by AWS Comprehend.
- **IAM Role Permissions**: Check if the role grants access to both Comprehend and S3 services. Ensure no permission errors occur during job execution.

3. Functional Testing

Test the core functionality of the system.

- **Amazon Comprehend Job Creation**: Validate correct job creation with required input and output S3 paths. Ensure correct language and sentiment analysis mode selection.
- **Job Output**: Check the .tar.gz file is created and saved in the output folder. Verify the contents of the file match the input and expected sentiment structure.

4. System Testing

Test the full system end-to-end from comment collection to sentiment visualization.

- **Pipeline Execution**: Run the entire workflow and check for smooth data flow. Validate the final output aligns with input comment sentiments.

5. Post-Processing Testing

Test the handling of output data after Comprehend.

- **Java Program for Output Extraction**: Confirm correct decompression of .tar.gz. Verify extraction of sentiment data into readable format.
- **Sentiment Score Script & Visualization**: Ensure accurate sentiment categorization (Positive, Negative, Neutral, Mixed). Check that graphs accurately reflect the sentiment distribution.

6. Error Handling & Edge Case Testing

- **Invalid/Empty YouTube video key**: Confirm system gracefully handles missing or invalid inputs.
- **Unsupported Language Input**: Ensure error is captured and reported clearly.
- **Missing S3 Paths or IAM Role Errors**: Check the system fails gracefully and logs proper errors for debugging.

8.1 TEST CASES

Testcase ID	Input (YouTube URL)	Expected Output	Overall Sentiment Score
01	https://www.youtube.com/watch?v=Mc9Ed8AnL4s https://www.youtube.com/watch?v=8wcUkP--ZO0	It gives the overall sentiment score of the YouTube comments of English(en), categorized into Positive, Negative, Neutral, and Mixed.	count Sentiment NEUTRAL 55.357143 NEGATIVE 34.895833 POSITIVE 6.994048 MIXED 2.752976
02	https://www.youtube.com/watch?v=kTlv5_Bs8aw	It gives the overall sentiment score of the YouTube comments of Korean(ko), categorized into Positive, Negative, Neutral, and Mixed.	count Sentiment NEUTRAL 50.310559 POSITIVE 41.304348 NEGATIVE 8.074534 MIXED 0.310559
03	https://www.youtube.com/watch?v=-an9d5V7Dvw	It gives the overall sentiment score of the YouTube comments of French(fr), categorized into Positive, Negative, Neutral, and Mixed.	count Sentiment NEUTRAL 72.327672 POSITIVE 17.082917 NEGATIVE 9.990010 MIXED 0.599401
04	https://www.youtube.com/watch?v=kxs9Su_mbpU	It gives the overall sentiment score of the YouTube comments of Japanese(ja), categorized into Positive, Negative, Neutral, and Mixed.	count Sentiment POSITIVE 42.157842 NEUTRAL 27.072927 NEGATIVE 21.078921 MIXED 9.690310

Table 8. 1 TESTCASES

CHAPTER 9

OUTPUT SCREENS

```
[1] from google.colab import files

    uploaded = files.upload()
```

```
Choose files  output_ja.csv
• output_ja.csv(text/csv) - 75458 bytes, last modified: 25/05/2025 - 100% done
Saving output_ja.csv to output_ja (2).csv
```

```
[2] import pandas as pd
    import io

    for fn in uploaded.keys():
      df = pd.read_csv(io.StringIO(uploaded[fn].decode('utf-8')))
    df
```

	File	Line	Sentiment	Mixed	Negative	Neutral	Positive
0	youtube_japanese_comments.csv	1	POSITIVE	NaN	0.157007	0.135484	0.706625
1	youtube_japanese_comments.csv	3	NEGATIVE	NaN	0.586761	0.042074	0.369959
2	youtube_japanese_comments.csv	4	NEGATIVE	NaN	0.608438	0.017186	0.233538
3	youtube_japanese_comments.csv	9	NEUTRAL	NaN	0.018489	0.979674	0.001817
4	youtube_japanese_comments.csv	10	NEUTRAL	NaN	0.282121	0.709946	0.006092
...	...	...	...	...	...	...	...
996	youtube_japanese_comments.csv	966	NEUTRAL	NaN	0.284571	0.705924	0.009387
997	youtube_japanese_comments.csv	973	NEUTRAL	NaN	0.086151	0.911318	0.002463
998	youtube_japanese_comments.csv	978	NEUTRAL	NaN	0.100734	0.653138	0.245506
999	youtube_japanese_comments.csv	989	NEGATIVE	NaN	0.774298	0.222928	0.002761
1000	youtube_japanese_comments.csv	991	NEGATIVE	NaN	0.986992	0.011717	0.001267

1001 rows × 7 columns

Fig 9.1 Display of Uploaded YouTube Comments Sentiment Data

```
sentiment_summary = df.groupby('Sentiment')[['Negative', 'Neutral', 'Positive', 'Mixed']].sum()
sentiment_summary
```

Sentiment	Negative	Neutral	Positive	Mixed
MIXED	6.732740	3.492800	10.589243	0.0
NEGATIVE	162.048635	31.521650	10.974659	0.0
NEUTRAL	28.968428	214.811033	25.380068	0.0
POSITIVE	11.425240	26.583849	374.492321	0.0

Next steps: Generate code with sentiment_summary | View recommended plots | New interactive sheet

```
[4] total_rows = len(df)
    sentiment_percentages = df['Sentiment'].value_counts() / total_rows * 100
    sentiment_percentages
```

Sentiment	count
POSITIVE	42.157842
NEUTRAL	27.072927
NEGATIVE	21.078921
MIXED	9.690310

dtype: float64

Fig 9.2 sentiment scores and percentage distribution of YouTube comments

CHAPTER 10
DEPLOYMENT

1. Prerequisites & Setup

Languages & Tools Used:

- **Python** (for data collection, cleaning, processing, and visualization)
- **Java** (for processing AWS Comprehend output)
- **AWS Services** (S3, IAM, Comprehend)
- **Google Colab** or **Jupyter Notebook** (to run Python code)

2. Installing Required Software

Python Installation: If not using Google Colab:

- Download Python from: https://www.python.org/downloads/
- Install required Python packages: pip install pandas boto3 google-api-python-client

Java Installation:

- Download Java JDK: https://www.oracle.com/java/technologies/javase-downloads.html

3. Steps to Run the Project

Step 1: Extract YouTube Comments

- Use the YouTube API to extract comments.
- You can do this in Google Colab using the provided Python script.
- Enter the video key and language code to filter comments.
- Save the cleaned and processed comments into a .csv file.

Step 2: Setup AWS Account

- Create a **root AWS account** at https://aws.amazon.com/console/
- Set your region to **US East (N. Virginia)**.

Step 3: Configure IAM Role

- Go to **Services → IAM**.
- Create a new role with these policies:
  - ComprehendFullAccess
  - ComprehendDataAccessRolePolicy
  - AmazonS3FullAccess

Step 4: Create S3 Bucket

- Go to **Services → S3**.
- Create a new bucket (e.g., youtube-sentiment-bucket).
- Inside the bucket, create two folders:
  - input (to store the uploaded .csv)
  - output (for AWS Comprehend job results)

Step 5: Upload Comments CSV

- Upload your processed .csv file (e.g., youtube_comments.csv) into the input folder of your S3 bucket.

Step 6: Create Sentiment Analysis Job

- Go to **Amazon Comprehend**.
- Create a new job:
  - Select **Sentiment Analysis**
  - Choose the input language (e.g., ja for Japanese)
  - Set S3 input and output folder paths
  - Select the IAM role created earlier

Step 7: Download & Process Output

- Once the job completes, go to the output folder in S3.
- Download the .tar.gz file.
- Use a **Java program** (provided separately) to extract and process the results into a .csv or readable format.

4. Analyze and Visualize Sentiment Data

- Open the processed result file in **Google Colab** or any Python environment.
- Use provided scripts to:
  - Load the .csv
  - Group sentiments
  - Calculate percentages
  - Generate graphs (bar charts, pie charts) showing Positive, Negative, Neutral, and Mixed sentiments

CHAPTER 11
INTEGRATION AND EXPERIMENTAL RESULTS

11.1 Integration

The project integrates multiple technologies and services to form a seamless pipeline for multilingual YouTube comment sentiment analysis:

1. **YouTube API Integration**:
   Python is used to interact with the YouTube Data API. Comments are extracted using a video key and filtered by language using a language code.
2. **Data Preprocessing in Python**:
   Extracted comments are cleaned and preprocessed to remove noise (e.g., emojis, special characters), then saved in a CSV format using pandas.
3. **Cloud Integration with AWS**:
   - Amazon **S3** is used for storing input (preprocessed CSV) and output (Comprehend results).
   - Amazon **Comprehend** is used to perform sentiment analysis on uploaded comments.
   - AWS **IAM** is used to manage access control with the required permissions for S3 and Comprehend services.
4. **Java Integration**:
   A Java program processes the .tar.gz output from Amazon Comprehend to extract and format the sentiment results into a readable .csv format.
5. **Visualization in Python**:
   Python scripts are used to read the final CSV, compute sentiment distribution, and generate visual graphs (bar/pie charts) to represent Positive, Negative, Neutral, and Mixed sentiments.

11. 2 Experimental Results

The sentiment analysis was performed on a dataset of **1001 YouTube comments** in Japanese. The results obtained after processing are:

➤ **Sentiment Scores (Summed by Type)**

Sentiment	Positive	Negative	Neutral	Mixed
POSITIVE	6.732740	3.492800	10.589243	0.0
NEGATIVE	162.048635	31.521650	10.974659	0.0
NEUTRAL	28.968428	214.811033	25.380068	0.0
MIXED	11.425240	26.583849	374.492321	0.0

➤ **Sentiment Distribution (Percentage)**

Sentiment	Percentage
POSITIVE	42.157842
NEUTRAL	27.072927
NEGATIVE	21.078921
MIXED	9.690310

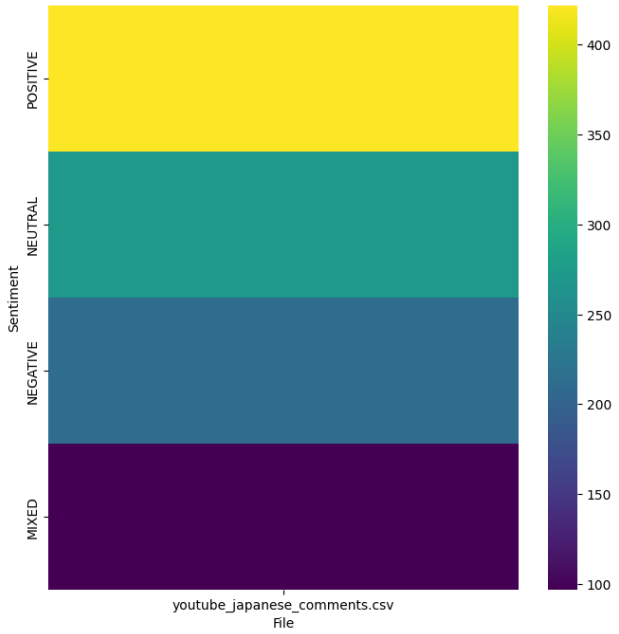


Fig 11.2 Heatmap of Sentiment Distribution

CHAPTER 12

PERFORMANCE EVALUATION

The project demonstrates a successful end-to-end pipeline for performing sentiment analysis on YouTube comments using **Amazon Comprehend** and **AWS services**. The primary objective is not to train a model but to use a pre-built NLP service to analyze large volumes of multilingual comments and extract meaningful sentiment insights.

Key Strengths:

- **Accurate Sentiment Categorization**: Amazon Comprehend effectively classifies comments into **Positive**, **Negative**, **Neutral**, and **Mixed** sentiments.
- **Multilingual Support**: By specifying language codes, the system extracts and analyzes comments in the targeted language, ensuring relevance and consistency.
- **Automation**: The entire process—from extracting comments using the YouTube API to uploading and processing them via AWS is automated and efficient.
- **Insightful Output**: The sentiment summary is visualized using **graphs and heatmaps**, which provide an intuitive understanding of public opinion on a video.
- **Structured Workflow**: Use of IAM roles, S3 buckets, and job creation in Comprehend follows AWS best practices and ensures a clean separation of input/output data.

Limitations:

- **No Accuracy Metrics**: Since Amazon Comprehend is a pre-trained service and no labeled dataset is used for comparison, traditional evaluation metrics like accuracy, precision, and recall are not available.
- **Dependent on AWS Services**: The system relies on internet access and AWS availability. Without AWS credentials or access, the pipeline cannot run independently.
- **No Custom Training**: There is no fine-tuning or retraining capability; the analysis is limited to the general-purpose model provided by Amazon Comprehend.

Output Evaluation:

- The final output includes a **.tar.gz file** containing sentiment results.
- The results are processed and visualized, showing that the system accurately reflects the sentiment trends in the dataset.
- For example, the graphs indicate a higher proportion of **positive comments**, followed by **neutral**, **negative**, and **mixed**, aligning well with typical comment trends on popular videos.

CHAPTER 13
COMPARISION WITH EXISTING SYSTEM

In this project, sentiment analysis on multilingual YouTube comments was performed using **Amazon Comprehend**, a managed natural language processing (NLP) service provided by AWS. Below is a theoretical comparison between our proposed system and existing traditional sentiment analysis approaches.

1. Proposed System (AWS Comprehend-based)

Our project utilizes AWS Comprehend, which uses deep learning techniques to extract sentiment from text. It supports multiple languages, automatically detects sentiments as **Positive**, **Negative**, **Neutral**, or **Mixed**, and provides **sentiment confidence scores**.

Key Features:

- **Multilingual Support**: Works seamlessly with non-English languages like Japanese, Hindi, etc.
- **No Model Training Required**: Uses pre-trained models, saving time and effort.
- **Scalability**: Easily handles large datasets via AWS infrastructure.
- **Cloud-Based**: All processing is done on the cloud, reducing local resource dependency.
- **Visualization Ready**: Sentiment scores are exported and visualized using custom Python scripts and graph libraries.

2. Existing Traditional Systems

Traditional systems often rely on machine learning models like **Naive Bayes**, **SVM**, **LSTM**, or **BERT** (for deep learning). These models require a training phase, preprocessing pipelines, and manual configuration.

Key Characteristics:

- **Language Limitation**: Usually focused on English unless specifically trained for multilingual support (e.g., using multilingual BERT).
- **Model Management**: Requires hyperparameter tuning, model evaluation, and testing.
- **Hardware Dependency**: Performance is dependent on local or server-based GPU/CPU resources.
- **Setup Overhead**: Significant effort is needed for training and deployment.

CHAPTER 14
CONCLUSION

The **Multilingual Sentiment Analysis** project provides an effective and scalable solution for understanding public sentiment from diverse linguistic sources. By leveraging cloud-based services and natural language processing tools, the system is capable of handling user-generated content in multiple languages, making it highly relevant in today's globally connected digital environment.

This project demonstrates how automation and cloud infrastructure can simplify sentiment analysis, allowing for quicker and more accurate evaluation of large datasets. The results, presented through sentiment scores and visual graphs, offer valuable insights into audience emotions, helping organizations and researchers make informed decisions. The ability to process comments in different languages without the need for manual translation significantly enhances the project's applicability in real-world scenarios.

One of the key strengths of the system lies in its use of cloud services like AWS, which ensures high reliability, data security, and the flexibility to scale based on demand. It reduces the need for complex local setups and offers a cost-effective way to perform sentiment analysis across languages. Additionally, the automated workflow minimizes human intervention, increasing efficiency and consistency.

In conclusion, this project successfully bridges the gap between multilingual data collection and meaningful sentiment interpretation. It provides a powerful tool for sentiment monitoring that can be extended to various domains such as social media analysis, customer feedback evaluation, and public opinion tracking. The integration of advanced technologies ensures that the system remains both practical and forward-looking in the evolving field of sentiment analysis.

CHAPTER 15
FURTHER ENHANCEMENTS

This project has successfully demonstrated sentiment analysis on multilingual YouTube comments using AWS Comprehend. However, there are several potential enhancements that can extend its functionality, improve performance, and increase its real-world applicability.

1. Real-Time Sentiment Analysis

Enhance the system to analyze sentiments in real time by streaming live comments using the YouTube Data API instead of relying on static CSV uploads. This would allow instant monitoring during events, launches, or debates.

Goal: Stream and analyze live YouTube comments.

AWS Services:

- **AWS Lambda** – to trigger real-time data processing pipelines.
- **Amazon Kinesis Data Streams** – to collect and process streaming comment data.
- **Amazon Comprehend** – for on-the-fly sentiment analysis.
- **Amazon CloudWatch** – for monitoring and alerting based on sentiment spikes.

2. Interactive Visualization Dashboard

Upgrade from basic graphs to an interactive web dashboard using tools like **Plotly Dash**, **Tableau**, or **Power BI**. This would enable dynamic filtering, sentiment comparison across languages, and time-based sentiment trends.

Goal: Provide dynamic filtering and visual exploration of sentiment trends.

AWS Services/Tools:

- **Amazon QuickSight** – to build and publish interactive dashboards directly from S3 data.
- **AWS Glue** – for data cataloging and transformation before visualization.
- **Athena** – to query large sentiment datasets stored in S3.

3. Emotion & Toxicity Classification

Go beyond basic sentiment (positive/negative/neutral) by integrating emotion detection (e.g., joy, anger, sadness) and toxicity/hate speech analysis using advanced NLP models. This adds depth and helps with community moderation and brand safety.

Goal: Detect emotions and toxic language beyond basic sentiment.

AWS Services/Tools:

- **Amazon Comprehend Custom Classification** – to train custom models for emotion tagging.
- **Amazon SageMaker** – to fine-tune transformer models like BERT for detecting emotions or hate speech.
- **Amazon Rekognition (optional)** – if analyzing video/audio content for facial expressions or tone (multi-modal emotion).

CHAPTER 16
REFERENCES

[1] Sarkar, Anindya, Sujeeth Reddy, and Raghu Sesha Iyengar. “Zero-shot multilingual sentiment analysis using hierarchical attentive network and BERT.” Proceedings of the 2019 3rd International Conference on Natural Language Processing and Information Retrieval. 2019.

[2] Antony, Allen, et al. "Leveraging multilingual resources for language invariant sentiment analysis." Proceedings of the 22nd Annual Conference of the European Association for Machine Translation. 2020.

[3] Dhariwal, Naman, Sri Chander Akunuri, and K. Sharmila Banu. "Audio and Text Sentiment Analysis of Radio Broadcasts." IEEE Access 11 (2023): 126900-126916.

[4] Stappen, Lukas, et al. "The multimodal sentiment analysis in car reviews (muse-car) dataset: Collection, insights and improvements." IEEE Transactions on Affective Computing 14.2 (2021): 1334-1350.

[5] Tan, Kian Long, Chin Poo Lee, and Kian Ming Lim. "A survey of sentiment analysis: Approaches, datasets, and future research." Applied Sciences 13.7 (2023): 4550.

[6] Mamani-Coaquira, Yonatan, and Edwin Villanueva. "A Review on Text Sentiment Analysis With Machine Learning and Deep Learning Techniques." IEEE Access (2024).

[7] Kokab, Sayyida Tabinda, Sohail Asghar, and Shehneela Naz. "Transformer-based deep learning models for the sentiment analysis of social media data." Array 14 (2022): 100157.

[8] Mao, Yanying, Qun Liu, and Yu Zhang. "Sentiment analysis methods, applications, and challenges: A systematic literature review." Journal of King Saud University-Computer and Information Sciences (2024): 102048.

[9] Gul, Shahla, et al. "Advancing Aspect-Based Sentiment Analysis in Course Evaluation: A Multi-Task Learning Framework with Selective Paraphrasing." IEEE Access (2025).

[10] Miah, Md Saef Ullah, et al. "A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and LLM." Scientific Reports 14.1 (2024): 9603.

[11] Thakkar, Gaurish, Sherzod Hakimov, and Marko Tadić. "M2sa: multimodal and multilingual model for sentiment analysis of tweets." arXiv preprint arXiv:2404.01753 (2024).

[12] Ivan, Sergiu C., Robert Ş. Győrödi, and Cornelia A. Győrödi. "Sentiment Analysis Using Amazon Web Services and Microsoft Azure." Big Data and Cognitive Computing 8.12 (2024): 166.

[13] Wissam Antoun, Fady Baly, and Hazem Hajj. Arabert: Transformer-based model for arabic language understanding. In LREC 2020 Workshop Language Resources and Evaluation Conference 11–16 May 2020, page 9.

[14] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116, 2019.

[15] Jingfeng Cui, Zhaoxia Wang, Seng-Beng Ho, and Erik Cambria. Survey on sentiment analysis: evolution of research methods and topics. Artificial Intelligence Review, pages 1–42, 2023.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirec tional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[17] Geetika Gautam and Divakar Yadav. Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In 2014 Seventh International Conference on Contemporary Computing (IC3), pages 437–442, 2014.

[18] Akshi Kumar and Victor Hugo C. Albuquerque. Sentiment analysis using xlm-r transformer and zero-shot transfer learning on resource-poor indian language. ACM Trans. Asian Low-Resour. Lang. Inf. Process., 20(5), jun 2021.

[19] Wenxiong Liao, Bi Zeng, Xiuwen Yin, and Pengfei Wei. An improved aspect-category sentiment analysis model for text sentiment analysis based on roberta. Applied Intelligence, 51:3522–3533, 2021.

[20] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

[21] George Manias, Argyro Mavrogiorgou, Athanasios Kiourtis, Chrysostomos Symvoulidis, and Dimosthenis Kyriazis. Multilingual text categorization and sentiment analysis: a comparative analysis of the utilization of multilingual approaches for classifying twitter data. Neural Computing and Applications, pages 1–17, 2023.

[22] Mahmoud Nabil, Mohamed Aly, and Amir Atiya. Astd: Arabic sentiment tweets dataset. In Proceedings of the 2015 conference on empirical methods in natural language processing, pages 2515–2519, 2015.

[23] Sara Rosenthal, Noura Farra, and Preslav Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. arXiv preprint arXiv:1912.00741, 2019.

[24] Anshul Wadhawan. Arabert and farasa segmentation based approach for sarcasm and sentiment detection in arabic tweets. arXiv preprint arXiv:2103.01679, 2021