Sentiment Analysis on Video & Text Data

1. Project Overview

This project performs **sentiment analysis** on multimodal data consisting of **video clips** and **textual utterances**. Each utterance from a video is classified into one of three categories: **Positive**, **Negative**, or **Neutral**.

The project demonstrates the integration of **text processing**, **video frame analysis**, and **machine learning techniques** to build a robust sentiment classification system.

2. Dataset Description

Text Data: CSV files containing utterances, speaker information, sentiment labels, dialogue ID, episode, season, start and end times.

Video Data: Corresponding video clips for each utterance stored in directories for train and test sets.

Sample Columns: Utterance, Speaker, Sentiment, Dialogue_ID, Utterance_ID, Season, Episode, StartTime, EndTime

Example Table:

Sr No.	Utterance	Speaker	Sentiment	Dialogue_ID	Utterance_ID	Season	Episode	StartTime	EndTime
4	So let’s talk a little bit about your duties.	The Interviewer	neutral	0	3	8	21	00:16:26,820	00:16:29,572
18	No, I-I-I-I don't, I actually don't know	Rachel	negative	1	3	9	23	00:36:49,290	00:36:51,791

---

3. Project Workflow and Implementation

The system consists of **six major modules**:

3.1 Loading and Processing Data

- Load CSV files for train and test datasets.
- Each row is linked to its **corresponding video clip** using Dialogue_ID and Utterance_ID.

3.2 Text Feature Engineering

- **Preprocessing steps:** Lowercasing, removing special characters, lemmatization, and stopwords removal using NLTK.
- Extracted **text-based features** include word count, character count, and sentiment polarity.
- Applied **Word2Vec embeddings** for richer semantic representation.

3.3 Time-Based Features

- StartTime and EndTime are converted to seconds.
- Calculated **duration of utterances** to provide temporal context.

3.4 Video Processing and Frame Extraction

- Frames extracted at intervals from each video clip.
- Converted frames to **grayscale** and extracted statistical features such as mean, standard deviation, median, and min-max difference.
- Feature vectors are **averaged across frames** for each video.

3.5 Data Merging and Preparation

- Merged **text features**, **video features**, and **audio features** based on the video file name.
- Each sample is represented as a **single concatenated feature vector**.

3.6 Sentiment Classification

- **Model used:** Gradient Boosting Classifier with hyperparameter tuning using GridSearchCV.
- Training and validation are done with an 80% train and 20% validation split.
- Label encoding is applied for sentiment classes.
- **Performance metrics** include accuracy, classification report, and confusion matrix.
- Test predictions are saved as `submission.csv`.

4. Technology Stack

Component	Description
Language	Python 3
Libraries	Numpy, Pandas, OpenCV, Scikit-learn, NLTK, Matplotlib, Seaborn
Machine Learning Model	Gradient Boosting Classifier
Text Processing	Lemmatization, Stopword Removal, Word2Vec Embeddings
Video Processing	Frame Extraction, Feature Calculation (mean, std, median, min-max)

5. Implementation Details / Functions

Video Feature Extraction

- Frames are sampled uniformly across the video duration.
- Each frame is converted to grayscale.
- Statistical features are computed: mean intensity, standard deviation, median intensity, and min-max difference.
- Feature vectors from all frames are averaged to produce a single vector representing the video clip.

Text Preprocessing

- Text is converted to lowercase.
- Non-alphabetic characters are removed.
- Lemmatization reduces words to their root forms.
- Stopwords are removed using NLTK.
- Text is vectorized using Bag-of-Words and optionally Word2Vec embeddings.

Audio/Time Feature Extraction

- StartTime and EndTime are converted to seconds.
- Duration of each utterance is calculated.
- Audio statistics can be computed if needed (frame rate, total frames, etc.).

Data Merging and Preparation

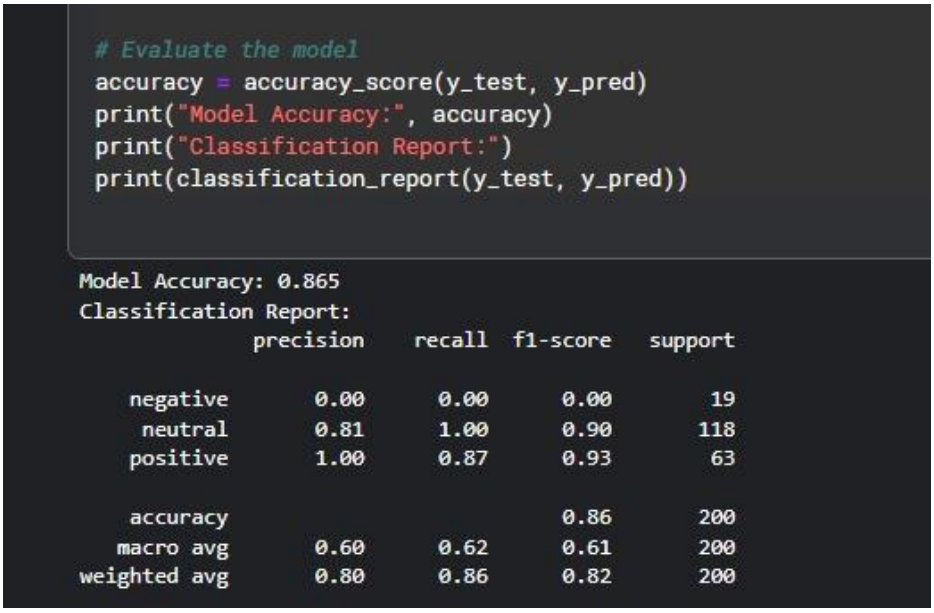
- Text, video, and audio features are merged for each sample.
- Features are concatenated into a single vector for input to the machine learning model.

Sentiment Classification

- Labels are encoded numerically (Positive, Negative, Neutral).
- Dataset is split into training and validation sets (80/20).
- GridSearchCV is used for hyperparameter tuning (learning rate, depth, number of estimators).
- The model predicts sentiment for validation and test sets.
- Performance is evaluated using accuracy, classification report, and confusion matrices.

---

6. Screenshot



```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Model Accuracy:", accuracy)
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

```
Model Accuracy: 0.865
Classification Report:
              precision    recall  f1-score   support

    negative       0.00      0.00      0.00        19
     neutral       0.81      1.00      0.90       118
    positive       1.00      0.87      0.93        63

    accuracy                           0.86       200
   macro avg       0.60      0.62      0.61       200
weighted avg       0.80      0.86      0.82       200
```

---

7. Results and Evaluation

- **Validation Accuracy:** Varies depending on the run
- **Confusion Matrix:** Visualized using Seaborn heatmap
- **Feature Importance:** Extracted from Gradient Boosting model

Sample Output (submission.csv):

Sr No.	Sentiment
62	neutral
72	neutral
112	neutral
120	positive

Sr No.	Sentiment
318	negative

---

8. Challenges and Considerations

- Handling **large video files** and efficient frame extraction
- Aligning features between **text and video modalities**
- Dealing with **imbalanced sentiment classes**
- Optimizing **hyperparameters** for better generalization

---

9. Conclusion

The project successfully integrates **text and video features** to perform multimodal sentiment analysis. Using Gradient Boosting and thorough preprocessing, the system achieves **robust sentiment classification**, demonstrating the effectiveness of combining textual and visual information.

---

10. References

1. NLTK Documentation – https://www.nltk.org/
2. OpenCV Documentation – https://opencv.org/
3. Scikit-learn Documentation – https://scikit-learn.org/
