Project Report: Word-Based Sentiment Scoring and Trend Analysis on Historical Texts – Team 11

---

1. Project Overview

This project analyzes sentiment trends in historical literary texts using a Hadoop-based distributed data processing pipeline. The goal is to explore how sentiments expressed in books, plays, and letters from the 18th to 21st centuries evolve over time. The system uses Java MapReduce and Hive, employing NLP techniques such as lemmatization and sentiment scoring, and integrates metadata such as book ID, title, and publication year for contextual analysis.

---

2. Dataset

The dataset comprises multiple text files sourced from Project Gutenberg, each representing a digitized book. Metadata such as book ID, title, and release year were extracted from the text content or inferred from the file names.

---

3. System Architecture

The project is structured into five main tasks, forming a pipeline:

1. Preprocessing and metadata extraction
2. Word frequency analysis with lemmatization
3. Sentiment scoring using a lexicon
4. Trend aggregation over decades
5. Bigram analysis using Hive UDF

---

4. Task-wise Implementation

Task 1: Preprocessing MapReduce Job

Objective: Extract metadata and clean text for further analysis.

Mapper:

- Reads each line of the book.
- Accumulates full content.
- Extracts metadata: book ID (from file name), title, and release year (from header).
- Cleans text: converts to lowercase, removes punctuation, filters stop words.
- Emits a composite key: `bookID_year_title` and cleaned text as value.

Reducer:

- Concatenates all lines (if multiple for a book).
- Outputs key-value with full preprocessed content.

Sample Output:

```
105_1994_persuasion\tproject gutenberg ebook persuasion
use anyone united states most parts world
```

Challenges:

- Variability in text formatting across different books.
- Ensuring accurate extraction of title and year.

Task 2: Word Frequency Analysis with Lemmatization

Objective: Tokenize text and compute lemma frequency per book and year.

Mapper:

- Splits lines using tab into metadata and content.
- Uses Stanford CoreNLP Simple API for lemmatization.
- Emits: `bookID, lemma, year` with value `1`.

Reducer:

- Aggregates frequency of each lemma for a book in a given year.

Sample Output:

```
10148, 3, 2003\t12
10148, 4, 2003\t7
10148, 5, 2003\t4
```

Challenges:

- Integration of CoreNLP with Hadoop MapReduce.
- Tokenizing malformed or numeric-heavy text.

Task 3: Sentiment Scoring

Objective: Compute total sentiment score for each book using AFINN lexicon.

Mapper:

- Loads AFINN-111 sentiment lexicon during setup.
- Tokenizes each word and checks for sentiment score.
- Emits: `(bookID, year)` and sentiment score.

Reducer:

- Aggregates sentiment score for each book.

Sample Output:

```
10148, 2003\t4702
105, 1994\t2845
8183, 2005\t-185
```

Challenges:

- Managing memory during lexicon loading.
- Avoiding false positives from malformed inputs.

Task 4: Trend Analysis & Aggregation

Objective: Aggregate scores by decade to observe long-term trends.

Mapper:

- Extracts decade from year.
- Emits `(bookID, decade)` and score.

Reducer:

- Sums scores per decade.

Sample Output:

```
10148, 2000s\t4702
8183, 2000s\t-185
```

Challenges:

- Handling missing or invalid year data.

Task 5: Bigram Analysis with Hive UDF

Objective: Extract bigrams from lemmatized texts using Hive and Java UDF.

Implementation:

- Java UDF accepts cleaned, lemmatized text and returns a list of bigrams.
- Hive loads input as external table.
- Query invokes UDF to extract and count bigram occurrences.

---

5. Evaluation & Insights

- Positive sentiment scores were higher in 20th–21st century literature.
- Lemmatization helped group variations of words, improving clarity.
- The bigram analysis highlighted recurring phrase patterns.

Challenges Faced:

- Difficulty in parsing inconsistent text formatting.
- Memory limitations while working with large lexicons or book files.
- Handling numeric or unrelated tokens.

---

6. Future Improvements

- Integrate POS tagging for contextual lemmatization.
- Switch to SentiWordNet for richer lexicon support.
- Normalize sentiment scores for text length.
- Add visualizations for decade-wise sentiment progression.
- Deploy the pipeline using Apache Oozie or Spark for better performance.

---

7. Team Contributions

- **Task 1 – Preprocessing:** Harish Varma Gottumukkula
- **Task 2 – Lemmatization & Frequency:** Madhavi Badugu
- **Task 3 – Sentiment Scoring:** Amrutha Reddy
- **Task 4 – Trend Aggregation:** Vamsi Kopparthi
- **Task 5 – Bigram Hive UDF:** Harsha Paladugu

---

8. Conclusion

This project demonstrated how large-scale distributed processing can be leveraged to analyze historical text data. By combining metadata, NLP, and sentiment scoring techniques in a Hadoop ecosystem, we were able to gain insights into the sentiment evolution of literature over time.

Technologies Used:

- Hadoop MapReduce (Java)
- Hive + Java UDF
- Stanford CoreNLP
- AFINN Lexicon

All code, data, and results are available in the group's GitHub repository.