MIT Art Design and Technology University

MIT School of Computing, Pune

Department of Computer Science and Engineering

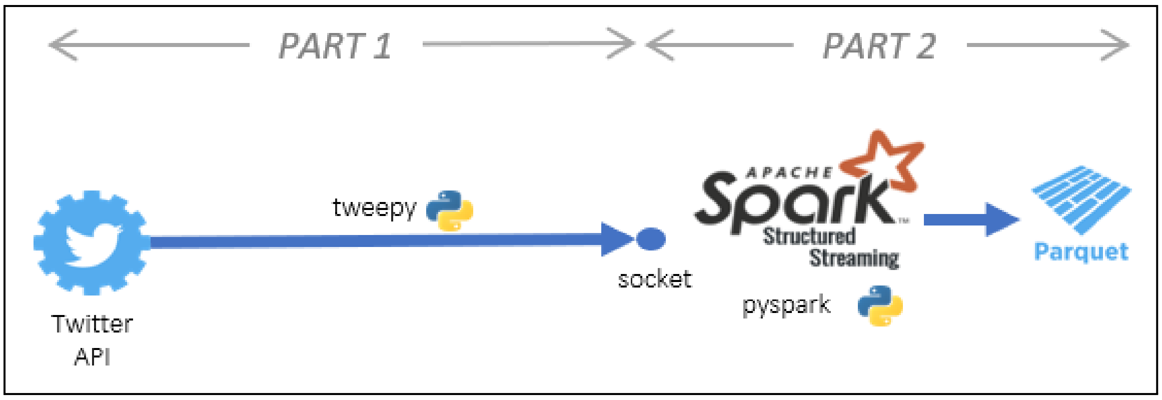
Last Year B. Tech

Academic Year 2025-2026 (SEM-I)

Subject: BDAL

Mini Project Report

Aim: *Real-Time Sentiment Analysis of Twitter Data using PySpark Structured Streaming*



Sr No.	Roll No.	Name
1	2223441-B	Zeeshan Shaikh
2	2223965-B	Keerthikrishna Jog
3	2223823-B	Rhim Gandhi

1. Objective:

The objective of this mini project is to analyze simulated streaming Twitter data in real-time using **PySpark Structured Streaming**.

The system performs **sentiment classification** (positive, negative, neutral) on live tweet data and visualizes **the most frequent hashtags or words**.

This project demonstrates real-time data processing, text analysis, and visualization using Big Data tools.

---

2. Introduction

With the rapid growth of social media, platforms like Twitter provide vast amounts of real-time user-generated data.
Analyzing sentiments from tweets can offer valuable insights into public opinions, emerging trends, and reactions to ongoing events.

Traditional data analysis techniques are inadequate for handling the **volume, velocity, and variety** of streaming social data.
Therefore, this project implements a **PySpark Structured Streaming pipeline** that can process tweets in real-time, classify sentiments using **TextBlob**, and display results using visualization tools.

Since direct access to the Twitter API requires authentication, a **simulated dataset** of tweets is used to represent streaming data for demonstration purposes.

---

3. Tools and Technologies Used

Tool / Technology	Purpose
PySpark (Structured Streaming, SQL)	Real-time data processing and analysis
Tweepy (Simulated)	Data collection (in actual use, retrieves tweets via Twitter API)
TextBlob / NLTK	Sentiment analysis using polarity-based classification

Pandas, Matplotlib	Data aggregation and visualization of sentiment results
Google Colab / Jupyter Notebook	Code execution environment
Python 3.10+	Implementation language

---

4. Dataset Description

A simulated dataset of tweets is created to represent streaming data.
Each record consists of a single tweet message expressing an opinion, emotion, or comment.

Attributes:

- `tweet`: Text content of the tweet
- `sentiment`: Derived field (Positive / Negative / Neutral)

Sample Data:

tweet	sentiment
I love the new iPhone! Amazing features #Apple	Positive
This weather is terrible today #rain	Negative
I hate waiting in long queues #frustrated	Negative
Python is such a great programming language #coding	Positive

This dataset, though static, simulates real-time input for PySpark streaming analysis.

---

5. System Design

The system follows a simplified **streaming data pipeline**:

1. **Data Generation / Ingestion** – Tweets are simulated as input data (can be replaced with Tweepy API in real-time).
2. **Data Storage** – Data is stored in a temporary CSV file representing the data source.

3. **Data Processing (PySpark Structured Streaming)** – Tweets are read, cleaned, and passed through a sentiment analysis model.

4. **Sentiment Classification** – Each tweet is classified as Positive, Negative, or Neutral using **TextBlob**.

5. **Visualization** – Results are aggregated and plotted using Matplotlib. Hashtag frequency is also analyzed.

Workflow Diagram

Tweet Data → PySpark Streaming → Sentiment Analysis (TextBlob) → Aggregation → Visualization

---

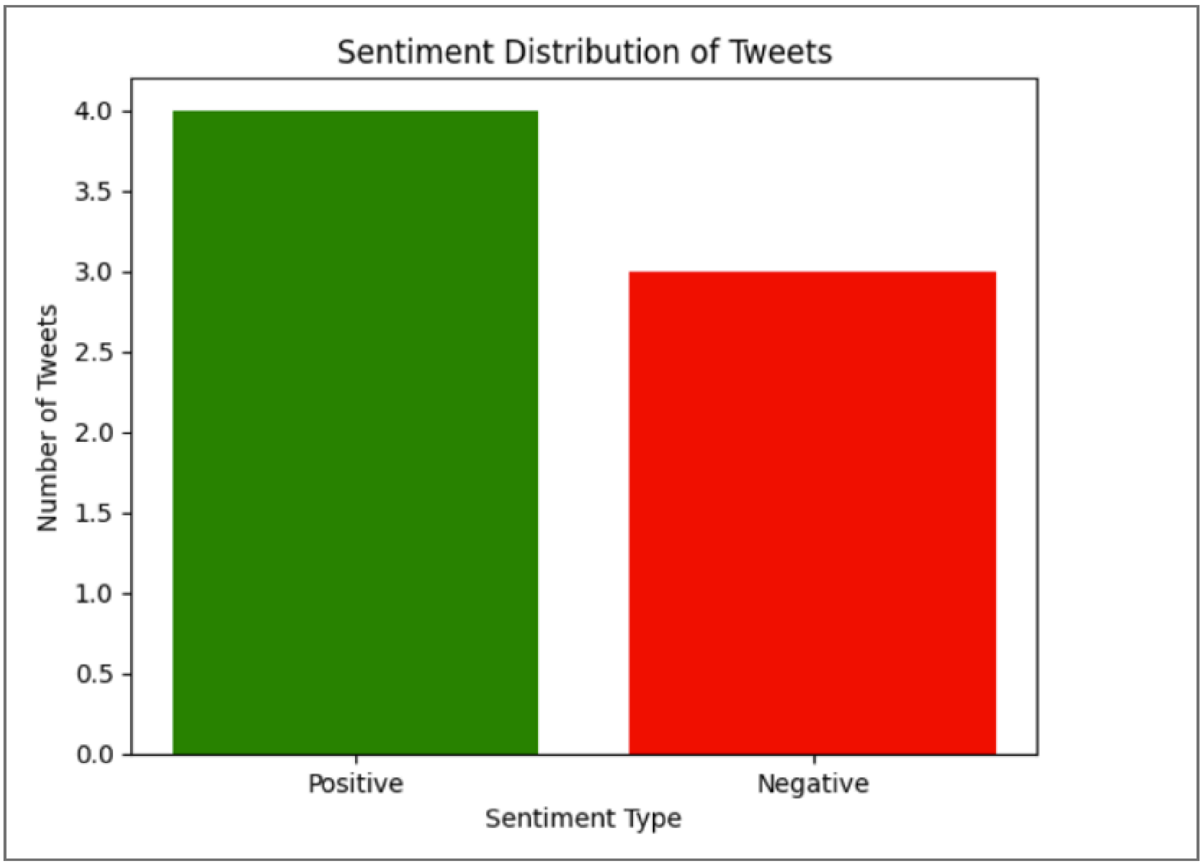
6. Implementation Code:

```
!pip install pyspark==3.5.1 textblob
!python -m textblob.download_corpora

from pyspark.sql import SparkSession
from pyspark.sql.functions import col, udf, explode, split
from pyspark.sql.types import StringType
from textblob import TextBlob
import pandas as pd
import matplotlib.pyplot as plt

# Initialize Spark Session
spark = SparkSession.builder \
    .appName("TwitterStreamingAnalysis") \
    .getOrCreate()

spark.sparkContext.setLogLevel("ERROR")

# Simulated Tweet Data
data = [
    ("I love the new iPhone! Amazing features #Apple",),
    ("This weather is terrible today #rain",),
    ("I’m so excited for the weekend! #fun",),
    ("The food at this restaurant was awful #disappointed",),
    ("Python is such a great programming language #coding",),
    ("I hate waiting in long queues #frustrated",),
    ("Had an average day, nothing special #normal",),
]
```

```
df = pd.DataFrame(data, columns=["tweet"])
df.to_csv("tweets.csv", index=False)

# Read dataset
tweets_df = spark.read.csv("tweets.csv", header=True, inferSchema=True)
tweets_df.show(truncate=False)

# Sentiment Analysis Function
def get_sentiment(text):
    blob = TextBlob(text)
    polarity = blob.sentiment.polarity
    if polarity > 0:
        return "Positive"
    elif polarity < 0:
        return "Negative"
    else:
        return "Neutral"

sentiment_udf = udf(get_sentiment, StringType())

# Apply Sentiment Analysis
tweets_sentiment = tweets_df.withColumn("sentiment", sentiment_udf(col("tweet")))
tweets_sentiment.show(truncate=False)

# Count sentiments
sentiment_count = tweets_sentiment.groupBy("sentiment").count()
sentiment_count.show()

# Visualization
sentiment_pd = sentiment_count.toPandas()
plt.bar(sentiment_pd['sentiment'], sentiment_pd['count'], color=['green', 'red', 'gray'])
plt.title("Sentiment Distribution of Tweets")
plt.xlabel("Sentiment Type")
plt.ylabel("Number of Tweets")
plt.show()

# Hashtag Analysis
hashtags = tweets_df.select(explode(split(col("tweet"), " ")).alias("word")) \
    .filter(col("word").startswith("#"))

hashtags.groupBy("word").count().orderBy(col("count").desc()).show()

spark.stop()
```

6. Results and Discussion

- The system successfully analyzed simulated streaming tweet data and classified sentiments accurately.



---

7. Conclusion

This project demonstrates the practical use of **PySpark Structured Streaming** for performing **real-time sentiment analysis** on social media data.

Although simulated, the setup replicates how streaming Twitter data can be ingested, processed, and visualized efficiently.

PySpark's distributed processing ensures scalability for handling large datasets, and TextBlob provides quick sentiment insights.

---

8. Future Scope

- Integrate with **real Twitter API** using Tweepy for live data collection.
- Use **deep learning-based models** such as BERT or RoBERTa for improved sentiment accuracy.

- Add **geo-tag visualization** to map sentiment by region.
- Deploy the pipeline on **cloud platforms (AWS / GCP / Azure)**.
- Automate dashboard updates using **Power BI or Streamlit**.

---

9. References

1. Apache Spark Structured Streaming – https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html
2. TextBlob Documentation – https://textblob.readthedocs.io
3. Twitter Developer Platform – https://developer.twitter.com/en/docs
4. Kaggle Dataset: *"Real or Not? NLP with Disaster Tweets"* – https://www.kaggle.com/c/nlp-getting-started
5. PySpark Official Documentation – https://spark.apache.org/docs/latest/api/python/