Real-Time Sentiment Analysis

Noor Ahed Joud Dababneh Own Al-Raggad

Supervisor: Dr. Mu'nis Qasaymeh

Cloud Computing

Department of Artificial Intelligence
King Abdullah II School of Information Technology

University of Jordan

2024/2025

Contents

Abstract

This project presents a real-time sentiment analysis pipeline designed to process large-scale social media streams. By leveraging PySpark for data ingestion, TensorFlow's RoBERTa model for sentiment classification, and Firebase for real-time data storage and visualization, the system provides valuable insights into public sentiment. The cloud-native architecture ensures scalability and reliability, making it suitable for dynamic environments. The project demonstrates the potential of modern NLP techniques in real-time applications, with future work focusing on improving sentiment accuracy and expanding data sources.

1 Introduction

The rapid growth of social media platforms has transformed the landscape of communication, providing a wealth of user-generated content that encapsulates sentiments, opinions, and reactions to global events [1] [2]. Platforms such as Twitter, Facebook, and Instagram are not only outlets for personal expression but also rich data sources for understanding public opinion and behavior. The ability to analyze these vast, unstructured data streams in real time is critical for applications in marketing, public opinion tracking, crisis management, and event monitoring.

Sentiment analysis, a branch of natural language processing (NLP), focuses on extracting and interpreting emotions, opinions, and intents embedded in text. Traditionally, sentiment analysis has centered on categorizing data into positive, negative, or neutral sentiments [3] [4]. However, advances in machine learning have expanded its scope to include contextual sentiment interpretation, enabling more nuanced insights into user-generated content [5]. In this context, real-time sentiment analysis aims to classify sentiments from continuously streaming data, providing actionable intelligence for decision-making in dynamic environments.

Accurate and scalable sentiment analysis is essential for processing large volumes of data and extracting meaningful trends. By leveraging machine learning and cloud-native architectures, real-time sentiment analysis pipelines can provide immediate insights into customer opinions, social trends, and public reactions[2]. The integration of real-time processing with advanced visualization tools facilitates informed decision-making across domains such as marketing strategy, policy formulation, and brand reputation management.

The proposed project introduces a real-time sentiment analysis pipeline designed to process large-scale social media streams efficiently. This system employs PySpark for data ingestion, machine learning models using TensorFlow LLM APIs for sentiment classification, and Firebase for real-time data storage and visualization. To ensure scalability and reliability, the pipeline is built on cloud-native platforms, leveraging Kubernetes. These components work cohesively to deliver a robust framework capable of handling the demands of dynamic, data-intensive environments.

By addressing challenges such as unstructured data processing, contextual sentiment classification, and scalable deployment, this pipeline demonstrates the transformative potential of modern NLP techniques. It provides a foundation for understanding and acting on real-time insights, empowering organizations to navigate an increasingly data-driven world.

2 Background

The implementation of a real-time sentiment analysis pipeline requires a combination of cutting-edge tools and technologies to ensure efficiency, scalability, and adaptability. This section explores the key concepts and tools integral to the project, along with their relevance to a cloud-native approach.

2.1 PySpark

Apache Spark and its Python API, PySpark, provide robust solutions to the challenges of large-scale data processing. PySpark [6] is a powerful big data framework that supports distributed computing, making it highly efficient for handling extensive datasets. Its capabilities include in-memory processing and seamless integration with various machine learning libraries, significantly accelerating the analysis and processing of large volumes of textual data [7]. PySpark's distributed computing model ensures rapid processing of massive datasets, which is essential given the high throughput of data generated in real-time applications. Additionally, Spark MLlib, Spark's machine learning library, offers comprehensive tools for feature extraction, transformation, and classification, enabling the development of scalable sentiment analysis models [7].

Despite its potential, the application of PySpark in real-time sentiment analysis has not been extensively explored. This gap presents an opportunity to evaluate PySpark's effectiveness in managing and analyzing large-scale textual data streams. Prior studies have often focused on smaller datasets or employed traditional machine learning techniques, overlooking the scalability and efficiency provided by big data frameworks like PySpark [21]. By leveraging PySpark to process and analyze vast amounts of unstructured data, it is possible to enhance the understanding of sentiment trends and improve predictive analytics. Furthermore, integrating traditional machine learning models with PySpark could provide a robust approach, combining the strengths of both technologies [22].

2.2 Cognitive Computing

Cognitive computing, inspired by human thought processes, combines advanced algorithms and AI to tackle challenges in decision-making and natural language processing [8]. The rise of big data has further amplified the significance of cognitive computing. With an estimated "2.5 quintillion bytes of data" generated daily [9], the need for systems capable of processing, analyzing, and deriving actionable insights from diverse and massive datasets has become paramount. Big data's characteristics—volume, velocity, variety, veracity, and value—pose unique challenges to traditional data analysis methods. Cognitive computing addresses these challenges by combining multidisciplinary techniques, including big data frameworks like Hadoop and advanced machine learning (ML) approaches.

In sentiment analysis, cognitive computing enables nuanced insights from unstructured and structured data. Techniques like TF-IDF [10] and machine learning models such as SVM and Decision Trees classify sentiments with high accuracy, while methods like Binary Brain Storm Optimization (BBSO) and Fuzzy Cognitive Maps (FCMs) enhance feature selection and classification.

By leveraging big data tools like Hadoop MapReduce and PySpark, cognitive computing supports real-time sentiment analysis of high-velocity streams, offering actionable insights into public opinion and emerging trends. This integration drives informed decision-making and deeper understanding of user behaviors.

2.3 TensorFlow LLM

TensorFlow, an open-source machine learning framework developed by Google, has revolutionized the development and deployment of Large Language Models (LLMs). With its scalability, flexibility, and extensive ecosystem of tools, TensorFlow facilitates the training and fine-tuning of LLMs on vast datasets, enabling tasks such as sentiment analysis, text generation, and contextual understanding.

LLMs leverage Transformer architectures, characterized by self-attention mechanisms, to process and understand complex language structures. TensorFlow's efficient handling of these architectures supports both pre-training and fine-tuning phases, optimizing performance on domain-specific tasks.

Frameworks like TensorFlow Hub and TensorFlow Lite further extend LLM deployment capabilities, making it feasible to integrate these models into applications requiring real-time inference.

For sentiment analysis pipelines, TensorFlow offers seamless integration with other tools, enabling real-time data processing, model serving, and scalability. Pre-trained LLMs accessible via TensorFlow APIs, such as BERT and GPT-like models, provide robust baselines that can be fine-tuned to achieve state-of-the-art results in natural language understanding tasks.

By combining TensorFlow's versatility with the power of LLMs, this project enables efficient, scalable, and accurate sentiment analysis, meeting the demands of high-velocity data streams.

2.4 Firebase

Firebase, a Google-backed development platform, simplifies the creation and management of web and mobile applications with its comprehensive suite of tools [11]. Key features include authentication, real-time database, cloud messaging, and crash reporting. Firebase Authentication supports login methods like email-password, phone authentication, and third-party providers, ensuring seamless user onboarding. The real-time database enables synchronized data updates across devices, even offline, while Firebase Cloud Messaging facilitates reliable cross-platform notifications. Crashlytics provides real-time crash reporting, helping developers identify and resolve issues efficiently.

Firebase's applications are vast, ranging from real-time collaboration tools and live chats to progressive feature rollouts. Its integration with Google Cloud ensures scalability, reliability, and security, while reducing development overhead. However, limitations include restricted query capabilities, challenges in scaling for larger systems, and dependency on Google, which can reduce backend flexibility. Despite these constraints, Firebase remains a powerful solution for developers seeking efficient, cross-platform app development.

2.5 Cloud Native Relevance

The adoption of a cloud-native architecture is fundamental to the scalability and resilience of the sentiment analysis pipeline. Cloud-native systems are designed to leverage distributed computing resources, containerization, and microservices architecture to maximize performance and adaptability. Technologies like Kubernetes or serverless platforms on GCP, AWS, or Azure enable the deployment of this pipeline with high availability and fault tolerance. By building the system with a cloud-native approach, the project ensures that it can scale to handle fluctuating data loads, integrate seamlessly with external services, and maintain consistent performance across diverse deployment environments.

3 Methodology

This section outlines the systematic approach taken to implement the project, including a detailed description of the tools, libraries, and cloud services utilized.

1. **Initialization of Firebase Environment**

   The project began with setting up Firebase as the backend to handle database operations securely and efficiently. The Firebase Admin SDK was integrated into the Python environment using Google Colab. Credentials for Firebase were obtained and authenticated to enable seamless communication with the Firestore database.

2. **Database Configuration**

   The Firestore database was initialized using the `firebase_admin` library. This setup enabled efficient storage and retrieval of structured data. The `firestore.client()` method was employed to establish a connection with the database.

3. **Data Collection**

   The project involved retrieving real-time data, such as tweets, using an API. Authentication tokens and necessary keys were stored securely using the `userdata` module in Google Colab, ensuring the protection of sensitive information.

4. **Data Processing**

   Python's `datetime` module was employed for processing timestamps associated with the retrieved data. The data was organized into structured dictionaries to facilitate easier manipulation and analysis.

5. **Model Integration**

   The RoBERTa (Robustly Optimized BERT Pretraining Approach) model was employed to analyze and process textual data. This transformer-based model, known for its efficiency and accuracy in natural language understanding tasks, was fine-tuned to suit the specific requirements of the project [12]. The model was integrated into the pipeline using the Hugging Face Transformers library, enabling advanced text classification and sentiment analysis capabilities.

6. **Deployment**

   The project was deployed using Docker to ensure consistent and reproducible environments across different systems. A `Dockerfile` was created to define the container's behavior, including setting up the working directory, copying necessary files (e.g., `service_account.json`), installing dependencies from a `requirements.txt` file, and exposing the required port. The `requirements.txt` file contained the following dependencies:

   - `streamlit`
   - `torch`
   - `transformers`
   - `firebase-admin`
   - `scipy`

   The Streamlit app (`app.py`) was set to launch automatically when the container started, providing an interactive interface for users.

7. **Real-Time Analysis and Interaction**

   The system was designed to process incoming data dynamically. This involved filtering and organizing information based on predefined criteria to provide meaningful insights or trigger specific actions.

3.1 Tools and Libraries

1. **Programming Language**

   Python was the primary language used for the project due to its versatility and rich ecosystem of libraries.

2. **Firebase Admin SDK**

   This library was employed to connect and interact with the Firestore database.

3. **Google Colab**

   A cloud-based platform was chosen for coding and executing the project, leveraging its built-in support for Python and integration with Google Cloud services.

4. **Firestore**

   A scalable NoSQL database was used for storing and retrieving data efficiently.

5. **Hugging Face Transformers**

   This library was used to implement the RoBERTa model for natural language processing tasks.

6. **Docker**

   Used for deploying the project in a containerized environment, ensuring portability and consistency.

3.2 Cloud Services

1. **Google Cloud Platform (GCP)**

   Firebase, which operates under GCP, was the primary cloud service utilized for backend operations. The integration ensured scalability and reliability of the database.

2. **Google Colab**

   Serving as the primary development environment, Colab provided the necessary computational resources and tools for implementation.

This systematic approach ensured the project was implemented effectively, leveraging state-of-the-art tools and technologies to meet its objectives.

4 Results and Discussion

The sentiment analysis pipeline was able to process social media data streams and classify sentiments in real time. The system showed the ability to handle large amounts of data and provide insights into public sentiment trends, though performance varied based on data volume. Visualizations of sentiment trends offered an overview of shifts in public opinion, particularly around specific events. While the system performed well, challenges arose related to fluctuations in data load, which were managed by optimizing the processing pipeline. Additionally, some ambiguities in sentiment classification, such as handling sarcastic or unclear language, presented challenges.

5 Conclusion

The real-time sentiment analysis pipeline successfully demonstrated the integration of big data frameworks and machine learning for dynamic data processing. It provided actionable insights into public sentiment trends, enabling timely and informed decision-making. Key achievements include efficient real-time classification with low latency, robust handling of unstructured data, and seamless cloud-native deployment.

While the project showcased the potential of real-time sentiment analysis, challenges such as enhancing model generalizability and further optimizing scalability remain. Future work will focus on refining model accuracy through transfer learning with diverse datasets and extending the system to support multimedia inputs like images and videos. Expanding deployment to multi-cloud environments will also improve fault tolerance and global accessibility.

References

[1] R. Panikar, R. Bhavsar, and B. V. Pawar, "Sentiment analysis: a cognitive perspective," in *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, pp. 1258–1262, 2022.

[2] B. Kumar, Sheetal, V. S. Badiger, and A. D. Jacintha, "Sentiment analysis for products review based on nlp using lexicon-based approach and roberta," in *2024 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, pp. 1–6, 2024.

[3] Y. Msala, M. Hamlich, and A. Mouchtachi, "A new robust heterogeneous multi-robot approach based on cloud for task allocation," in *2019 5th International Conference on Optimization and Applications (ICOA)*, pp. 1–4, 2019.

[4] O. Hamed and M. Hamlich, "Hybrid formation control for multi-robot hunters based on multi-agent deep deterministic policy gradient," *MENDEL*, vol. 27, pp. 23–29, Dec. 2021.

[5] A. Goel, J. Gautam, and S. Kumar, "Real time sentiment analysis of tweets using naive bayes," in *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, pp. 257–261, 2016.

[6] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A Fault-Tolerant abstraction for In-Memory cluster computing," in *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*, (San Jose, CA), pp. 15–28, USENIX Association, Apr. 2012.

[7] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar, "Mllib: Machine learning in apache spark," *Journal of Machine Learning Research*, vol. 17, no. 34, pp. 1–7, 2016.

[8] D. K. Jain, P. Boyapati, J. Venkatesh, and M. Prakash, "An intelligent cognitive-inspired computing with big data analytics framework for sentiment analysis and classification," *Information Processing Management*, vol. 59, no. 1, p. 102758, 2022.

[9] H. Thakkar and D. Patel, "Approaches for sentiment analysis on twitter: A state-of-art study," 2015.

[10] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing Management*, vol. 24, no. 5, pp. 513–523, 1988.

[11] P. Chougale, V. Yadav, A. Gaikwad, and B. Vidyapeeth, "Firebase-overview and usage," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 3, no. 12, pp. 1178–1183, 2021.

[12] F. Barbieri, J. Camacho-Collados, L. Espinosa Anke, and L. Neves, "TweetEval: Unified benchmark and comparative evaluation for tweet classification," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, (Online), pp. 1644–1650, Association for Computational Linguistics, Nov. 2020.