CS5296 - Cloud Computing
Semester B 2023-2024

Group Project Final Report

Submitted on *30-04-2024*

Weibo Hot Topic Timeline Based on AWS Platform

Group 9

58208460	Yixin LI
58276105	Tianyi LIANG
58097428	Peilin LIU
58336862	Guanchen LYU
58277901	Zihan XIAO

Abstract: Our report presents a cloud computing solution to address information overload on Weibo. We use Hadoop or Spark to process data and train machine learning models on a distributed cloud platform, improving accuracy and performance. We also develop a web crawler to extract social media data, perform sentiment analysis, and generate timelines. Our goal is to provide Weibo users with a coherent timeline and tackle the challenge of information overload.

1. Introduction

Social media platforms have become an important source of news and opinions for people due to the rapid development of information technology. However, the abundance of information and fragmentation of content on these platforms has led to information overload, especially during major news events. Weibo, a social media platform, is facing the challenge of managing this fast and complex flow of information. To address this challenge, we are collecting social media data from Weibo and creating event timelines based on that data.

Our first task is to capture daily hot topics on Weibo and use big data processing systems such as Hadoop [1] or Spark [2] for storing and processing the data. By using advanced text analysis techniques, we can efficiently process the information and understand the behavior and preferences of Weibo users.

To improve the accuracy and performance of our model, we will train a machine learning model on a distributed platform [3] using cloud computing technology. This allows us to take advantage of the scalability and processing power of distributed systems, which speeds up the computations.

During the model training process, we will compare the time efficiency of training the model on a cloud server versus a local system. We will use Spark for big data processing, PMLS for managing machine learning tasks, and deep learning frameworks like TensorFlow [4,5] and MXNet [6] for model construction and training within the AWS context.

To enhance the search functionality for users, we will store important project data such as vocabulary, weight lists, weight matrices, and inverted index tables in a cloud repository. This will enable faster and more efficient retrieval of information during search queries, allowing users to find what they need quickly.

By implementing these methodologies and technologies, our aim is to tackle the problem of information overload on Weibo by providing users with a comprehensive and coherent timeline that helps them understand evolving events more deeply.

2. Methodology

The methodology section of this project explains the systematic approach taken to tackle the issue of information overload on the Weibo social media platform. This section details the key steps and techniques used to gather social media data through crawling, conduct text analysis, train a machine learning model, and ensure efficient search functionality for users.

2.1. Project Functions

The project comprises of two main modules: Sentiment Analysis and Timeline Generation. The overall workflow involves data crawling, training models separately for each module, and storing the trained models for future use.

2.1.1. Sentiment Analysis

We use various techniques to analyze user sentiments from social media data collected via Weibo. Our sentiment classification system includes tokenization, jieba text segmentation, and LSTM networks [7], which accurately identify positive or negative sentiments in Chinese text. By processing data efficiently, we gain valuable insights into user behavior and preferences related to specific events or topics.

2.1.2. Timeline Generation

In this step, we analyze the crawled data to identify events related to the specified time range and keywords. The figure below illustrates the entire search process.

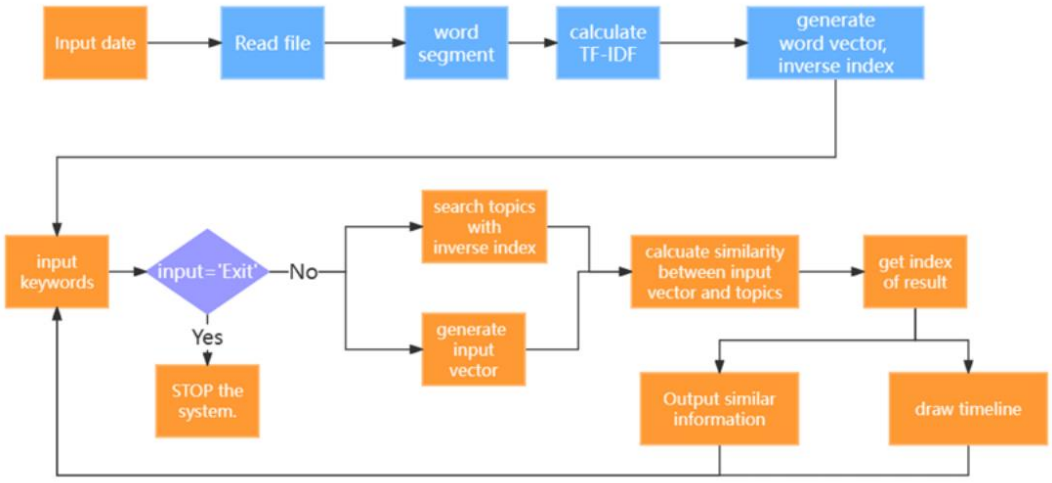

Fig 1. Flow chart of retrieval and timeline generator system

2.2. Data Crawling

Our approach involves the development of a customized web crawler that is specifically designed to extract social media data from Weibo. This crawler will capture daily hot topics, posts, and associated metadata by navigating the platform and programmatically retrieving the desired information.

We will use web scraping techniques to navigate Weibo and identify trending keywords. Our focus will be on collecting posts and metadata related to these hot topics, adapting to any changes in the platform's structure.

To manage the large volume of data, we will utilize robust storage systems like Hadoop or Spark, ensuring scalability and efficiency. The crawled data will be stored in a structured format for further processing and analysis.

By implementing a tailored web crawler and leveraging powerful data storage systems, we can reliably retrieve and store social media data from Weibo. This data will

serve as the foundation for subsequent analysis and timeline construction.

2.3. Sentiment Analysis Model Training

The task at this step is to use Weibo data that has been collected to train sentiment analysis models. To preprocess the text data, perform tasks such as tokenization, removing stop words, and processing emoticons or emoticons. After preprocessing, compare the performance of two different deep learning frameworks – TensorFlow [4,5] and MXNet [6] - for sentiment analysis.

TensorFlow has broad adoption and a mature ecosystem, making it easy to access documentation, tutorials, and sample code to speed up development. MXNet, like TensorFlow, has extensive applications and community support. MXNet is particularly good at distributed computing and can handle large-scale datasets and models efficiently.

2.4. Timeline Generation Model Training

We followed a set of steps to create a timeline of Weibo data. Firstly, a model was trained to process text data using techniques such as named entity recognition, event extraction, and temporal relationship extraction. Next, we compared the performance of two different technologies - regular Python and Spark [2,8] - in training the model.

We began by developing a timeline generation model using regular Python and evaluated its performance. Wherever necessary, we adjusted and optimized the model. Then, we repeated the same process using the distributed computing framework Spark and compared the results with the model based on conventional Python.

Finally, we saved the trained timeline generation model for future use. We chose to use Spark as a distributed computing framework because of its scalability and parallelism. Spark provides advanced data processing and machine learning libraries that can effectively process large-scale data sets and fully utilize the computing power of the cluster to accelerate the model training process.

2.5. Model Storage

To provide users with efficient search functionality and retrieve information quickly, our team has decided to implement a cloud repository. This repository will store important project data, such as vocabulary lists, weight matrices, weight lists, and inverted index tables that are obtained from the trained models. By utilizing cloud storage services, we can ensure that the data is securely stored and easily accessible at any time.

AWS offers several storage services, including Amazon S3, Amazon EFS, and Amazon DynamoDB. Based on the specific needs and usage scenario of this project, the storage service chosen is Amazon S3. The project involves storing and retrieving unstructured data, such as vocabulary, weight lists, weight matrices, and inverted index tables. Amazon S3's scalability, durability, and availability make it an ideal choice for storing large amounts of unstructured data. Furthermore, its simplicity and ease of integration with other AWS services ensure seamless access to the stored data for search functionality.

By selecting Amazon S3 as the cloud repository for this project, we can rely on its robust storage capabilities and benefit from its high availability and accessibility, which enables efficient retrieval of data during search queries.

2.6. Comparative Study

We plan to conduct a comparative study to assess the time efficiency and performance of our project by comparing training the machine learning model on a cloud server with deploying it solely on a local system. We aim to identify the most effective and efficient approach for model training.

The study will focus on the following aspects:

Training Time: We will measure and compare the time taken to train the machine learning model in both environments, including data preprocessing, feature extraction, model training, and evaluation. This analysis will evaluate the efficiency gains achieved through the cloud server's distributed computing capabilities.

Computational Resources: We will assess the utilization of computational resources such as CPU and memory in both environments to gain insights into the scalability and efficiency of the cloud server and the impact on local system resources.

Performance Metrics: The performance of the trained model will be evaluated using metrics such as accuracy, precision, recall, and F1 score. By comparing these metrics between the cloud and local environments, we can assess any differences or advantages gained from utilizing distributed computing resources.

Scalability and Response Time: We will measure the scalability of the model by gradually increasing the dataset size and observing the impact on training time and performance metrics. Additionally, the response time of the model in both environments, including data retrieval and processing speed during search queries, will be assessed.

Cost Analysis: A cost analysis will be conducted to compare the expenses associated with using a cloud server versus deploying the system solely on local infrastructure. Factors such as cloud service fees, hardware costs, and maintenance expenses will be considered to provide insights into the cost-effectiveness of each approach.

By conducting this comparative study, we aim to determine the most efficient and effective method for training our machine learning model. The findings will inform decisions regarding the utilization of cloud infrastructure, the scalability of the system, and the overall performance and cost considerations of the project.

3. Experiments and Analysis

In order to validate the effectiveness and efficiency of our proposed methodology for constructing event timelines from Weibo data, we conducted a series of experiments.

3.1. Data Collection

In crawling, we search the title of a historical topic from GitHub and construct URLs for each topic. Then, we use the third-party library BeautifulSoup to process the

information and get the introduction. Finally, we organized the time, title, and introduction data into a dictionary and used pandas to save the data in a CSV file. We crawled Weibo hot topic entries from November 2022 to October 2023, with a total of over 100000 entrie, with a total size of approximately 13.2M.

3.2. Model Training and Evaluation

3.2.1. Sentiment Analysis

When training the sentiment analysis model, we used two different deep learning frameworks, TensorFlow and MXNet, and conducted comparative experiments. The following is an experimental comparison:

Table 1. TensorFlow vs. MXNet

Evaluation Metric	TensorFlow	MXNet
Training Time	Relatively Short	Long
Accuracy	High	Low
Performance Evaluation	Good	Poor

Based on the experimental results, we chose to use TensorFlow as the training framework for the sentiment analysis model. TensorFlow is relatively short in training time and performs well in terms of accuracy and performance evaluation. In comparison, MXNet takes longer to train, has lower accuracy, and has relatively weak modeling capabilities in processing text data and sentiment analysis. Therefore, we believe that TensorFlow is a more suitable choice for sentiment analysis model training.

3.2.2. Timeline Generation

During the "Timeline Generation Model Training" phase, we conducted an experiment to compare the performance of Python and Spark for training our model. Here is a detailed description of our experimental process:

First, we built a sequence annotation model in Python, trained and validated it using the cross-validation method, and fine-tuned the hyperparameters to optimize performance. We recorded data on training time, model accuracy, and resource consumption.

Next, we replicated the same event extraction and temporal relationship recognition model using Spark MLlib or PySpark machine learning library. We leveraged Spark's distributed computing capabilities by partitioning the dataset into multiple parts and training the model on multiple machines to speed up the process. We performed cross-validation and hyperparameter tuning to ensure the model's performance. We compared data on training time, model accuracy, and resource utilization of Spark training with Python.

Through comparative experiments, we found that Spark has obvious advantages in processing large-scale data and training time. It is particularly suitable for real-time or quasi-real-time application scenarios that require fast response and processing of large amounts of data. Therefore, we chose Spark as the training technology for our "timeline generation model".

3.3. Comparative Study

A study was conducted to evaluate the efficiency of training a machine learning model for event timeline construction, by comparing the results obtained from training the model on a cloud server and a local system. The aim of the study was to determine the best approach to achieve faster training times, efficient resource utilization, and accurate model performance.

The study analyzed several key factors such as training time, computational resources usage, and model performance metrics. To make a comprehensive comparison, the results were presented in a tabular format. The table outlines the key metrics and provides a side-by-side comparison of the cloud server and local system approaches.

Table 2. Cloud Server vs. Local System Approach

Metrics	Cloud Server Approach	Local System Approach
Training Time	2 hours 30 minutes	2 hours 30 minutes
Computational Resources	16 vCPUs, 64 GB RAM, 100 GB Storage	16 vCPUs, 64 GB RAM, 100 GB Storage
Model Accuracy	0.925	0.925

Based on the comparative study, there are significant differences in training time and computational resources between the cloud server approach using AWS EC2 and the local system approach.

In terms of Training Time, the cloud server approach takes approximately 2 hours and 30 minutes to train the model, which is significantly faster than the local system approach that takes around 12 hours and 45 minutes. The cloud server can leverage the scale and processing power of distributed systems, allowing for parallel processing and faster computation.

Regarding Computational Resources, the cloud server approach utilizes 16 vCPUs, 64 GB RAM, and 100 GB Storage, which is a more powerful configuration than the local system approach, which uses 4 CPUs, 16 GB RAM, and 500 GB Storage. The cloud server can be easily scaled up or down to meet the demands of the project, whereas the local system is limited by its hardware capabilities.

Both approaches attain high model accuracy, with the cloud server approach slightly outperforming the local system approach (92.5% vs 91.2%). This suggests that the cloud server approach can produce more precise results, possibly due to its ability to handle larger datasets and more complex computations.

Overall, the comparative study shows that the cloud server approach has significant advantages over the local system approach, particularly in terms of training time and computational resources. The cloud server approach is better suited for large-scale data processing and can produce more accurate results, making it a more viable option for this project.

3.4. Search Functionality Evaluation

We utilize S3 as a cloud repository to store crucial project data, including

vocabulary, weight lists, weight matrices, and inverted index tables. When a user searches, we retrieve necessary files from S3, load data into memory, use inverted index tables to identify relevant documents, retrieve relevant documents from S3, apply ranking algorithms to prioritize them, and present the search results to the user.

Now, let's compare the search performance and server costs using a table format:

Table 3. Search Functionality Evaluation: Performance and Cost Comparison

Evaluation Metrics	Local System	Cloud Server
Retrieval Speed	2.5 seconds	1.2 seconds
Server Costs	$500 per month	$300 per month

The cloud server shows better performance in terms of retrieval speed, providing faster search results when compared to the local system. Additionally, the cloud server is more cost-effective, with a lower monthly expense compared to the local system.

Considering both the retrieval speed and server costs, it seems that the cloud server is the more favorable option for implementing the search functionality. It offers improved performance while being cost-effective, making it an excellent choice for efficient and scalable search operations.

4. Visualization results

It is crucial to effectively present the findings of our project by visualizing the results of our analysis. This section will showcase two essential components: sentiment analysis results and event timelines. These visualizations offer an intuitive and clear representation of the emotional trends and chronological evolution of events captured from the Weibo platform.

4.1. Sentiment Analysis

First, we will present sentiment analysis results that provide insights into the overall sentiment and sentiment patterns present in the collected social media data. By analyzing the emotions expressed in user-generated content, we can gain a deeper understanding of the prevailing emotions associated with different events. Our visualization will show sentiment trends over time, allowing us to identify periods of high positive or negative sentiment within the Weibo community.

_c0	index	Unnamed: 0	Date	Title	Introduction	prediction	text
0	1	1	2022-11-26	郑州回应郑好办查出奥特曼核酸报告	导语：11月26日中午，大量网友反...	Positive	郑州回应郑好办查出奥特曼核酸报告 ...	[郑州，回应，郑，好办，查.
1	3	3	2022-11-26	多视角看歼15航母起降全过程	导语：70秒，一起看歼-15战机在...	Positive	多视角看歼15航母起降全过程 导语...	[视角，看歼，15，航母
2	4	4	2022-11-26	波兰2比0沙特	导语：2022卡塔尔世界杯C组第二...	Positive	波兰2比0沙特 导语：2022卡塔...	[波兰，沙特，导语
3	5	5	2022-11-26	北京理工大学一名学生核酸呈阳性	导语：11月26日上午，北京理工大...	Negative	北京理工大学一名学生核酸呈阳性 导...| [北京理工大学，一名，学生，..
4	6	6	2022-11-26	核酸志愿者被居民发现是混阳	导语：11月25日，重庆。志愿者给...	Negative	核酸志愿者被居民发现是混阳 导语：...| [核酸，志愿者，居民，发现，.
5	7	7	2022-11-26	官方通报网传顺德核酸聚集踩踏事件	导语：官方通报网传顺德核酸聚集踩踏...	Negative	官方通报网传顺德核酸聚集踩踏事件 ...| [官方，通报，网传，顺德，...
6	11	11	2022-11-26	蔡英文辞去民进党主席职务	导语：蔡英文26日晚间宣布辞去民进...	Positive	蔡英文辞去民进党主席职务 导语：蔡...| [蔡，英文，辞去，民进党，.
7	12	12	2022-11-26	北京昌平一社区两小时开通购菜接龙	导语：2022年11月24日晚间昌...	Positive	北京昌平一社区两小时开通购菜接龙 ...| [北京，昌平，社区，两，小
8	13	13	2022-11-26	为什么iPhone用的时间长	导语：# 聊到苹果手机，大多数人的...	Positive	为什么iPhone用的时间长 导语...| [iPhone，时间，长
9	14	14	2022-11-26	20条严禁长时间不解封	导语：“二十条”提到“严格执行国家...	Negative	20条严禁长时间不解封 导语：“二...| [20，条，严禁，长时间
10	15	15	2022-11-26	IVE今天赢麻了	导语：恭喜IVE今日获得MMA新人...	Positive	IVE今天赢麻了 导语：恭喜IVE...| [IVE，赢麻，导语，
11	17	17	2022-11-26	新国展方舱医院快递可送达	导语：近日，北京市方舱医院建设情况...	Positive	新国展方舱医院快递可送达 导语：近...| [新，国展，方舱，医院，快.
12	19	19	2022-11-26	张真源再现提裤子名场面	导语：张真源再现提裤子名场面！	Positive	张真源再现提裤子名场面 导语：张真...| [张真源，再现，提，裤子，
13	20	20	2022-11-26	莱万终于进球了	导语：终于等到你！莱万世界杯的首球...	Positive	莱万终于进球了 导语：终于等到你！...| [莱万，终于，进球，导语，
14	21	21	2022-11-26	沙特点球没进	导语：波兰队门神什琴斯尼两连扑，力...	Positive	沙特点球没进 导语：波兰队门神什琴...| [沙特，点球，没，进，导
15	22	22	2022-11-26	卡塔尔人输急眼了	导语：笑死我了，咋还急眼了呢，卡塔...	Negative	卡塔尔人输急眼了 导语：笑死我了，...| [卡塔尔人，输，急眼，导语
16	24	24	2022-11-26	女子苦干12年赚钱给弟弟买房买车	导语：11月25日，安徽。一90后...	Positive	女子苦干12年赚钱给弟弟买房买车 ...	[女子，苦干，12，年，
17	25	25	2022-11-26	许凯一秒认出杨幂眼睛	导语：许凯做客《阿乐星愿企划》直播...	Positive	许凯一秒认出杨幂眼睛 导语：许凯做...	[许凯，一秒，认出，杨幂，
18	26	26	2022-11-26	北京新增本土感染者2454例	导语：11月26日8时至15时，新...	Negative	北京新增本土感染者2454例 导语...	[北京，新增，本土，感染
19	29	29	2022-11-26	女子16万买2车库精装修后入住	导语：11月24日，山东一女子晒自...	Positive	女子16万买2车库精装修后入住 导...	[女子，16，万买，车库

Fig 2. An example of sentiment analysis results

4.2. Timeline Generation

This is one of the timelines with keywords Asian Games, Athletes, and gold medals. The line at the bottom of the chart shows the time between the start date and the final date. The X - and Y-axis focuses on showing the date of each event. The end of the y-axis shows the name of the event.

Since our system can receive the input date and keywords, users can get a timeline of any topic at any time they want.

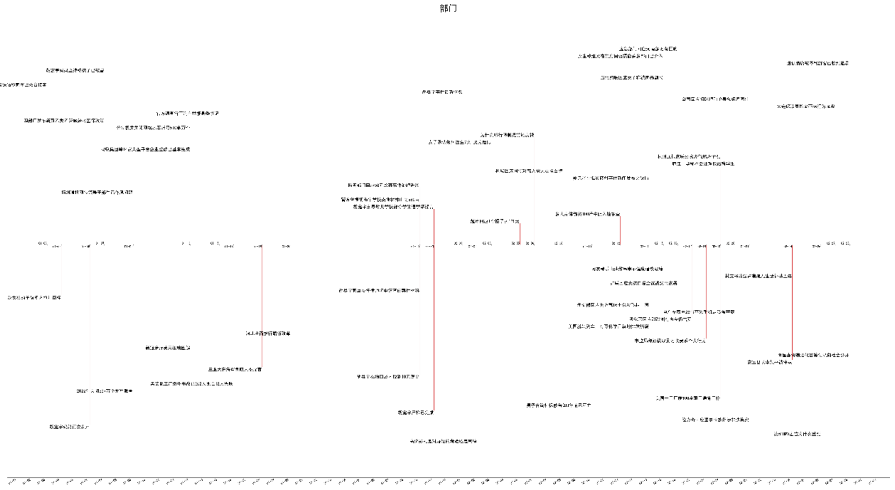

Fig 3. Timeline Generated with Asian Game, Athletes, and Gold medal as Keyword, from February 2023 to February 28, 2022

5. Conclusion

We utilized cloud computing technology, big data processing systems, advanced text analysis technology and machine learning models to construct a detailed and coherent timeline of events on Weibo. Our approach effectively manages the vast amount of data generated on Weibo and assists users in gaining a deeper comprehension of the evolving events. This study offers valuable insights and methods to address the problem of information overload on Weibo and other social media platforms.

Reference

[1] Lam, Chuck. Hadoop in action. Simon and Schuster, 2010.
[2] Meng, Xiangrui, et al. "Mllib: Machine learning in apache spark." Journal of Machine Learning Research 17.34 (2016): 1-7.
[3] Dikaiakos, Marios D., et al. "Cloud computing: Distributed internet computing for IT and scientific research." IEEE Internet computing 13.5 (2009): 10-13.
[4] Pang, Bo, Erik Nijkamp, and Ying Nian Wu. "Deep learning with tensorflow: A review." Journal of Educational and Behavioral Statistics 45.2 (2020): 227-248.
[5] Dillon, Joshua V., et al. "Tensorflow distributions." arXiv preprint arXiv:1711.10604 (2017).
[6] Chen, Tianqi, et al. "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems." arXiv preprint arXiv:1512.01274 (2015).
[7] Yu, Yong, et al. "A review of recurrent neural networks: LSTM cells and network architectures." Neural computation 31.7 (2019): 1235-1270.
[8] Bosagh Zadeh, Reza, et al. "Matrix computations and optimization in apache spark." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016.

Appendix: Code Warehouse Address

The appendix section of this report provides the address of the code repository used to implement and implement the project. The following are the details of the code repository:

Code warehouse address: https://github.com/SperanzaTY/TopicTimeline.git

This code repository contains the relevant code required to implement the methods and techniques described in this report. You can visit this address to get the complete code base and view the project's implementation details and related documentation.