A data-driven framework for archiving and exploring social media data

Qunying Huang and Chen Xu

Yongqi An, 20599957
Oct 18, 2016
Introduction

• Social media applications are widely deployed in various platforms from personal computers to mobile devices.
• Usages: marketing, intelligent ‘geo-sensor’ (Asiana Flight 214 accident), coordinating activities (disaster relief).
• Potentially invaluable due to its obvious subjective nature.
• Social media example: Twitter
  • Publish huge volumes of user-generated data daily.
  • Service design causes astonishing versatility in contents.
  • Almost a norm for public relationship management to monitor closely in the Twitter sphere.
• Social media data present challenges in four dimensions:
  • Volume
  • Velocity
  • Variety
  • Veracity (non-technical)

• Goal: present a prototype to address the demands on high performance computing framework for processing social media data timely and effectively.
Literature Review

• Traditional surveys vs. Twitter data.
• Social media data can be used to detect events.
• For big data management:
  • Non-traditional methodologies: NoSQL, scalable SQL
  • Computational infrastructures: traditional cluster computing, grid computing --> cloud computing, CPU/GPU heterogeneous computing
• In big data and geospatial computing field, cloud computing has been increasingly viewed as a viable solution.
• Data have to be managed in a way that is suitable for distributed parallel processing.  => NoSQL

• Problems with traditional GIS:
  • Designed and implemented to target stand-alone single computers.
  • Lacks the capacities to process new data types.

• Problems with Twitter data:
  • Majority of data have no explicit geographic location information.
  • Much useful information in Twitter is textual only.

• One previous study.

• Integrating cloud computing and NoSQL is prominent.
Methodology

• Basic type of account gives access to 1% of total real-time tweets.
• Approach starts with data access, and then data are stored locally and processed later for analysis purpose.
• Three primary strategies:
  • Data sets are archived as different collections rather than a table, each collection could have many document entries (e.g. temporal, spatial region).
  • Parallel computing is applied to harvest, query and analyze tweets to or from different collections simultaneously.
  • Data are duplicated across multiple servers to support massive concurrent access of the data sets.
Methodology – System Architecture

- Data resources: in this study is Twitter.
- Data repository: responsible for archiving and retrieving data sets.
- Data processor: provides data processing and analytical functions.
- Data client: provides information analysis for end users through visualization or animation with interactive tools.
System Architecture - Data repository

• Data repository layer is designed to store and manage the harvested social media data sets.

• Two primary reasons to use MongoDB instead of RDMS:
  1. In traditional RDMS, each record has the same structure.
     • Tweets are not uniform and structured in nature.
     • Document DBs use JSON.
  2. Traditional relational DBs are not designed to deal with the scale and agility goals of modern applications.
     • RDMS has to find all relevant data and link them all to generate query results.
     • Alternative solution causes problems.
     • Queries in MongoDB only look for keys and values in a document, which can be easily distributed across different servers.
System Architecture - Data processor

• Data processor layer is used to receive and parse a specific data request from the client, and performs the functionalities of accessing, querying and analyzing.

• Parallel computing here is used at two levels:
  • Process level
  • Client level
Methodology – Workflow

- Request parser: parse and extract the query parameters, create a new service task and add it into the service request queue.
- Request integrator: schedule and dispatch tasks to the data manager.
- Response generator: integrate query results from the data manager module, perform analytical functions, and send JSON to web portal.
Methodology – Workflow

- Data archiver: access and receive real-time data from network systems.
- Data preprocessor: preprocess data collections in MongoDB to facilitate data query and access (e.g. tokenization, building up indexes).
- Data manager: disseminate the request to related data collections on different servers.
Methodology – Workflow

- Multiple data servers can be leveraged to handle the big social media data.
- MongoDB naturally supports master-slave replication:
  - Master: store data, write operations
  - Slave: automatically synchronize with master, read operations
Experiments - Performance analysis

• Three servers: database is configured on two servers (master && slave), the other serves as the client.

• Three groups of tests are performed, including:
  • Spatiotemporal query
  • Concurrent query
  • Sentiment analysis
Performance analysis - Spatiotemporal query

- 1, 2, and 4 threads are issued to handle the query of targeted tweets from 0 to 160 million collection that is hosted on 1 or 2 DB servers.
- A query from one user is decomposed into multiple sub-queries with each sub-query handled by one thread searching a subset of data sets.
Performance analysis - Spatiotemporal query

- 1 thread 1 server:
  - only one query
  - query 5, 10, 40, 80 and 160 million record collection

- 2 threads 1 server:
  - two queries are constructed
  - each query directed to a 2.5, 5, 20, 40 and 80 million record collection

- 2 threads 2 servers:
  - two queries are constructed
  - each query directed to a 2.5, 5, 20, 40 and 80 million record collection
  - one query is handled by the master DB server, and the other is sent to the slave
Performance analysis - Spatiotemporal query

- Takes around 62, 186, 268, 503 and 1179ms using 1 thread 1 server.
- Improved to 31, 50, 150, 292 and 631ms using 2 threads 1 server.
- Reduced to 20, 36, 96, 201 and 402ms using 4 threads 1 server.
- Using two DB servers does not have much gains.
Performance analysis - Spatiotemporal query

• Compared to one-thread, two- and four-threads querying from one server can improve performance by 45% and 63% in average.
Performance analysis - Concurrent query

• Tests the scenarios of one and two servers to process 10, 20, 40, 80, 160 and 320 concurrent requests.
• 1 server takes around 0.747, 1.369, 2.871, 6.554, 17.048 and 29.041s.
• 2 servers use about 0.516, 0.799, 1.616, 3.127, 7.313 and 14.469s.
• 2 servers can reduce the average response time for each user query by a factor of 45% in average.
Performance analysis - Sentiment analysis

- Not too much difference with multiple servers.
- 1 thread uses about 2.8, 5.5, 19.3 and 38.7 minutes.
- 2 threads use about 1.5, 3.0, 12.0, and 22.6 minutes.
- 4 threads uses about 1.1, 2.0, 7.7, and 14.6 minutes.
- Compared to one-thread scenario, two- and four- thread can improve the performance by a factor of 40% and 60% in average.
Experiments – Prototype

- A prototype has been implemented based on JAVA and JSP to demonstrate how to archive, retrieve and process social media data based on the proposed framework.
- Twitter4j
- Google Maps API
- LingPipe
Visualization Example

Steps:
- set up the input parameters of the query
- After obtaining query results back, visualize the results
Sentiment Analysis Example

- Results indicate that almost twice users hold a negative attitude towards such a natural disaster event.
Conclusion

• Proposed and prototyped a data-driven framework to address the big data archiving, retrieving and computing by leveraging non-relational DB and parallel computing.

• Results show that the proposed framework can facilitate big data query and analysis with an improvement of more than 40%.
References


Discussion - Strengths and Weaknesses

• Strengths:
  • Reasonable methodology
  • Clear architecture & workflow
  • Great comparison between Traditional DBs and MongoDB
  • Well-designed experiments

• Weaknesses:
  • Spent too much on Introduction and Literature Review
  • No description of the prototype system
  • Lacks depth in explaining experiment results
  • Spelling mistake.. 😞
Discussion - Related Papers

• Requirement of integrating cloud computing and NoSQL:

• Reasons for using MongoDB:
Discussion - Future Work

• Cloud storage could be investigated and leveraged for the massive social media data archiving.
• Integrate elastic computing power for handling the computing demands from mining big social media data.
• Investigate other computational infrastructures to improve computing performance.
Discussion - Additional Questions I

- It can be observed that using two servers does not have much gains according to the graph below, what is the possible reason here?
- Is it still necessary to use two when making spatiotemporal queries?
Discussion - Additional Questions II

- In sentiment analysis, it does not make too much difference with multiple DB servers either. Why?
Thanks! 😊