<https://github.com/Alona-Aldushyna/Summer_2023_August_Repeat_Assessment.git>

*Detailed information about the performed data storage and processing activities, including data preparation and data processing in the MapReduce framework.*

Storage and processing of data in the MapReduce environment is carried out in several stages. Data preparation:

• The data to be processed with MapReduce must be in a specific format. The input data is usually text files containing records,with keys and values separated by a special separator such as a tab or comma.

• If the data is in a different format (CSV, JSON, XML)it must first be converted to format suitable for MapReduce processing. Various tools and libraries will help with this, such as Pandas for processing CSV files, or libraries for working with JSON and XML.

• After pre-processing, the data can be split into parts and distributed across the cluster, so that processing occurs in parallel on several nodes.

2. Mapper (Mapper):

• The mapper step is the first step in the MapReduce process. Each entry of the input data is divided into key-value pairs (key value). The mapper receives one or more of these pairs as input and applies a specific function (mapping) to each of them.

• In the tweet processing task, the mapper can take lines from a CSV file and extract users and tweet texts from them. The mapper can then preclean the texts of the tweets by removing unwanted characters and converting the texts to lowercase. The result for each user is a "user-cleared\_tweet\_text" pair.

3. Sorting and merging (Shuffle and Sort):

• After the mapper has processed all its inputs, the sort and merge phase occurs. In this step, key-value pairs from all mappers are sorted by key and grouped by the same keys. This is necessary so that all records with the same keys end up on the same reducer.

• In our case, records for each user will be grouped together.

4. Reducer:

• The reducer stage is the second step in the MapReduce process. The reducer receives as input a group of key-value pairs, grouped by the same keys. It then applies a specific function (reducing) to each group.

• In our example, the reducer simply collects all of the cleaned tweet texts for each user and outputs them to a new file, keeping the cleaned tweets for each user on separate lines.

5. Saving the result:

• After processing all the data, the reducer outputs the processing result, which can be saved to new file, passed on for additional processing.

The general workflow of MapReduce is that data is split into blocks, mappers process each block in parallel, the results are sorted and merged and then the reducers are applied to the grouped data to get the final result.

Showing how a Hadoop streaming program can run Python as a MapReduce application on Hadoop Cluster, the Top 10 Users application can be implemented as two Python programs: mapper.py and reducer.py, mapper.py is the Python program that implements the logic for mapping the top ten users.

This code is simple Python program that uses a MapReduce-like approach to aggregate data and find the top 10 users with the highest number of mentions.

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Значок на компьютере

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

Автоматически созданное описание

Similarly, we create a mapper and a rediuser to clean tweets from unwanted characters and reduce them to lowercase. We then pass in the cleared tweet text as the value, leaving the username as the key. The reducer simply outputs the cleaned tweet texts without modification:

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

*Discussing the rationale and justification for the choices made in terms of data processing and storage and the choice of programming language, machine learning models and the algorithms that were implemented.*

Justification for the choice of data processing and storage in Hadoop MapReduce:

The choice of Hadoop MapReduce for data processing in a 1 600 000 tweet project is due to the following reasons:

1. Scalability: Hadoop MapReduce is highly scalable allowing you to process large amounts of data. In this project, the number of tweets is large, so Hadoop MapReduce allows you to distribute data processing to cluster of nodes, which reduces the execution time of tasks.

2. Fault Tolerance: Hadoop MapReduce provides fault tolerance due to its distributed nature and the ability to automatically restart nodes on failure. this is important for processing large amounts of data minimize the risk of losing results.

3. Distributed Computing: Hadoop MapReduce allows you distribute computations to different cluster nodes, which increases the performance and efficiency of data processing.

4. HDFS integration: Hadoop MapReduce natively integrates with Hadoop, which provides easy storage and access to data in the cluster.

5. Ecosystem: Hadoop offers an extensive ecosystem of tools such as Pig, HBase, Spark, and others that can be used for advanced data processing and analysis.

Justification for the choice of programming language - Python:

1. Simplicity and Convenience: Python is a simple and straightforward programming language, which makes it attractive for rapid development and prototyping.

2. Large Community and Libraries: Python has a huge active developer community and extensive library, including libraries for data processing (eg Pandas, NumPy), machine learning (TensorFlow, Keras, scikit-learn), and text analysis (NLTK).

3. Support for Hadoop MapReduce: To implement Hadoop MapReduce tasks in Python, convenient libraries such as MRJob are available that make it easy to create mappers and reducers in Python.

Justification for the choice of machine learning models and algorithms:

1. Time series and Dickey-Fuller test:

For time series analysis, such as the number of tweets at different points in time, time series analysis is used, including the Dickey-Fuller test to test for stationarity.

2. Sentiment Analysis:

To analyze the sentiment of textual data such as tweets, sentiment analysis techniques can be used to determine whether the text is positive, negative, neutral.

3. ARIMA:

ARIMA (Autoregressive Integrated Moving Average) is a statistical time series forecasting method that takes into account the dependence of the current value of the series on the previous values and uses autoregression and moving average.

4. MinMaxScaler:

MinMaxScaler data scaling allows you to scale your data to a range of 0 to 1, which can improve the performance of some machine learning algorithms.

5. LSTM model, tensorflow keras GRU, Conv1D:

LSTM model: The LSTM model was chosen because it is well suited for the processing and analysis of text data and sequential time series. Tweets are text data, and LSTM can highlight dependencies and patterns in word and phrase sequences, which can help analyze text sentiment (positive, negative, neutral) and predict user activity or interest based on text data.

TensorFlow Keras GRU: GRU (Gated Recurrent Unit) is a simplified version of LSTM and it can be chosen when data is limited and performance and resource usage are important. While LSTM might be too complex for a given size of data, GRU might be a good trade-off between complexity and performance. GRU can also do well with text data and time series analysis.

Conv1D: Conv1D can be useful for analyzing textual data, especially if you want to highlight important features and patterns in a sequence of words or characters. Conv1D can also be used to detect features in time series, which can be useful for predicting user activity based on their behavior over time.

Choosing to use all of these models gives you the ability to compare their performance and accuracy in your specific tweet analysis task. LSTM and GRU are typical recurrent neural networks suitable for processing serial data, while Conv1D can be an interesting experiment to analyze textual data in a one-dimensional form.

*Benchmarking for MongoDB and MySQL using YCSB for benchmarking.*

YCSB is a reference tool developed by Yahoo. It provides a standard way for the user to evaluate the system under normal workloads and focus on performance and scaling.

It is often used to compare the relative performance of NoSQL database management systems.

Let's do a comparative analysis of two databases: mysql vs mongodb, using the ycsb benchmarking tool, having a dataset containing 1,600,000 tweets, pre-processed tweets and removed the signs.

Loaded a data set with 1,600,000 tweets into both databases as shown below.

Изображение выглядит как текст, программное обеспечение, веб-страница, Веб-сайт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

After running some tests with the benchmarking tool using YCSB, for MongoDB and MySQL, we got the following results:

For MySQL:

Изображение выглядит как текст, снимок экрана, документ

Автоматически созданное описание

For MongoDB:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Let's analyze the received reports and compare MySQL and MongoDB databases based on the data provided:

1. Bandwidth (Throughput):

• MongoDB: Throughput is about 1094 operations per second.

• MySQL: Throughput is about 1392 operations per second.

Based on test results, MySQL shows higher throughput than MongoDB.

2. Runtime (RunTime):

• MongoDB: Execution time is 914 milliseconds.

• MySQL: Execution time is 718 milliseconds.

MySQL also exhibits lower execution times compared to MongoDB.

3. Garbage Collection:

• MongoDB: Relative time spent on garbage collection is 0.55%.

• MySQL: Relative time spent on garbage collection is 0.56%.

Both database engines show similar results in terms of garbage collection time.

4. Complexity of insert operations (INSERT):

• MongoDB: Average insert execution time is about 385 microseconds, with a minimum of 83 microseconds and a maximum of 101503 microseconds.

• MySQL: The average insert time is about 369 microseconds, with a minimum of 198 microseconds and a maximum of 27775 microseconds.

According to the results of testing,insert operations in MongoDB and MySQL are approximately equal in complexity and execution time.

So, according to the results of the comparative analysis:

• MySQL provides higher throughput and faster operation times than MongoDB.

• Both database engines show similar garbage collection performance.

• Insert operations in both databases are performed with approximately the same complexity and execution time.

Key differences between MongoDB and MySQL

MySQL is a relational database management system, and MongoDB is a NoSQL database system.

MySQL uses SQL, which most developers are familiar with. Conversely, MongoDB uses the MongoDB Query Language (MQL). Although there are similarities between MQL and SQL, learning MQL usually requires additional effort.

Next, we will consider some other key differences.

The data model

MySQL -relational database system that stores data in columns, rows, and tables. Data is stored in rows, and each column represents different types of data. Then you define relationships between the data using foreign and primary keys. Each table contains a primary key that is used to identify it, and a foreign key creates a relationship.

MongoDB is a document-oriented database in which all data is stored as binary JSON (BSON) documents. BSON allows you to serialize multiple forms of data. Using BSON documents allows you to store unstructured, semi-structured and structured data. Instead of a database schema, MongoDB takes a flexible approach by storing documents in collections.

Scalability

In the MySQL database system, the available scaling options are limited. You can choose one of the following options:

• Vertical scalability by adding additional resources to the current database server

• Read replication by creating read-only copies of the database on other servers

Read replication is limited to a maximum of five copies. Replicas can also lag behind the primary copy, which leads to large-scale performance problems. Vertical scalability is also limited by the infrastructure used.

In contrast, MongoDB's design provides a significant advantage in terms of scalability. The solution has two key functions for scaling:

• Replica sets – groups of MongoDB servers containing identical data

• Segmentation – different parts of data distributed across different servers

MongoDB allows you to create segmented clusters,so parts of your data are replicated across multiple servers. For example, if you have a large number of customer records, you can distribute them so that names A through J and names K through Z are in their own set of replicas. Thus, MongoDB can be scaled horizontally to optimize reading and writing performance at the required scale.

Productivity

The MySQL solution is intended for high-performance combined several tables that are properly indexed. However, it is necessary to insert data by time, so the recording speed decreases.

MongoDB documents follow a hierarchical data model and store most of the data in one document, which reduces the need to combine multiple documents. Joins are supported by the $lookup operation, but they are not optimized for performance. However, MongoDB offers an insertMany() API for fast data insertion with a priority on write performance.

Flexibility

As a relational database management system, MySQL has a more rigid structure than MongoDB. MySQL uses a fixed schema and organizes data into a string and a table. To use MySQL, you need to structure the data and place it in a tabular system.

By storing data in the form of JSON, MongoDB allows you create complex applications with different types of data, Example, you can create new fields by updating the fields of a nested array. You can also use the aggregation pipeline - a MongoDB function that allows you to transform data by combining several operations into one workflow.

*The analysis of any changes is configured, occurring during the period*

*2009/04/06 – 2009/04/29*

Изображение выглядит как линия, График, диаграмма, текст

Автоматически созданное описание

The analysis of changes configured on the basis of the graph of the provided code allows the following conclusions to be drawn:

The adjusted analysis was conducted for the period from April 6,2009 to April 29,2009.

Relative number of tweets:

The graph shows the number of tweets with different sentiments (positive, negative and neutral) for each date in the selected period.

The graph shows that on April 19, 2009, there was a significant surge of positive and neutral tweets, which may indicate a positive event or user activity on that day.

You can also easily notice the low activity on the schedule on April 17, 2009: on this day, relatively low activity of tweets with different moods is noted.

***Next, I looked at the forecasts for 7,30,90 days in 3 other Jupiter notebooks: Keras, Arima and Forcast, which I also added to the GIT repository. Sorry for the inconvenience.***

For forecasting time series for various forecast horizons (7, 30, 90 days), we used various methods and models. Here is a brief description of each method:

ARIMA (Autoregressive Integrated Moving Average):

ARIMA is a classic statistical method for time series analysis and forecasting. It uses a combination of autoregression (AR), integration (I) and moving average (MA). We used the statsmodels library to train ARIMA models and get 7, 30 and 90 day forecasts.

the results:

Изображение выглядит как текст, снимок экрана, График, линия

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, График, линия

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, График, линия

Автоматически созданное описание

Forecaster (Forecasting using ARIMA):

sktime provides a user-friendly interface for time series forecasting using ARIMA and other forecasting methods. In this case, we have used ForecasterAutoreg, which automatically converts ARIMA to a convenient format for working with sktime. Here we trained the model on 30 lags (previous values) and received forecasts for 7, 30 and 90 days.

Изображение выглядит как текст, линия, Шрифт, График

Автоматически созданное описание

Изображение выглядит как текст, линия, График, Шрифт

Автоматически созданное описание

Autoreg (Prediction using Convolutional Neural Networks):

We have used tensorflow.keras.layers.Conv1D to create a neural network for time series prediction. Convolutional Neural Networks (CNN) have the ability to extract information from time sequences and can be effective for prediction. In this case, we used 1D convolutional layers to predict 7, 30, and 90 day values.

Изображение выглядит как диаграмма, линия, График

Автоматически созданное описание

For stationary time series, several other forecasting methods could be applied:

SES (Simple Exponential Smoothing) - Simple exponential smoothing. It is a method that uses exponential smoothing to predict time series without a trend and seasonality.

Holt-Winters - Triple exponential smoothing. This method is used to predict time series with trend and seasonality.

ARMA (Autoregressive Moving Average) - Autoregressive moving average. This method models the time series as a linear combination of an autoregressive and a moving average component.

SARIMA (Seasonal Autoregressive Integrated Moving Average) - Seasonal autoregressive integrated moving average. This method extends ARIMA for modeling seasonality in data.

TBATS (Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components) - A method that combines trigonometric seasonality, Box-Cox transformation, ARMA errors and trend and seasonal components.

Regression-Based Forecasting - You can also apply multiple regression or machine learning methods to predict time series using other variables as predictors.

In general, the analysis carried out allowed us to successfully predict the mood of the time series for various periods in the future,which can be useful for business decision making and planning.