WikiStats: Project Proposal

Sidharth Bambah, Columbia University sb4283
Jeswanth Yadagani, Columbia University jy3012

Overview

Wikipedia is one of the largest multilingual encyclopedia's in the world. It is created and maintained as an open collaboration project by a community of volunteer editors using a wiki-based editing system.

Every day new articles are being published and existing ones updated leading to a constant information flow throughout Wikipedia. Being such a large community with millions of articles on nearly any subject, it would be quite useful to analyze the true size of Wikipedia's community and the dynamic nature of the content available on the website.

Objective

This project, WikiStats, aims to create a Wikipedia monitoring platform that is easily understood and accessible. Using a large scale stream processing system coupled with a dynamic web frontend hosted on the cloud, WikiStats will provide valuable statistical information regarding the fluid nature of articles in the encyclopedia.

Particularly, the system intends to monitor the creations and changes of articles by various members of the Wikipedia community. This information will be used to produce robust visualizations that describe the impact of the community's modifications on the site as a whole.

The intended audience for this work is the Wikipedia community, particularly its software developers, resource allocation groups, and designers. Additionally, the platform can be of great importance to public policy makers and news organizations due to its ability to show which topics are the most modified and hence, consequential at the time.

Implementation/Scope

Data Source

WikiStats will make use of the Recent Changes datastream provided by WikiMedia as a subset of the Web APIs for Wikipedia. This stream utilizes the EventStreams web service, which exposes a continuous stream of structured event data. This is done over HTTP using chunked transfer encoding through the Server-Sent Events (SSE) protocol.

Tools

In order to consume, manipulate, and analyze the stream provided through Wikipedia's API, this system will utilize a diverse array of tools.

A Kafka broker will be configured using a Google Cloud Compute engine. Then, a Python script will use the SSEClient library to consume the stream and publish each event to the Kafka topic.

Additionally, Apache Spark Streaming will be used to ingest the incoming messages on the Kafka topic and pass the events through a stream processing pipeline for analysis.

The analyzed and aggregated statistics will be passed to the Web-based frontend, written with the ReactJS framework, and visualized using charting libraries, such as ChartJS and D3.

Output

The final output of this project will be a cloud deployed dynamic web application that utilizes the RecentChanges feed to produce useful statistics for the entire Wikipedia article corpus.

The visualizations on the web page will show if Wikipedia as a whole is growing or contracting and should provide distinct information for each individual wiki within Wikipedia. These visualizations will also dynamically update to provide up-to-date information as soon as the summary statistics are computed. The user will also have the option to select which wiki he wants the statistics for allowing for quite fine-grained control.

Milestones

Step 1: Data Processing Pipeline

First, a Google Cloud Dataproc cluster will be instantiated with a few worker nodes. This will allow for Pyspark jobs to be submitted and operations to be optimally placed among the workers. Additionally, the Google infrastructure will handle fault-tolerance and replication for the system allowing for simple restarts if the system were to fail for any reason.

Next, a Python script will be submitted to the cluster. This script will consume the messages on the Kafka topic, which are brokered by the aforementioned Google Compute Engine, and convert them to a Spark Streaming Dstream object. This object will be passed through a pipeline that utilizes various algorithms to produce summary statistics for each wiki.

Step 2: Web-Based Dashboard

The ReactJS web application will be built using a collection of dynamic components. It should receive the statistics produced by the Dataproc job over a WebSocket and pass the information to the charting components written with ChartJS and D3. This will allow for near real-time updates of these statistics providing a comprehensive view of the current changes happening across Wikipedia.

Step 3: Cloud Deployment

Upon completion of the entire platform, the system will be deployed to the cloud for public access.

After the ReactJS is compiled, it will be uploaded to an Amazon S3 bucket and allowed public web access. Additionally, the Spark jobs will continue to run on the Dataproc cluster and the Kafka broker will remain live on the Google Cloud Compute virtual machine.