Why Hadoop is a better platform for Large Scale Sentiment Analysis?

Kanha Shukla¹, Abhijeet Raj², and Ankush Bansal³
Galgotias College Of Engineering and Technology, Gr. Noida, India

Abstract—Sentiment analysis on Twitter data has attracted much attention recently. One of the system’s key features, is the immediacy in communication with other users in an easy, user-friendly and fast way. Consequently, people tend to express their feelings freely, which makes Twitter an ideal source for accumulating a vast amount of opinions towards a wide diversity of topics. This amount of information offers huge potential and can be harnessed to receive the sentiment tendency towards these topics. Now with the case on the desk we have multiple platforms on which the Sentiment analysis can be done like oracle, Teradata, MySQL, Hadoop. In this paper, we discuss the issue why hadoop is a better platform for large scale Sentiment analysis. With its extensively smart work division methodologies Hadoop here has emerged as the time as well as data efficient platform for performing large scale analysis.

Index Terms—Hadoop, Sentiment Analysis, data partition, fault tolerant

I. INTRODUCTION

Sentiment analysis (or opinion mining) on Twitter data has attracted much attention recently. One of the system's key features, is the immediacy in communication with other users in an easy, user-friendly and fast way. Consequently, people tend to express their feelings freely, which makes Twitter an ideal source for accumulating a vast amount of opinions towards a wide diversity of topics. This amount of information offers huge potential and can be harnessed to receive the sentiment tendency towards these topics. However, since none can invest an infinite amount of time to read through these tweets, an automated decision making approach is necessary. In the context of this work, we utilize hashtags and emoticons as sentiment labels to perform classification of diverse sentiment types. Hashtags are a convention for adding additional context and metadata to tweets. Although the problem of sentiment analysis has been studied extensively during recent years, existing solutions suffer from certain limitations. One problem is that the majority of approaches is bounded in centralized environments. Moreover, sentiment analysis is based on, it terms of methodology, natural language processing techniques and machine learning approaches. In this paper, we propose MR-SAT: a novel MapReduce Algorithm for Big Data Sentiment Analysis on Twitter implemented in Hadoop [17, 19], the open source MapReduce implementation. Our algorithm exploits the hashtags and emoticons inside a tweet, as sentiment labels, in order to avoid the time-intensive manual annotation task. After that, we build the feature vectors of training and test set and proceed to a classification procedure in a fully distributed manner. With Hadoop we have the feature of Scalability, Cost effectiveness, flexibility and fault tolerant warrants. With the data partition algorithm used in Hadoop infrastructure there is data security and backup plans. Hadoop's unique storage method is based on a distributed file system that basically 'maps' data wherever it is located on a cluster. The tools for data processing are often on the same servers where the data is located, resulting in much faster data processing. If you're dealing with large volumes of unstructured data, Hadoop is able to efficiently process terabytes of data in just minutes, and petabytes in hours. The MapR distribution goes beyond that by eliminating the Name Node and replacing it with a distributed No Name Node architecture that provides true high availability. Our architecture provides protection from both single and multiple failures. As compared to the other traditional tools hadoop is a better way to get our large scale analysis work done. This conclusion will be well proved throughout the research paper with picture and charts and table of comparisons with other platforms and hadoop.
Compare Traditional DW and Hadoop DW

**Presentation layer**
- SSAS
- PowerView
- SSRS

**Database layer**
- SQL Server
- ODBC, OLE, ADO.NET interface

**File system layer & File processing layer**
- NTFS
- Hard disk

**Hadoop D/W**
- SSAS
- ODBC Interface
- Hive, Impala, Presto, etc.

**Hadoop Root**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Without Caching</th>
<th>With Caching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis run without caching</td>
<td>9.4 sec</td>
<td>5.5 sec</td>
</tr>
<tr>
<td>Reading from local disk &amp; Computation</td>
<td>32.7 sec</td>
<td>0.4 sec</td>
</tr>
<tr>
<td>Writing to local disk</td>
<td>4.3 sec</td>
<td>5.1 sec</td>
</tr>
<tr>
<td></td>
<td>26.8 sec</td>
<td>5.1 sec</td>
</tr>
</tbody>
</table>
Graphs---

(a) RAM requested by tasks. Values are normalized by the maximum RAM available on a single node in the Google cluster.

(b) Number of tasks per job

Graph showing distributions obtained by applying hadoop infrastructure on the analysis procedure

(a) CPU task requirements

(b) Machine downtime

Graph plot showing memory management and work division while processing the threads in hadoop environment
HDFS ARCHITECTURE

### HDFS Architecture

**Client**
- **Read**: Datanodes
- **Write**: Datanodes

**Client**
- **Read**: Datanodes
- **Write**: Datanodes

**Metadata ops**
- **Metadata (Name, replicas, …):** /home/foo/data, 3, …

**Namenode**
- **Replication**
- **Blocks**

**Rack 1**
- **Read**: Datanodes
- **Write**: Datanodes

**Rack 2**
- **Read**: Datanodes
- **Write**: Datanodes

---

**Facts and Data**

Hadoop is not a database. HBase or Impala may be considered databases but Hadoop is just a file system (hdfs) with built in redundancy, parallelism. Traditional databases/RDBMS have ACID properties - Atomicity, Consistency, Isolation and Durability. You get none of these out of the box with Hadoop. So if you have to for example write code to take money from one bank account and put into another one, you have to (painfully) code all the scenarios like what happens if money is taken out but a failure occurs before its moved into another account. Hadoop offers massive scale in processing power and storage at a very low comparable cost to an RDBMS. Hadoop offers tremendous parallel processing capabilities. You can run jobs in parallel to crunch large volumes of data. Some people argue that traditional databases do not work well with un-structured data, but its not as simple as that. There are many applications built using traditional RDBMS that use a lot of unstructured data or video files or PDFs that I have come across that work well. Typically RDBMS will manage a large chunk of the data in its cache for faster processing while at the same time maintaining read consistency across sessions. I would argue Hadoop does a better job at using the memory cache to process the data without offering any other items like read
References


