Twitter sentiment analysis with Hadoop

Francesco Bongini

September 2019

Introduction

In the era of the Internet, social media has become an integral part of modern society. Social networks like Facebook, Instagram and Twitter get huge amount of data each day.
Hadoop framework stems from the need to process large amounts of data.

Figure 1: Social networks most popular

Sentiment analysis

Sentiment analysis involves to classify texts in positive, negative or neutral, often using machine learning models.

In this project, sentiment analysis is performed on a dataset of tweets from Twitter, in order to classify positive and negative tweets.

Figure 2: Classifing positive and negative tweets

Training

Given a set of examples, the algorithm learns to recognize positive and negative tweets.

The training phase is sequential. Parallel version would need a critical section to allow threads to access the shared memory. In order to guarantee the mutul exclusion, we lose the benefits of the parallelizazion.

Training with LingPipe

LingPipe library in Java allows to train the model very easily.

Algorithm 1 Sequential version

```
Inizialize classifier
For each tweet do
    classifier.handle(tweet)
EndFor
SaveModel()
```

The model is then stored and will be load in the classification phase.

Classification

The classification phase consists in classifing new tweets, not always used in the training part. Two versions:

- Sequential version
- Distributed version

Sequential version

With LingPipe the classification phase is very easy. Here is the pseudocode:

Algorithm 2 Sequential classification

```
model=LoadModel()
pos=0
neg=0
For each tweet do
    classified=classify(model, tweet).bestCategory()
    if classified==“0"
        neg=neg+1
    else
        pos=pos+1
    end if
EndFor
Return pos, neg
```

Distributed version

Distributed version is performed using **Hadoop** framework. MapReduce is a processing technique and a program model for distributed computing based on Java. The MapReduce algorithm contains two important tasks, namely Map and Reduce.

Map

Algorithm 3 Map

```
model=addCacheFile(model)
classified=classify(model, tweet).bestCategory()
**Return** classified, 1
```

The model is added to the distributed cache, in order to improve the speed.
The map function classifies the tweet and returns the sentiment classified and the value 1.

Partitioner

The partition phase takes place after the Map phase and before the Reduce phase. In this phase, all the values for each key returned from the mapper are grouped together, making sure that all the values of a single key go to the same reducer.

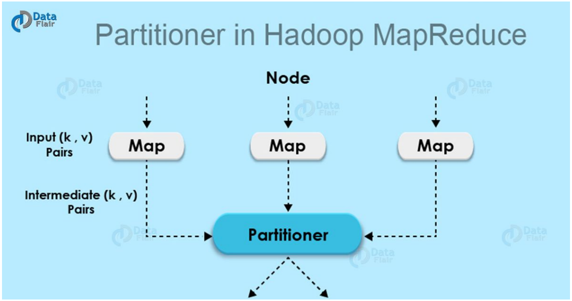

Figure 3: Classifing positive and negative tweets

Reduce

The reduce function receives the tuples from the map and counts the number of positive and negative classified tweets.

Algorithm 4 Reduce

```
count=0
For each tuple do
    count=count+1
EndFor
Return k, count
```

Amazon EMR

The performance analysis of the distributed version is performed using Amazon EMR. It provides tools for big data processing and analysis, across a Hadoop cluster.
A cluster is a collection of nodes. A node is a process running on a virtual or physical machine.

In the following analysis, each node has a r4.xlarge istance with four cores and 30.5 GB of RAM.

Speedup

Speedup is an index that measures the relative performance of two systems processing the same problem. It is defined as:

$$S_p = t_s / t_p$$

where P is the number of processors, t_s is the completion time of the sequential algorithm and t_p is the completion time of the parallel algorithm.

1000 tweets

Hadoop is not a good solution for small data.

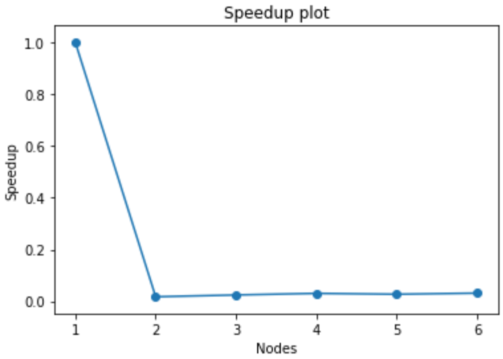

Figure 4: Speedup 1000 tweets

The sequential version is faster than the distributed one.

1.6M tweets

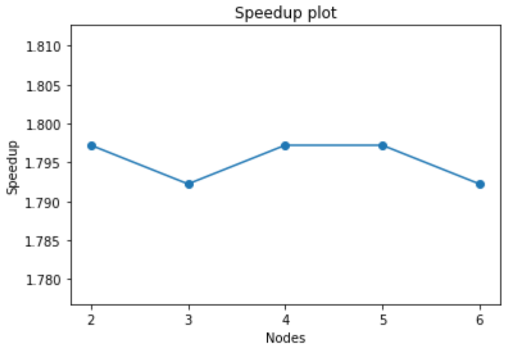

Figure 5: Speedup 1.6M tweets

By default, EMR considers just 2 mappers if the input size is only one file.

1.6M tweets in 5 input splits

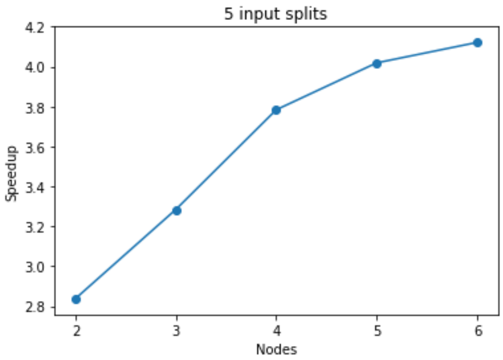

Figure 6: Speedup 1.6M tweets

For each input split Hadoop creates one map task to process records in that input split. That is how parallelism is achieved in Hadoop framework.

1.6M tweets in 10 input splits

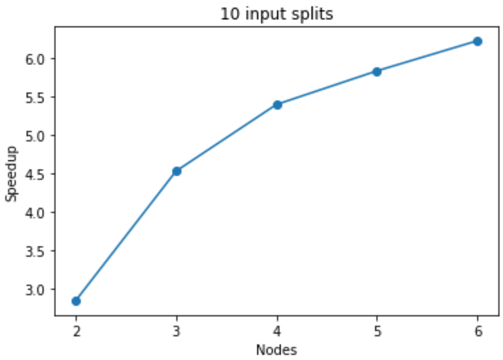

Figure 7: Speedup 1.6M tweets

Same trend of the previous example but with speedup higher.

4.8M tweets in 10 input splits

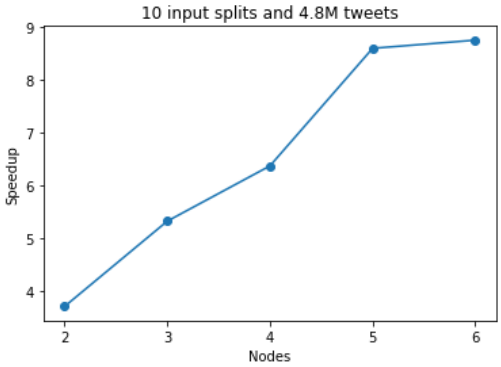

Figure 8: Speedup 4.8M tweets

4.8M tweets in 20 input splits

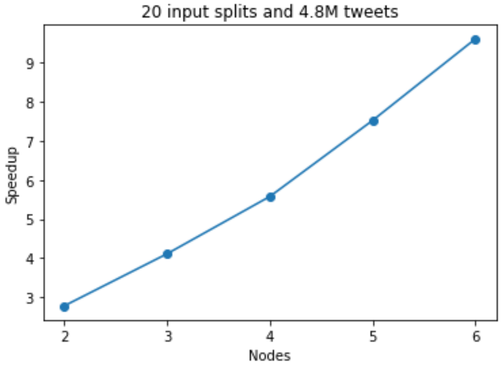

Figure 9: Speedup 4.8M tweets

Conclusions

Today, Hadoop is rapidly becoming popular as an open source database management system for processing enormous data sets using the programming tool called MapReduce.
In my experimental analysis, Hadoop dimostrated to be a valid solution for processing of huge amount of data. The most of time we reached high values of speedup.