Global Journal of Information Technology

Volume 06, Issue 1, (2016) 65-71

http://sproc.org/ojs/index.php/gjit

Detecting similar opinion holders for massive sentiment analysis

Erdem Alparslan, Bahcesehir University, Çırağan Caddesi Osmanpaşa Mektebi Sokak No: 4, Istanbul 34349, Turkey.

Adem Karahoca *, Bahcesehir University, Çırağan Caddesi Osmanpaşa Mektebi Sokak No: 4, Istanbul 34349, Turkey.

Suggested Citation:

Alparslan, E., & Karahoca, A. (2016). Detecting similar opinion holders for massive sentiment analysis. *Global Journal of Information Technology*. *6*(1), 65-71.

Received 17 January, 2016; revised 21 February, 2016; accepted 10 March, 2016.

Selection and peer review under responsibility of Prof. Dr. Adem Karahoca, Bahcesehir University, Turkey

Abstract

Sentiment Analysis is the study of acquisition, extraction and interpretation of human opinions, sentiments, attitudes and emotions from both structured and unstructured data sources. Also called opinion mining, the field is becoming crucial for various application areas including market researches, politics, sociology and economics. Therefore, many outstanding research efforts are performed on the fields including both theoretical and practical aspects. This paper aims to develop a supportive framework for sentiment analysis, focusing on the similarity of opinion holders in a massive dataset. We used e-commerce review dataset of Amazon spanning May 1996 – July 2014. The whole review set includes more than 140 million entries. As a preprocessing task each review is structured and expressed on a quadruple form of 4 dimensions: Target entity, opinion holder, sentiment and time. The aim of this study is to find out similar opinion holders for a given customer on a certain product in real time. We have defined a new method spanning all the opinions of an individual. The idea behind this calculation of similarity is rating of the same product with the same sentiment factor by two different opinion holders. The real-time calculation is also performed on Hadoop clusters. Performance enhancements and accuracy rates are then discussed.

Keywords: sentiment analysis, opinion mining, big data analytics, Map-Reduce

*ADDRESS FOR CORRESPONDENCE: **Adem Karahoca,** Bahcesehir University, Çırağan Caddesi Osmanpaşa Mektebi Sokak No: 4, Istanbul 34349, Turkey. *E-mail address*: adem.karahoca@bahcesehir.edu.tr / Tel.: +90 212 381 05 60

1. Introduction

Modern marketing strategies are mostly built on customer satisfaction and tailor-made individual customer orientation. One of the most useful information regarding customer orientation is customer review. Customers highly need the others' opinions about the product that they are looking for. In a dense environment the number of reviews which are commented on a single product may exceed hundreds or thousands. Apparently it is not possible for a customer to read and evaluate this amount of reviews at a time. We are facing the huge amount of reviews problem by this scenario. Promoting the helpful reviews for the customer is a crucial marketing effort which is often studied by especially e-marketing companies. Most of these efforts are defining most helpful reviews on a vote-basis elimination technique. The reviews are marked as "helpful" by the other customers. The assumption is: "Top rated reviews are the most helpful review for this product".

Vote-basis helpful review detection is an easy and useful idea. However, it rules out the individuals' discriminations, likings, preferences on aspects and socio-economic conditions. We know that each product on the market founds a buyer and meet one's needs. Suggesting a single review as the most helpful review to all kind of customers with different perceptions, purchasing powers and expectations may mislead the customers' transactions and may cause dissatisfaction [1]. To overcome this problem, we studied on a person-basis helpful reviewer detection. Similar Opinion Holders Algorithm (SOHA) counts per-customer based helpful reviewers and promotes their reviews on a certain product. The idea behind SOHA is detecting the similarities between an individual customer and all the other customers and rating them according to the selected similarity indicator. SOHA is a higher order algorithmic framework. Similarity detection methods can be applied on SOHA as an add-on.

The study is structured very straightforward. In the second section we introduced the application framework consists of the dataset used, the SOHA algorithm itself and the parallelization for streaming big data problems. Results are given and discussed under the third section. Last section concludes the idea and the study mentioning the future directions.

2. Application Framework

2.1. Application Dataset

E-commerce datasets are the best data sources for sentiment analysis applications. In this study we applied our methods on Amazon product-review dataset [2]. The dataset contains product reviews and metadata information from Amazon website, including 143.7 million reviews spanning the years from 1996 to 2014. The dataset is categorized under sub-classes due to the product category which the review is written for. We have selected the category "Electronics" for our study.

{
"reviewerID": "A2SUAM1J3GNN3B",
"asin": "0000013714",
"reviewerName": "J. McDonald",
"helpful": [2, 3],
"reviewText": "I bought this for my husband who plays the piano. He is having a wonderful time playing these old hymns. The music is at times hard to read because we think the book was published for singing from more than playing from. Great purchase though!",
"overall": 5.0,
"summary": "Heavenly Highway Hymns",
"unixReviewTime": 1252800000,
"reviewTime": "09 13, 2009"
}

Reviews and metadata files in JSON format are handled with Pandas Python library [3]. Each review is deserialized in a data frame and transformed in a quadruple form of 4 dimensions: Target entity, opinion holder, sentiment and time. In this study we prefer to express sentiments based on reviewer rates. No any Natural Language Processing (NLP) task is performed. This means that aspect level sentiments are not considered. However, entity level sentiments are expressed in 3 classes: Positive, neutral and negative.

1.1. Similar Opinion Holders Algorithm (SOHA)

Problem Definition: Given a set of reviewers *R* = $\{r_1, \ldots, r_n\}$ and a set of products *P* = $\{p_1, \ldots, p_m\}$ and each p_i has a set of reviews R_i with each r_i reviewed a set of products P_i, we need to identify *k* closest reviewers in *R* to a certain reviewer r_c for a product p_d.

For a definite product-reviewer pair $[p_d, r_c]$ firstly we define all the reviewers of p_d which is said R_d. The rating similarities between r_c and the each reviewer in R_d give us the set of k closest opinion holders for r_c. The similarity calculation schema is an add-on for SOHA. In this study we hold the similarity calculation metric as easy as possible without any NLP tasks.

```
1.  for each reviewer r'_i ∈ R_d
2.      select the set of products which r'_i has already reviewed  P' ⊆ P
3.      for each p'_j ∈ P'
4.          if  r_c  has already reviewed  p'_j
5.              if  rate(r_c,p'_j) == rate(r'_i,p'_j)
6.                  sim_score(r'_i) += 1
7.              else
8.                  sim_score(r'_i) -= 1
9. order the element of  R_d due to the similarity scores
10. select top k reviewer from  R_d
```

Listing 1. SOHA Algorithm on a single node

This algorithm above works in a sequential order in a linear time to define closest opinion holders for a single, unique opinion holder. This does not aim to extract inter-relationships between all the opinion holders. In a real timed e-commerce application if a customer face with a product page including reviews, the system must be able to decide the closest reviewers and suggest their opinions as top reviews. Due to the huge amount of historical data and below-a-second response time, we preferred to study one of the most prospering parallelization paradigms, MapReduce on Hadoop clusters.

1.2. Parallelization by MapReduce Paradigm

The Map-Reduce programming model is accepted as a de-facto computation framework for big data analysis. It was first introduced by Google in 2004 and developed by Yahoo. Several implementations of Map-Reduce paradigm exist including Microsoft Dryad, Google Sawzall and Apache Hadoop. Hadoop, an open source development project supported by Apache Foundation is the most popular and documented framework to realize Map-Reduce based applications.

Map-Reduce is a distributed computational framework enabling data intensive analysis. It is inspired by the functional programming paradigm [4]. Based on divide-and-conquer method, it recursively breaks down a complex computational problem into sub-problems which allow to be solved directly by a standard computing environment. These sub-problems are assigned to many

worker nodes called Mappers and processed by these mapper nodes in parallel. The intermediate results of the sub-problems are combined based on some key and assembled by Reducer nodes. The overall computation result is delivered by reducer nodes. The processing unit used in the communication between mapper nodes and the reducer nodes is a key-value pair. The mapper outputs are denoted by a key and reducer nodes regroup values from various mappers based on these keys. The process of passing key-value pairs from mappers to the reducer is known as shuffling. Each reducer is assigned a subset of the intermediate key space, called a partition.

The mapper and reducer nodes are called worker nodes and master nodes in Hadoop computation environment [5]. The master node takes the input, divides it into smaller sub-problems, and distributes them to worker nodes in Map step.

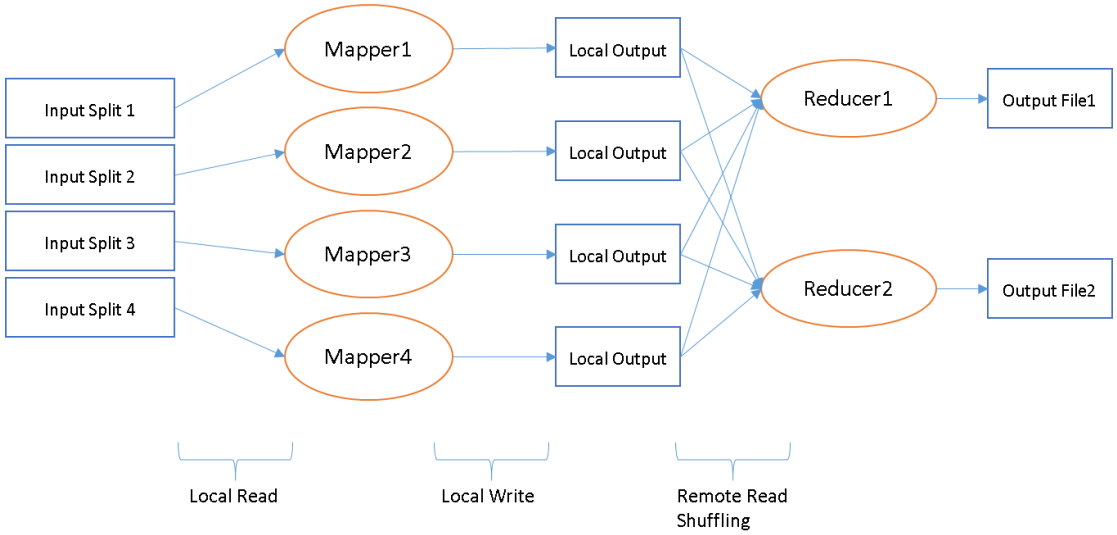

Figure 1. Map-Reduce Sample Infrastructure

Figure 1 shows a sample data flow of Map-Reduce framework. In this sample setup, input dataset is split into 4 buckets. Each bucket is processed by its assignee mapper node. The computation part of the algorithm is handled in mapper modes. A mapper node delivers its local output as a key-value pair. All the keys delivered from all mappers are allocated to different reducer nodes. Reducer nodes are responsible of reading these local outputs from the network and combining them based on their keys. The aggregation part of the algorithm is handled in reducer nodes. The aggregation results based on key combination are delivered to the output files.

This study by nature is highly convenient to adapt in a MapReduce framework. As seen in listing 2, SOHA algorithm transacts on various reviewers for a single product. For each reviewer it calculates a similarity score depending their historical review records. A small adaptation in SOHA algorithm can easily parallelize the execution framework.

Let *o* is the number of mapper nodes in our MapReduce setup. The set of products P can be split to various mapper nodes as $P_1, P_2, \ldots, P_o$. The SOHA algorithm can be ran on each mapper node for its' dedicated subset of products P_m.

Mapper Execution:

for each reviewer $r'_i \in R_d$

select the set of products which r'_i *has already reviewed* $P_m' \subseteq P_m$

for each $p'_j \in P_m'$

if r_c *has already reviewed* p'_j

if $rate(r_c, p'_j) == rate(r'_i, p'_j)$
$sim\_score(r'_i) += 1$
else
$sim\_score(r'_i) -= 1$

Reducer Execution:
Sum up $sim\_score(r'_i)$ *of each node to a global similarity rate*
order the element of R_d *due to the its global similarity rate*
select top k reviewer from R_d

Listing 2. SOHA Algorithm on Map reduce

2. Results and Discussion

In the previous sections we explained the structure of the dataset, clarified the Similar Opinion Holders Algorithm (SOHA) and enriched the algorithm by using MapReduce parallelization. This section aims to give the results of singular and parallel execution of SOHA algorithms on Amazon datasets via 1, 2, 4 or 8 nodes. The results are shown on Table 1.

Definition: Scale factor can be calculated by the terms on Listing 2. It is the combination of all elements in R_d and $P_m{}'$. Thus we may say that it is the product of the cardinality of R_d and the cardinality of $P_m{}'$.

Table 1. Execution times on various MapReduce setups

Scale Factor	1 node	2 nodes	4 nodes	8 nodes
100	0.22	0.13	0.08	0.05
1000	1.80	1.00	0.63	0.34
10000	16.50	7.90	5	2.4
100000	130	68	35	15

Figure 2 demonstrates the decrease in execution time of SOHA algorithm in case of the usage of Map Reduce parallelization. One may notice that the parallelization has a valuable effect on especially on bigger scale factors.

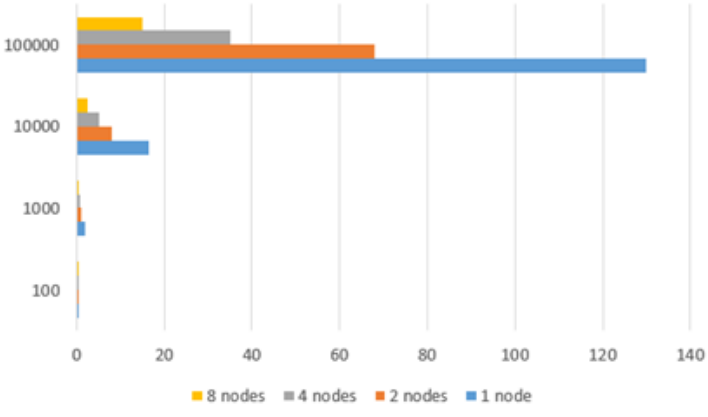

Figure 2. Noticeable decrease in execution time

Figure 3 apparently reveals the positive effect of parallelization on the execution time of SOHA algorithm. In comparison with single node execution, the more parallel environment yields the better execution times.

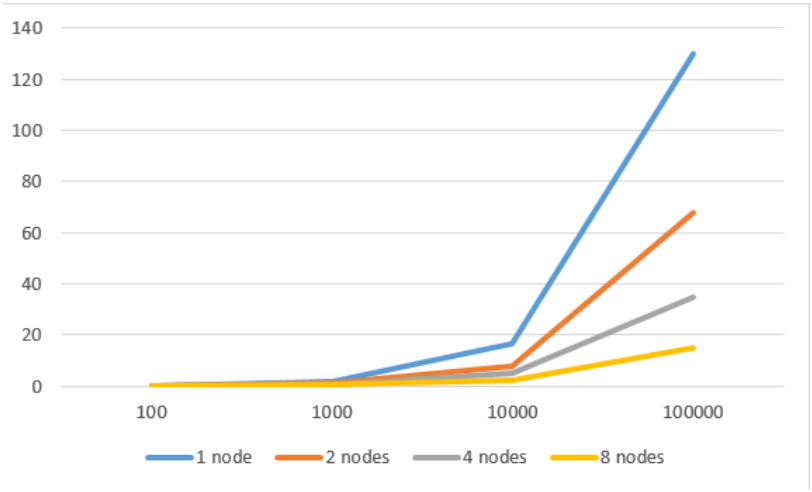

Figure 3. The effect of parallelization on SOHA

According to the results that we have obtained from various MapReduce setups we may obviously note that our SOHA algorithm is giving response in meaningful time whether the dataset has a big scale factor. The execution time of SOHA depends basically on scale factor. The more reviewer SOHA handles requires much more execution time. The same constraint is either on the number of products that each reviewer has reviewed. Scale factor depends on these two values. For the accuracy on calculation of similar opinion holders we need huge amount of reviewers, each has a large number of reviewed products. This means a great value of scale factor, hence increase in execution time. The operation details of the parallelization show us that mapping the problem into multiple mappers can separate the execution time in linear manner. Shuffling between mapper nodes does not cost much in our problem. By the advantage of linearly separable structure of SOHA, we can add as much as possible nodes to reduce the execution time in a usable interval.

3. Conclusion

In this study, we proposed an algorithm to detect similar opinion holders of a customer on an e-commerce dataset. The algorithm is designed adaptive to MapReduce parallelization as much as possible. We tried the algorithm on 1-node, 2-nodes, 4-nodes and 8-nodes Hadoop clusters. Based on obfuscated real world data we reached very meaningful results. The proposed algorithm, SOHA takes review rates in consideration. This may mislead the algorithm and also disallow the calculation of the similarity according to a sub aspect of a product. In the future, we plan to analyze the comment itself by using Natural Language Processing techniques. Also, we plan to exercise the reviews in aspect level. Entity level review consideration may sometimes misrepresent the opinion holder's intent.

References

[1] M. J. Shaw, C. Subramaniam, G. W. Tan, & M. E. Welge, (2001). "Knowledge management and data mining for marketing," *Decis. Support Syst.*, *31*(1), 127–137,

[2] J. McAuley, "Amazon product data," 2015. [Online]. Available: http://jmcauley.ucsd.edu/data/amazon/.

[3] Pandas, “Pandas Data Analysis Library,” 2015. [Online]. Available: http://pandas.pydata.org/.
[4] C. L. Philip Chen and C.-Y. Zhang, “Data-intensive applications, challenges, techniques and technologies: A survey on Big Data,” *Inf. Sci. (Ny).*, vol. 275, pp. 314–347, Aug. 2014.
[5] U. Gupta and L. Fegaras, “Map-based graph analysis on MapReduce,” *2013 IEEE Int. Conf. Big Data*, pp. 24–30, Oct. 2013.