



UPPSALA UNIVERSITET

Large Datasets for Scientific Applications

Project Report

Team 26

Oskar Stenerlöv

Ismini Stamatelou

Hamza Imran Saeed

June 13, 2019

Background

The data set used in this project is a collection of Reddit comments. Reddit is a social media forum on which countless of topics are being discussed every day. It is being ranked as the 6th most visited website in the United states, and 21th in the entire world as of june 2019 [1]. In order to structure the discussion of these topics, Reddit have created so called subreddits in which only content related to the subreddit is allowed. Discussions are started through posts that contain whatever the user wants to share on the subreddit [2]. There are many posts being made at the same time and through scores from other users, popular post will be showed before less popular ones, making the content of the post "important". Now, all users are also allowed to comment on the posts. Comments are the bulk of Reddit's data as a single post can have over thousands comments on it. The comments on their own does not say that much, but coupled with properties other than the actual content of the comment, such as time of submission and subreddit in which it was posted, one can extract valuable information. The data set was created in the purpose of research and it is publicly available to anyone; what one wants to do with the comments is limited only by imagination.

The group decided to focus on the content of the comments and under which subreddits they were posted. In essence, comments belonging to the most commented subreddits were filtered out and their content dissected. The goal was to see what was trending on the popular subreddits by word counting and displaying the most frequently used words. With this approach trends can be easily discovered, and if one wants to know what is trending on a specific subreddit, say politics, one can modify the algorithm to one's needs. In a study from 2015, a group of researchers did something similar. Their goal was to find out what type of flavors on e-cigarettes were popular and they utilized reddit data to do so. Similarly to the approach described above they managed to determine trending flavors by analyzing content on subreddits related to e-cigarettes [3]. It is however not clear from the article what type of software or platforms the authors used. Other similar websites have also been analyzed. Digg is such an example, where researchers wanted to determine certain social behavioural patters among the users with large data analytics, using the comments as their data [4]. At first glance the possibilities seem very limited but it quickly becomes clear that even the smallest bit of information can say quite a lot about where the world is heading, what to look into and what to look out for.

Data format

Reddit comments come as JSON objects, which is a lightweight, semi-structured format for storing and transporting data. It is minimal and readable and easy for machines to parse and generate. JSON objects have one or multiple key/value pairs of the following JSON data types: string, number, object, array, boolean or null. When it comes to the structure of the dataset, JSON is semi-structured , as opposed to relational database rows and other conventional formats. In our case, a comment is an object that contains many key/value pairs such as the authors name, the body which is basically the comment itself, the subreddit where the comment was posted, the date and other information about the comment. In our analysis we used the "body" and "subreddit" fields. Being familiar with JSON as a format from previous experiences, more time could be spent on the application pipeline and setup, alleviating the need of having to worry about understanding a new format. Other similar formats, such as XML is in theory an option, but when it comes to both read and write speed, JSON is faster in both regards. As the ability to handle lots of data in as short amount of time as possible, any improvements to speed counts, making JSON a superior format between the two.

```
{ "author": "Dethcola", "author_flair_css_class": "", "author_flair_text": "Clairemont", "body": "A quarry", "can_gild": true, "controversiality": 0, "created_utc": 1506816000, "distinguished": null, "edited": false, "gilded": 0, "id": "dnqik14", "is_submitter": false, "link_id": "t3_73ieyz", "parent_id": "t3_73ieyz", "permalink": "/r/sandiego/comments/73ieyz/best_place_for_granite_counter_tops/dnqik14/", "retrieved_on": 1509189606, "score": 3, "stickied": false, "subreddit": "sandiego", "subreddit_id": "t5_2qq2q" }
```

Figure 1: Example of a reddit comment

Computational experiments

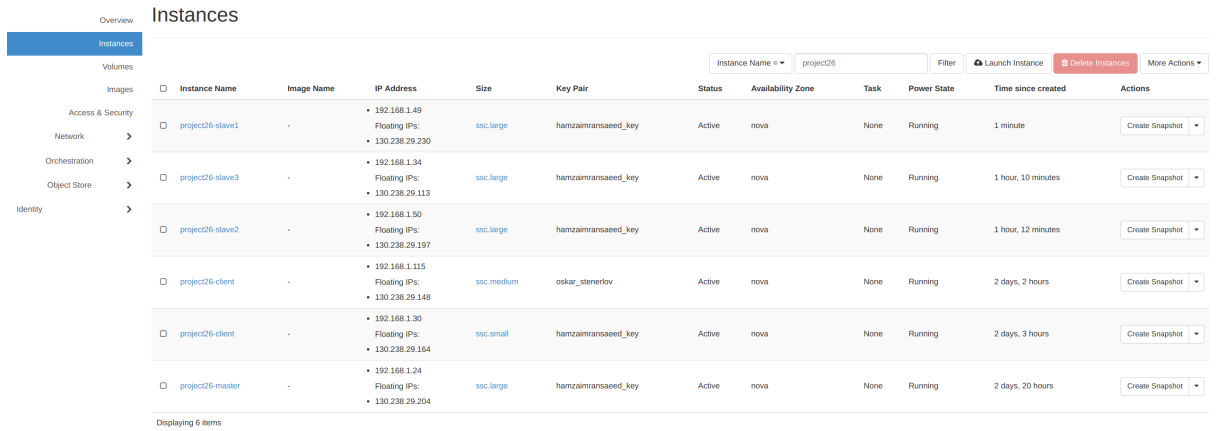
This section provides the motivation for our choice of tools, describes the distributed system that we designed and the scalability study that we conducted.

Tools & Implementations

We use Spark for our experiments because Spark provides a fault-tolerant abstraction for in-memory computations through structures such as RDDs (Resilient Distributed Datasets) and Dataframes. These data structures provide the ability to process the data in-parallel on multiple nodes, which made Spark a good choice to use for the kind of experiments we wanted to do since we needed to perform many computations on the data to achieve the desired results. The aim was to do this without having to load data from disk every-time as that would have been expensive and Spark offers the possibility to improve performance significantly without reloading the data from disk each time due to its ability to perform in-mememory computations [5].

Hadoop was also an option as it is often used to run queries on large datasets, through interfaces such as Pig [6] and Hive [7]. The limitation with hadoop is that we could not load the reddit dataset into memory across a number of machines and query it repeatedly as we could do with Spark. Instead, with Hadoop, each query would have incurred significant latency because it runs as a separate MapReduce job and reads data from disk [8].

For the purpose of this project, we deployed our own spark cluster as well as a distributed Hadoop cluster for HDFS to store the data. We created VMs at the Openstack environment on the SNIC Science Cloud as we did for our assignments. These VMs can be seen in the Figure 2 below.



Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
project26-slave1	-	• 192.168.1.49 Floating IPs: • 130.238.29.230	ssc.large	hamzaimransaeed_key	Active	nova	None	Running	1 minute	Create Snapshot
project26-slave3	-	• 192.168.1.34 Floating IPs: • 130.238.29.113	ssc.large	hamzaimransaeed_key	Active	nova	None	Running	1 hour, 10 minutes	Create Snapshot
project26-slave2	-	• 192.168.1.50 Floating IPs: • 130.238.29.197	ssc.large	hamzaimransaeed_key	Active	nova	None	Running	1 hour, 12 minutes	Create Snapshot
project26-client	-	• 192.168.1.115 Floating IPs: • 130.238.29.148	ssc.medium	oskar_stenerlov	Active	nova	None	Running	2 days, 2 hours	Create Snapshot
project26-client	-	• 192.168.1.30 Floating IPs: • 130.238.29.164	ssc.small	hamzaimransaeed_key	Active	nova	None	Running	2 days, 3 hours	Create Snapshot
project26-master	-	• 192.168.1.24 Floating IPs: • 130.238.29.204	ssc.large	hamzaimransaeed_key	Active	nova	None	Running	2 days, 20 hours	Create Snapshot

Figure 2: Openstack cluster nodes and master

The cluster we setup had a master node and 4 worker nodes. The HDFS can be accessed on 130.23.29.204:50070 and has a capacity of around 115 GB. A screenshot of the Hadoop web UI is shown in Figure 3. As can be seen in this figure, the Block Pool ID has 192.168.1.24 in it which corresponds to the IP of the **project26-master** machine in Figure 2. To design our scalability experiment, we wanted to have machines that had enough memory and VCPUs to be able to scale to larger datasets and be able to give us results in a feasible amount of time.

Thus, the following configurations were used for the 4 slaves and the master in our cluster:

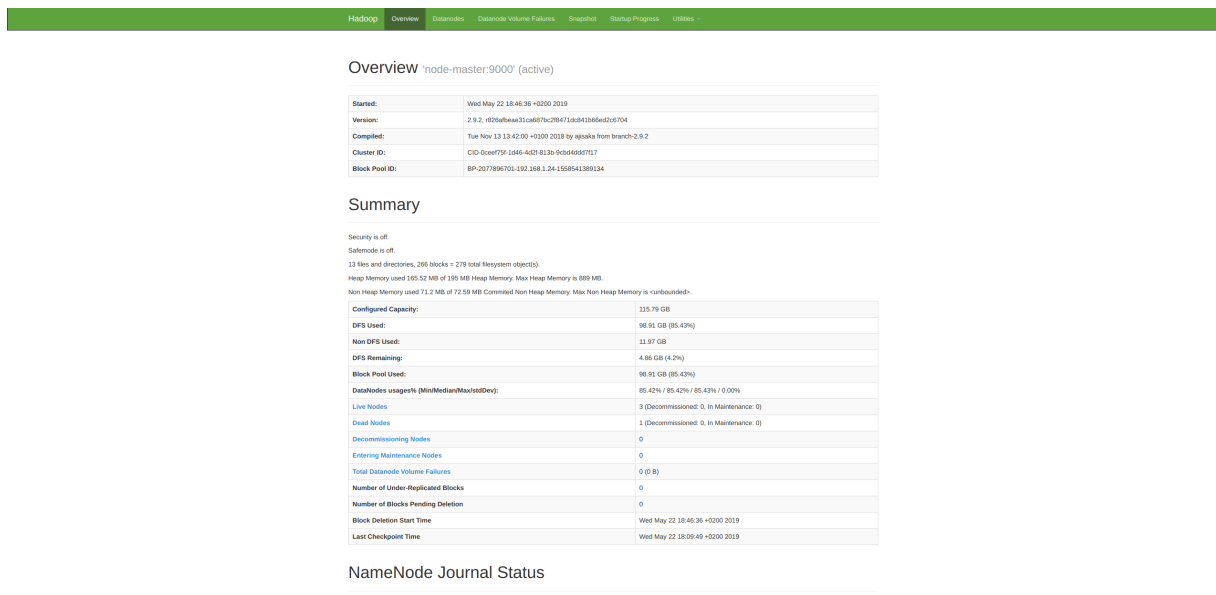


Figure 3: Hadoop and HDFS

- **Source:** We used the Ubuntu 18.04 LTS image as the source image for the machines.
- **Flavor:** *ssc.large* flavor was used which had 4 VCPUs, 8Gb Ram, 40 Gb total disk and root disk
- The Spark cluster with a total of 4 worker nodes had 16 cores(4 cores each) and a memory of 27.2 Gb in total (6.8 Gb on each worker). The Spark UI which shows the cluster nodes can be seen in Figure 4.

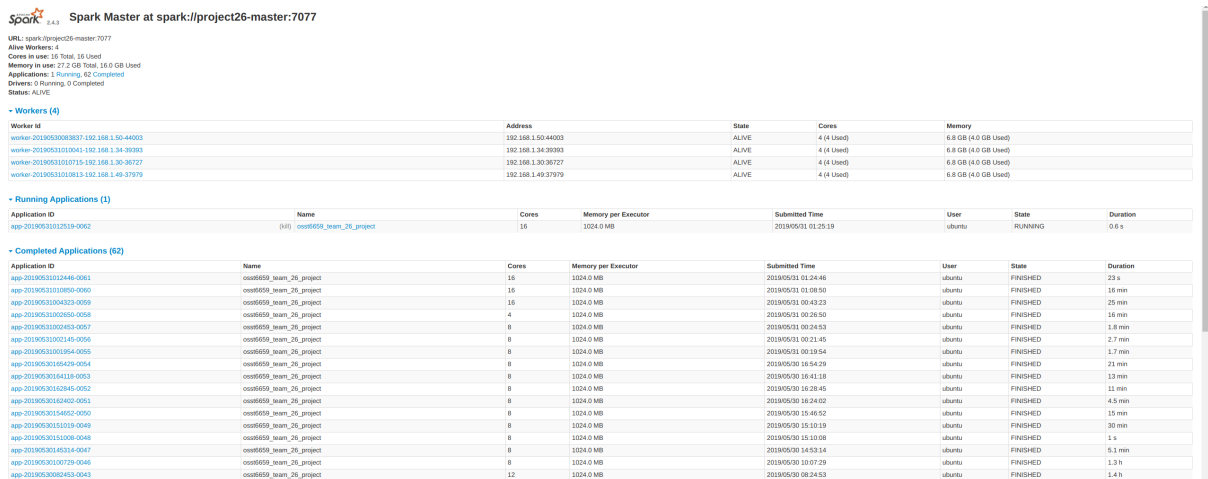


Figure 4: Spark cluster with slave nodes

For the computational experiment, we loaded the the data into data frames and RDDs from json files which were present on the HDFS. The aim of our computational experiment was to generate wordclouds of the most frequently discussed/used topics/words in the top subreddits. For this experiment the number of subreddits chosen was 5. We processed the data by sorting and filtering it and then eventually using spark functions such as map and reduce to get our results. An example of the wordcloud from the subreddit AskReddit is shown below.

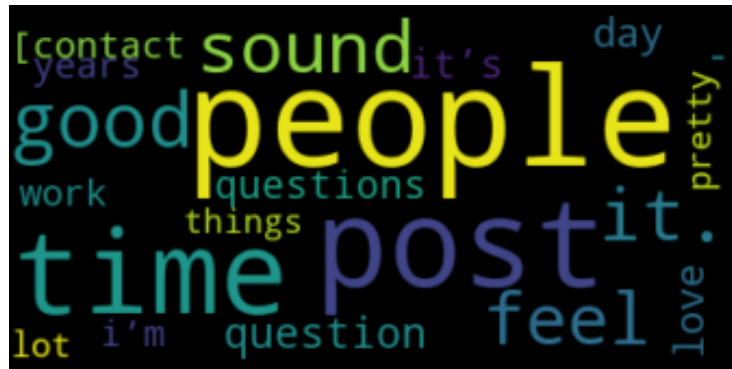


Figure 5: Wordcloud for Askreddit

Results

Below are the wordclouds and the histograms generated for the 5 most commented subreddits. Each histogram shows the ten most frequent words. Although the difference in size between the subreddits was quite big, we still got the same 5 subreddits as the most commented ones and the most frequent words remained almost the same.



Figure 6: Wordcloud for NBA

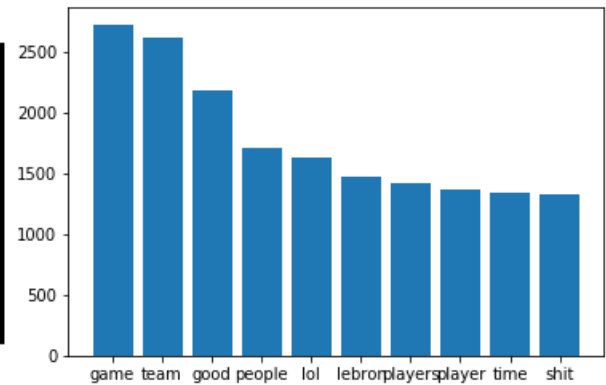


Figure 7: Histogram for NBA

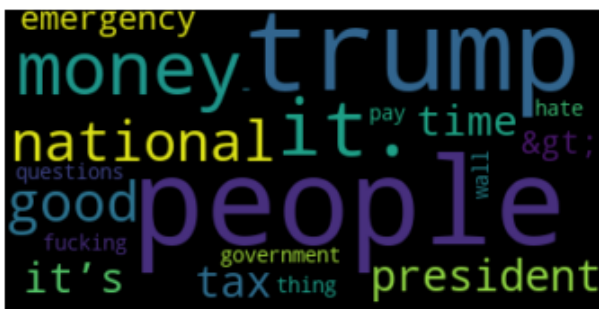


Figure 8: Wordcloud for Politics

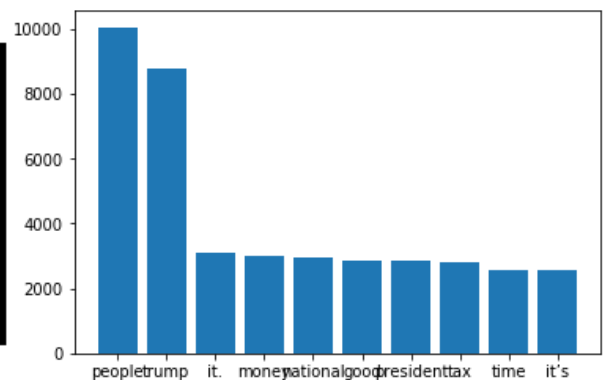


Figure 9: Histogram for Politics



Figure 10: Wordcloud for Dankmemes

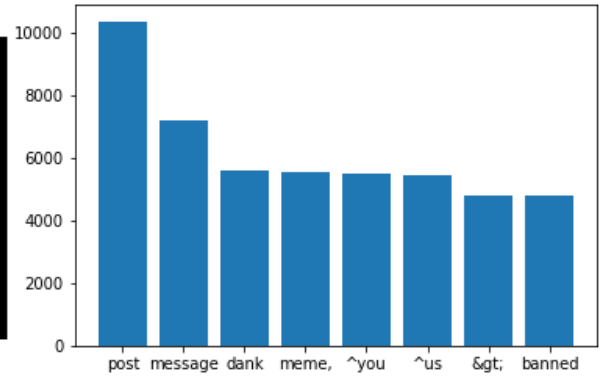


Figure 11: Histogram for Dankmemes



Figure 12: Wordcloud for ApexLegends

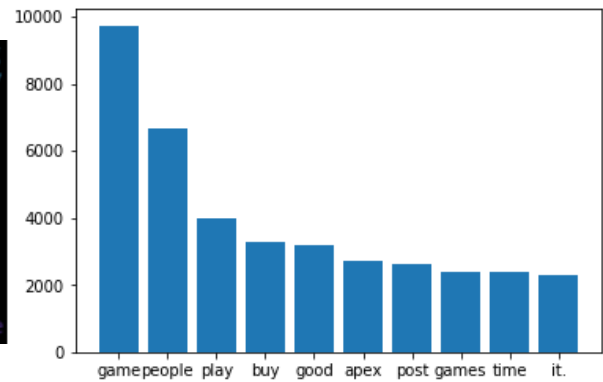


Figure 13: Histogram for ApexLegends

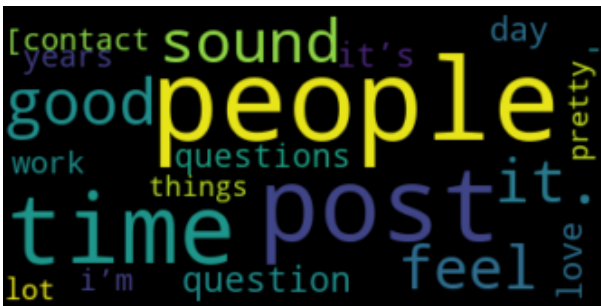


Figure 14: Wordcloud for Askreddit

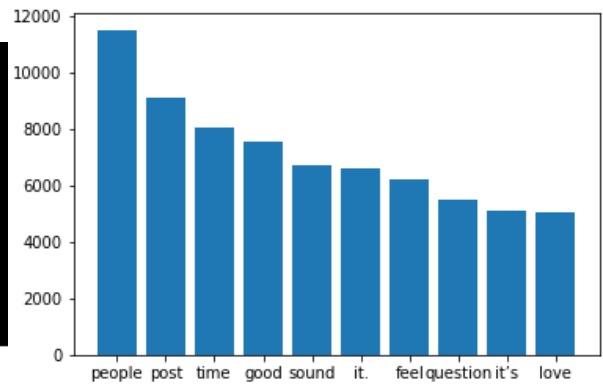


Figure 15: Histogram for Askreddit

Scalability Experiment

In order to test and confirm the scalability of the cluster, scalability studies were conducted. The experiment tested the horizontal scalability of our solution (e.g., by varying the number of workers). Horizontal scalability is defined as scaling a system by adding more resources. For example, by adding additional machines/workers to a cluster that can be used for storage or computation. On the other hand, vertical scalability is defined as scaling up by improving the capacity of the available resources. For example, the number of CPU cores or the size of the RAM is increased for the already existing resources [9]. In our case, the focus was on testing the horizontal scalability of the cluster setup.

The Reddit data was already available in ≈ 4 GB chunks for each day and thus we decided to have 4 different data set sizes for our experiments. These different sizes chosen were ≈ 4 GB, ≈ 8 GB, ≈ 16 GB and ≈ 32 GB as we wanted to test what the effect is when we double the data size. As our spark cluster had 4 worker nodes, we chose 3 different cluster configurations i.e 1 worker node, 2 worker nodes and 4 worker nodes. Similar to our data size doubling for the experiment, we also wanted to double the number of slaves for every run on the same data size to test the effect that has on the processing time. To measure the horizontal scalability, the run time for a specific job on the dataset was recorded for all three configurations, i.e 1, 2 and 4 worker nodes in the cluster. These times are shown in Table 1. Worker nodes were changed by running only the number of slave nodes required in the specific run. As seen in the figure 16 below, we can observe that the run time increased linearly with data size for the different cluster configurations and the run time was reduced equally for increasing number of nodes.

Number of Nodes	Time Taken	Data Size
1	5.17 min	4.7 Gb
2	2.67 min	4.7 Gb
4	1.98 min	4.7 Gb

Number of Nodes	Time Taken	Data Size
1	9.20 min	9.3 Gb
2	5.09 min	9.3 Gb
4	3.67 min	9.3 Gb

Number of Nodes	Time Taken	Data Size
1	16.23 min	17.5 Gb
2	9.55 min	17.5 Gb
4	6.43 min	17.5 Gb

Number of Nodes	Time Taken	Data Size
1	31.45 min	31.8 Gb
2	16.04 min	31.8 Gb
4	11.15 min	31.8 Gb

Table 1: Cumulative time taken by reduce and sort functionality using different cluster configuration over different data size.

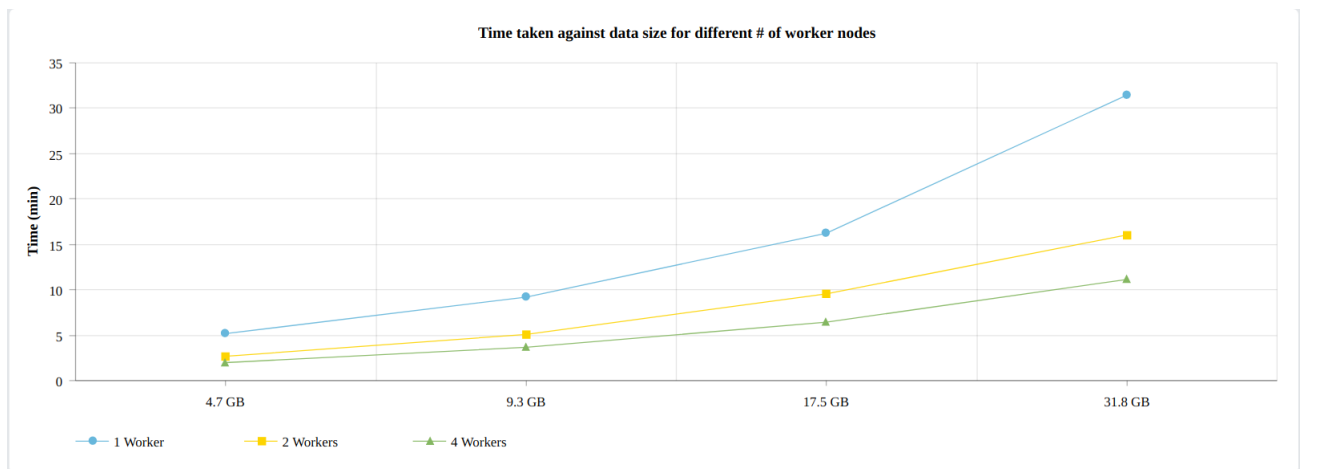


Figure 16: Graph showing time taken against data size for different worker nodes

Discussion and conclusion

As a result of the time measurements of our experiment, we reach the conclusion that the performance of the proposed solution is improved as the number of worker nodes is increased and the spark cluster configuration used is horizontally scalable. On the other hand, this does not imply that the architecture of the implementation is irrelevant to the size of the input dataset. For example, a spark cluster with 50 worker nodes that have 4 VCPUs, for the processing of 50MB of data, should be considered as inefficient.

From Table 1 it is clear that the amount of RAM, the number of CPU cores available and the number of workers in the cluster have an impact on the execution time of the experiment. When the data exceeds the amount of RAM available and as Spark does in-memory computations, we assume that the larger execution times can be attributed to the disk swaps between the HDFS and the spark cluster memory. On the other hand, when the data size is smaller than the available RAM, the computation time is a lot faster and this can be attributed to the fact that the whole dataset can be kept in memory by the cluster. The Graph in Figure 16 shows almost a linear increase in computation times when the data is doubled. There is a larger noticeable increase in the time for 1 worker node when the datasize is ≈ 32 GB. This can be attributed to the large number of reads from the HDFS because the data would not fit the memory of 1 worker node in the cluster.

When the data size is the same, when we double the number of worker nodes, the run times decrease. When we double the number of workers, in all the cases the run time decrease by a factor of ≈ 1.85 . Furthermore, when we double the nodes further, i.e 4 nodes, the run time decrease by a factor of ≈ 1.4 . This shows that although doubling the nodes does decrease the run time, the proportion of the change gets smaller as the number of nodes increases. There can still be a significant time difference especially when the size of the dataset is really large.

Reddit comments can provide us with a lot of useful information, and what we've done here is a very simple example that can be expanded much more. For example, one can analyze time intervals of comments on certain subreddits, and with time zones infer which nationality of the users on specific subreddits. The analysis can be expanded further, by reviewing how the content of subreddits change during noteworthy events such as Superbowl, US elections and Valborg, just to mention a few. Many datasets can also be used for more complicated tasks like, for example, finding users' common interests depending on which subreddits are commenting on and how likely is for a user to comment to a new subreddit that a similar user has already commented on. When it comes to our project, bigger datasets can be used for more accurate results as well as more slave nodes for more efficient and faster processes. The prepositions dictionary can also be expanded to remove more words that are frequently used but do not provide us with any useful information. All in all, we feel like we've only scratched the surface on this and are confident that anyone with the motivation and time to put some effort into a similar project would be able to achieve countless valuable analyses.

References

- [1] “Reddit Competitive Analysis, Marketing Mix and Traffic - Alexa.” [Online]. Available: <https://www.alexa.com/siteinfo/reddit.com>
- [2] T. Steinbaur, “Information and social analysis of reddit,” *Retrieved from TROYSTEINBAUER@ CS. UCSB. EDU*, 2012.
- [3] L. Wang, Y. Zhan, Q. Li, D. Zeng, S. Leischow, and J. Okamoto, “An examination of electronic cigarette content on social media: analysis of e-cigarette flavor content on reddit,” *International journal of environmental research and public health*, vol. 12, no. 11, pp. 14916–14935, 2015.
- [4] S. Jamali and H. Rangwala, “Digging digg: Comment mining, popularity prediction, and social network analysis,” in *2009 International Conference on Web Information Systems and Mining*. IEEE, 2009, pp. 32–38.
- [5] M. A. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster computing with working sets,” *Annals of emergency medicine*, vol. 39 6, pp. 691–2, 2002.
- [6] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins, “Pig latin: a not-so-foreign language for data processing,” in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 2008, pp. 1099–1110.
- [7] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, “Hive: a warehousing solution over a map-reduce framework,” *Proceedings of the VLDB Endowment*, vol. 2, no. 2, pp. 1626–1629, 2009.
- [8] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, “Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing,” in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2012, pp. 2–2.
- [9] R. Cattell, “Scalable sql and nosql data stores,” *SIGMOD Record*, vol. 39, pp. 12–27, 12 2010.