RE2: Expert Finding

Max de Boer
2793346
2793346@student.vu.nl

Zhe Liu
2756066
z.liu4@student.vu.nl

Giancarlo Ianni
2676807
g.ianniermudes@student.vu.nl

ABSTRACT

This report describes the LSDE second assignment in which the group created a Big Data pipeline and analyzed a 11.3TB dataset from Reddit. The goals were to find the most popular subreddits and to find the expert users for each subreddit. We also looked for a way to extract the topics from the different subreddits and the expert users. In the end the results were visualized in a simple Flask website using wordcloud and bar charts.

1. INTRODUCTION

For this research we used the data from the social media platform Reddit. Reddit is home to thousands of communities with varying topics. A community is also called a *subreddit*. Users can make posts in subreddits and they can comment on other posts. They can also make comments on each others comments, which will make comment chains. This makes Reddit an excellent platform for discussions and sharing knowledge, since users can also chose to *upvote* or *downvote* posts and comments on Reddit. You can give an upvote when you agree with a post or comment and you can give it a downvote if you don't agree with it. An upvote will increase the score of a post or comment, while a downvote will decrease the score[3]. Popular opinions have a higher probability of getting a positive score while controversial opinions will sometimes get negative scores as well. With this manner of grading comments, it is assured that an expert is not a user with a big amount of submissions and/or comments, and instead, for the quality of the posted content according to the community.

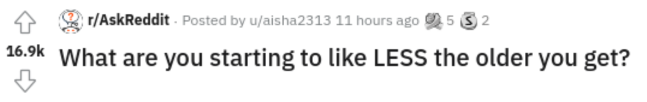

r/AskReddit · Posted by u/aisha2313 11 hours ago 5 2
16.9k
What are you starting to like LESS the older you get?

Figure 1: A post on the subreddit 'AskReddit'

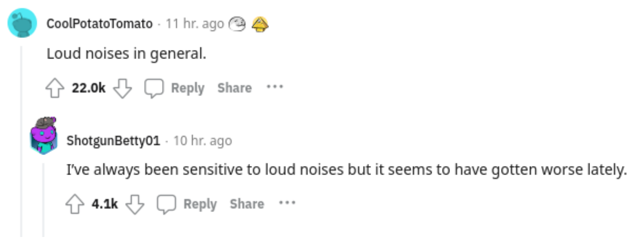

CoolPotatoTomato · 11 hr. ago
Loud noises in general.
22.0k Reply Share
ShotgunBetty01 · 10 hr. ago
I've always been sensitive to loud noises but it seems to have gotten worse lately.
4.1k Reply Share

Figure 2: Two of the comments for the post in Figure 1

1.1 The dataset

There are different types of posts on Reddit. A user can post stories, links images and videos [3]. The provided dataset only contains the texts of each post and doesn't contain the images or videos. Reddit was founded in 2005 and the first data is from December 2005. Reddit didn't have a big amount of users at the time so the amount of data from the first years is small. When Reddit started to get more popular, the size of the data increased. This can be seen in Figure 3 and 4.

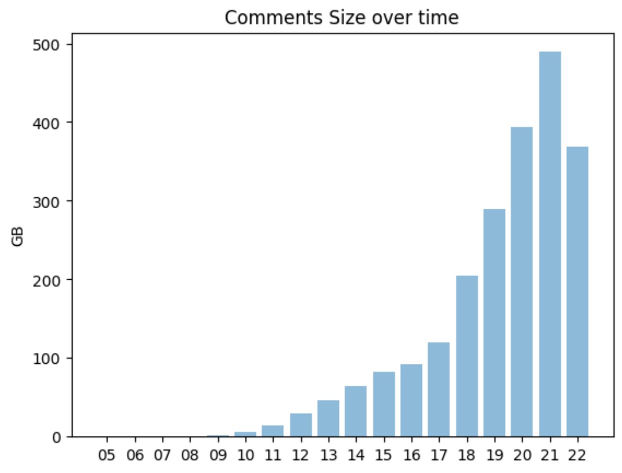


Figure 3: Amount of data for the comments per year in GB

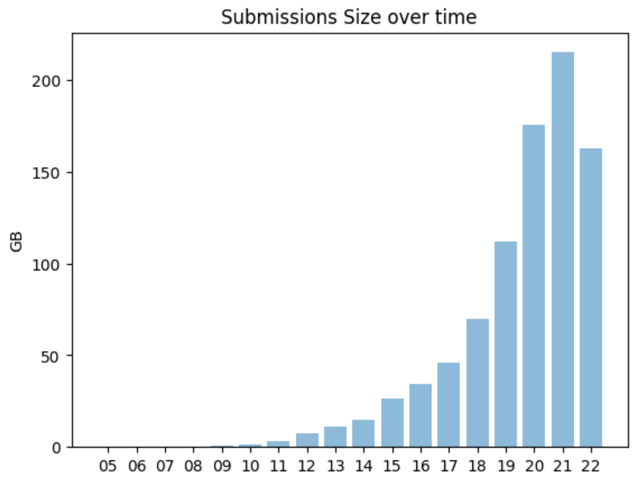


Figure 4: Amount of data for the submissions per year in GB

The amount of data grows each year almost exponentially. For the comments, there was more data in 2021 than in the period from 2005 to 2016. For this reason it was decided to only use the data from 2019-08 to 2022-08. This would result in the last 3 years of the data from Reddit. These years still contain most of the data and would give a more actual view of the current most popular subreddits and their experts.

2. RELATED WORK

The related work includes two parts, data cleaning and algorithms. Each part is discussed separately below.

2.1 Data Cleaning

Data cleaning is a significant step before applying algorithms to text. It makes a big difference to the final outcomes. The elements contained in Reddit comments have a lot of variance. Besides normal text, there are a large number of emoticons, URLs, tags, escaped characters, etc. in the text, which can affect the effectiveness of the algorithm. IT was decided finally to remove six elements for each sentence, namely emoticons, URLs, html tags, reddit tags, @ and some escape characters. Furthermore, data cleaning does not only mean removing elements. It became visible that sentences of some comments did not end up with a period, question mark, or exclamation mark. For those sentences, it was necessary to add a period at the end of it to make sure the algorithm can split sentences correctly.

2.2 Algorithms

There were two algorithms involved in the project.

- **RAKE:** It is used to extract important keywords or phrases from the text.
- **TextRank:** It is used to select meaningful sentences from the huge amount of comments from users.

The basic principles of these two algorithms and the reasons for using them is discussed next.

2.2.1 *Introduction to Algorithms*

RAKE: RAKE is short for *Rapid Automatic Keyword Extraction*, proposed by Rose et al. [5] in 2010. Apart from breaking the limit that keyword extraction is only based on corpus-oriented methods, RAKE also attaches importance to the co-occurrence of words. The main process of RAKE can divided into three parts. First, it generates the candidate keywords. RAKE generates the candidate keywords by splitting the text by stop words. Candidate keywords can be a word or a phrase. In the second part it calculates the score for each candidate keyword. The score can be calculated with the following formula:

$$score(w) = (deg(w)/freq(w))$$

$score(w)$ stands for the word score, $deg(w)$ stands for the word degree and $freq(w)$ is the word frequency. The word degree can be interpreted as the sum length of all candidate keywords which contain the word. Then for each candidate key phrase, the scores of each of these words are added up and ranked. At last, RAKE considers the top third of the total number of candidate phrases as the extracted keywords.

TextRank: The TextRank algorithm is a graph-based ranking algorithm for text. The basic idea of TextRank is derived from Google's PageRank algorithm, by partitioning text into several constituent units, constructing a node-linked graph, using the similarity or co-occurrence as the weights of edges, calculating the TextRank values of units through circular iterations, and finally selecting the units with high ranking. Mihalcea and Tarau [4] elaborates on the principles and applications of TextRank.

In TextRank-based automatic summarization, there are five main steps.

1. Splitting text: dividing the given text into several sentences and collecting them in a set $T = [S_1, S_2, \ldots, S_n]$, then constructing the graph $G = (V, E)$, V is the subset of T.
2. Pre-processing for sentences: splitting sentences from V, removing the stop words from them, then generating $S_i = [w_{i,1}, w_{i,2}, \ldots, w_{i,m}]$, where $w_{i,j} \in S_i$.
3. Calculating sentence similarity: according to the ratio of overlap between two sentences, the similarity of given sentences S_i and S_j is defined as
$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{\log(|S_i|) + \log(|S_j|)}$$
4. Calculating sentence weights: Iteratively propagate the weights to calculate the score of each sentence according to the formula.
5. Extracting sentences: the scores of the sentences obtained from the last step are sorted in descending order and the top N sentences are selected to form the summary.

2.2.2 *Reasons for Choosing Algorithms*

Our reasons for choosing these two algorithms contain two aspects, one is implementation and the other is performance.

In terms of implementation, RAKE and TextRank are quite mature and well-tested unsupervised learning methods for the extraction of keywords from text. They don't need us to prepare the dataset and training model in advance.

There are well-developed libraries that we can call directly. Moreover, according to the research from Thushara, Mownika, and Mangamuru [6], which compared different methods for extracting keywords from documents, RAKE is efficient and fast while still being quite precise. Since we have limited time to work on the project, the simple and easy deployment of RAKE and TextRank is an advantage over other algorithms.

In addition, both algorithms have good performance. Our project extracted the topics of subreddits from their popular titles and extracted the expertise of each expert from their qualified comments. The algorithm for extracting is critical. Even if both RAKE and TextRank can extract the keywords from text, research from Ganiger and Rajashekharaiah [2] pointed out that RAKE generates the important and weighted keywords faster than other algorithms. Plus, Baruni and Sathiaseelan [1] made a comparison between RAKE and TextRank. They found that RAKE has higher precision than TextRank and comparable recall scores.

Apart from keyword extraction, another important application of TextRank is summarization, selecting similar sentences from a text to make a summary. One fact we can not ignore is that even if we picked the comments from users with high scores, it doesn't mean each of these comments is relevant to the topic. Therefore, we use TextRank to select sentences with similar meanings to ensure that these comments are as relevant as possible before extracting keywords.

There is a special circumstance that one subreddit named *China_irl* has comments and titles in Chinese. Since RAKE is not suitable for Chinese, TextRank was used to extract the topics and expertises instead of RAKE.

2.2.3 Libraries for achieving algorithms

We applied the two algorithms with the support of implemented libraries. *python-rake* provides the function *RAKE()* to implement the RAKE algorithm and *gensim* provides the function *summarize()* to work with the TextRank algorithm. *jieba* is the specified library to dealing with Chinese text, providing the function of splitting text and *textrank()* interface.

3. RESEARCH QUESTIONS

The goal of this research can be split into two parts. First, we want to determine the 1000 most popular subreddits. For each subreddit we aimed to determine the topics of the subreddit and to visualize them. Second, we wanted to find the top-20 expert users for each subreddit after finding the 1000 most popular and find the topics that each expert was talking about. For each parts of the goal, there are several sub-questions.
For the first part:

- How to find the top 1000 popular subreddits or what attributes can help us to find them?
- How to find the topic of subreddits?
- How to visualize the topic?

For the second part:

- What kind of users can be called experts?
- How to find the expertise of experts from there comments
- How to visualize the expertise of experts?

Table 1: Attributes of comments

author	author_flair_css_class	author_flair_text
body	controversiality	created_utc
distinguished	edited	gilded
id	link_id	parent_id
retrieved_on	score	stickied
subreddit	subreddit_id	ups

4. PROJECT SETUP

In this section we will discuss our input data, the data pipeline and our output data in this section.

4.1 Input data

The provided dataset consisted of two parts, submissions and comments, adding up to 1.3TB of raw data. Both the comments and the submissions considered the period of 2005 to 2022 and each month had its own *bz2* file, consisting of compressed JSON files. The distribution of the data over the years was very skew, which can be seen in Figure 3 and 4. Since the amount of submissions and comments created grew over time in an almost exponential way, most of the data was in the period of 2019-2022, so we decided to only use the data from that period. The data we used starts at 2019-08 and ends at 2022-08, giving us the last 3 years of data. This data would also give a more actual view of the most popular subreddits and their experts.

4.1.1 Attributes

The data in the dataset has a lot of attributes, but not all attributes were useful for the research. All the attributes from the comments can be seen in Table 1. The attributes from the submissions are in Table 2.

Since not all attributes were useful for our research we only used a few attributes. The attributes that were used for the comments can be found in table 3 and the attributes from the submissions that were used are in table 4.

4.2 The Data Pipeline

In this section the different steps of the proposed data pipeline are explained. There is also a discussion of the thought process and the choices made for it. A schematic overview of our pipeline can be seen in Figure 5

4.2.1 Writing the data to parquet files

The provided dataset consisted of many *bz2* files, which had a considerable amount of data that was not relevant for the analysis, as described in Section 4.1.1, and, considering that Spark works efficiently when reading parquet files, it was decided to write the data from the period of 2019-08 to 2022-08 to parquet files, where only the required attributes for the analysis were selected. This made the size of the data considerably smaller and Spark could read this data in a faster way. It was not possible to read the data from the comments for 2020-04, 2020-05 and 2020-08, and because of that, it was not considered for the analysis either. And, in order to maintain consistency in the data, the submissions for those same months were also not considered.

4.2.2 Selecting the 1000 most popular subreddits

Since it was needed to find the 1000 most popular subreddits in order to select the experts, this was the next step in

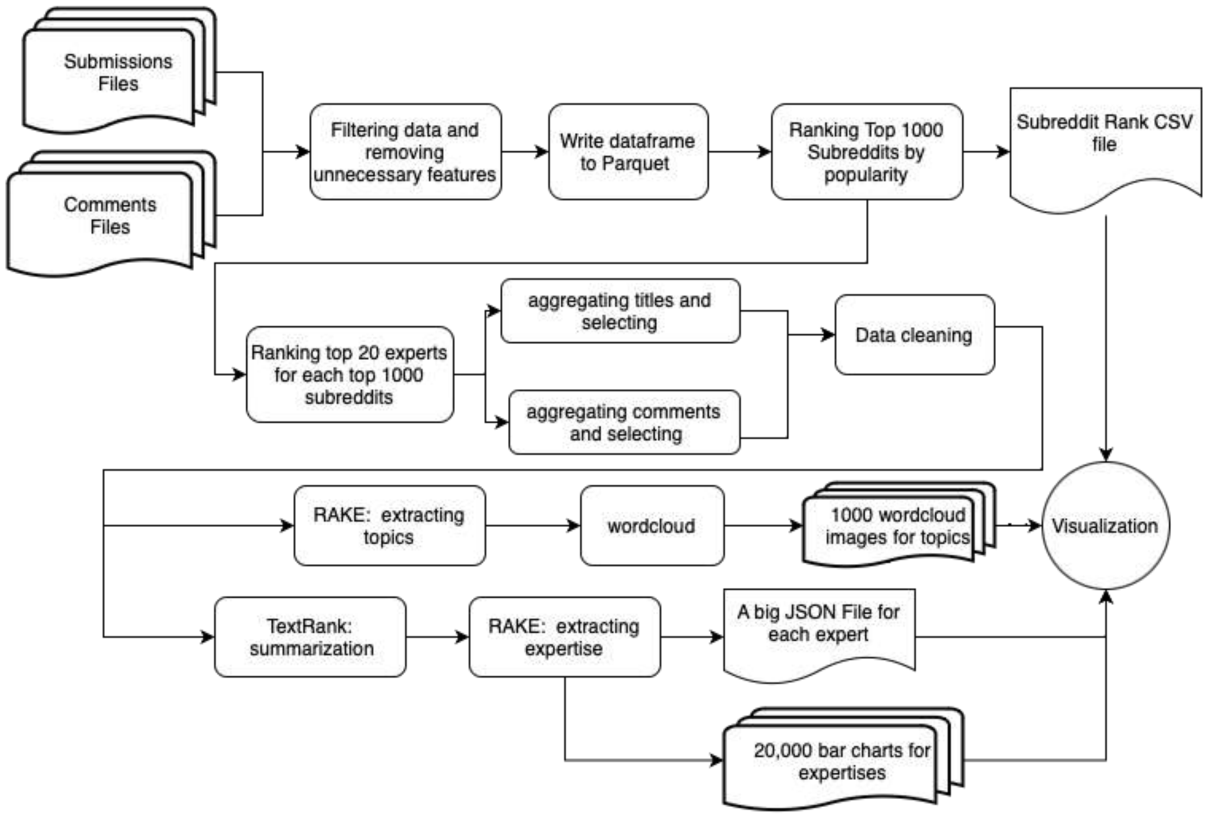

Submissions Files
Comments Files
Filtering data and removing unnecessary features
Write dataframe to Parquet
Ranking Top 1000 Subreddits by popularity
Subreddit Rank CSV file
Ranking top 20 experts for each top 1000 subreddits
aggregating titles and selecting
aggregating comments and selecting
Data cleaning
RAKE: extracting topics
wordcloud
1000 wordcloud images for topics
Visualization
TextRank: summarization
RAKE: extracting expertise
A big JSON File for each expert
20,000 bar charts for expertises

Figure 5: Data Pipeline

the pipeline. It was decided to rank the subreddits based on the total amount of comments a subreddit got on all the posts. The attribute **num_comments** from submissions was used for this, since this would be faster than counting all the comments for each subreddit. All our submission data was merged into a large dataframe and grouped by subreddit. The attribute **num_comments** was summed to get the total amount of comments and the subreddits were ordered on from greatest amount of comments to the smallest amount of comments. The total sum of the attribute **score** for the submissions was also summed so it could be used as a second ordering condition. The score was ordered in descending order as well. After that the first 1000 subreddits were selected. This took around 30 seconds to complete.

4.2.3 Selecting the top-20 experts

In order to find the top-20 expert users for each subreddit we only looked at the comments each user made, since most discussions on Reddit take place in the comments. All the comments were put into one big data frame and filtered the comments that were made in other subreddits that were not in the list of the 1000 most popular subreddits. The comments were grouped for each subreddit by author and ordered by score in descending order. Only the 500 comments with the best score for each author were kept for each subreddit, so that the quality of the comments would be more important than the quantity. The scores of these comments were summed for each user in each subreddit. The 20 users with the highest total scores for a subreddit were selected as the expert users. This also took around 30 seconds.

4.2.4 Extracting the topics from the subreddits

The keyword extraction algorithm RAKE was used to find the topics of the subreddits. For each subreddit the 1000 posts with the highest scores were selected. The topics were extracted from the titles of the posts, because most of the discussions in the comments are related to the post and the title of the post. The titles of the posts had to be cleaned before RAKE could be used, to improve the performance of RAKE. All emojis and most of the URLs were removed from the titles. Some subreddits also use tags like [**serious**] in the title, so these tags were removed as well. After that RAKE was run on each title and the results were collected in a list. This took around 10 minutes to run. This didn't work well for some subreddits because all the titles would be the same. In the subreddit 'maybemaybemaybe' every title would be 'maybe maybe maybe'. The package 'wordcloud' from Python was used to generate wordclouds from the list that was generated with RAKE for each subreddit. A wordcloud was chosen as visualization because it could

Table 2: Attributes of submissions

archived	author
author_flair_background_color	author_flair_css_class
author_flair_richtext	author_flair_text
author_flair_text_color	author_flair_type
brand_safe	can_gild
contest_mode	created_utc
distinguished	domain
edited	gilded
hidden	hide_score
id	is_crosspostable
is_reddit_media_domain	is_self
is_video	link_flair_css_class
link_flair_richtext	link_flair_text
link_flair_text_color	link_flair_type
locked	media
no_follow	num_comments
num_crossposts	over_18
parent_whitelist_status	permalink
post_hint	preview
retrieved_on	rte_mode
score	secure_media
selftext	send_replies
spoiler	stickied
subreddit	subreddit_id
subreddit_name_prefixed	subreddit_type
suggested_sort	thumbnail
thumbnail_height	thumbnail_width
title	url
whitelist_status	

Attribute	Description
subreddit	The name of the subreddit the comment was made in.
subreddit_id	The unique id from the subreddit the comment was made in.
author	The unique username of the author.
score	The score of the comment, which can be calculated by substracting the amount of downvotes from the amount of upvotes that the comment got.
body	The text from the comment that the author has made.

Table 3: The attributes from the comments that were used with their descriptions.

Attribute	Description
subreddit	The name of the subreddit the post was made in.
subreddit_id	The unique id from the subreddit the post was made in.
num_comments	The amount of comments a post got.
score	The score of the submission, which can be calculated by substracting the amount of downvotes from the amount of upvotes that the post got.
title	The title of the post.

Table 4: The attributes from the submissions that were used with their descriptions.

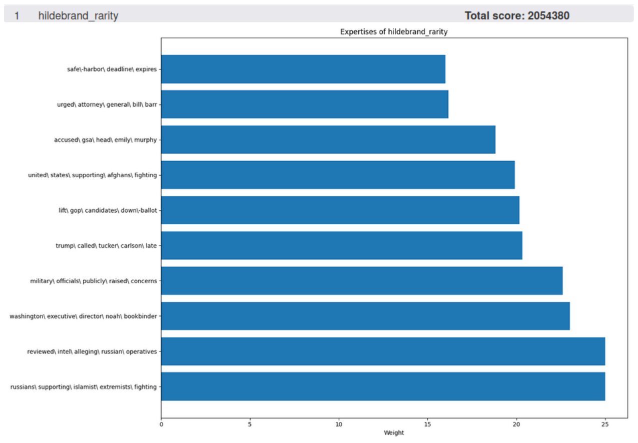


Figure 7: Expertises from 'hildebrand_rarity' on the subreddit 'politics'

give a quick overview of the found topics, while still providing some information about the importance of the topics.

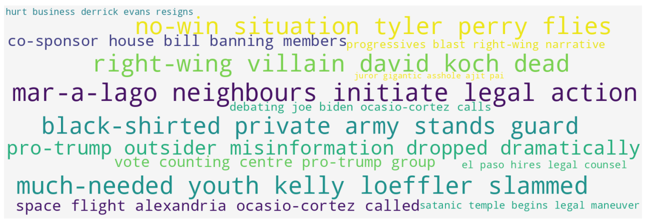


Figure 6: Wordcloud for the 'politics' subreddit

An example of a generated wordcloud can be seen in Figure 6. This wordcloud was for the 'politics' subreddit. Some recent events in the United States can be found in the wordcloud, like the controversy around Mar-a-Lago, the golfcourse of the former president Trump.

4.2.5 *Extracting the topics of the experts*

The comments were used to look for the experts since most of the discussions on Reddit are in the comments. The 500 comments with the highest score were selected for each expert. The comments had to be cleaned as well before RAKE could be used. The comments contained a lot of emojis and URLs which had to be removed. After that the package 'gensim.summarization' from Python was used to summarize the comments, so the important comments could be selected and the amount of time that RAKE would take could be reduced. This package uses a variation of the TextRank algorithm for this. After that RAKE was run on the remaining comments and the 10 results with the highest weight were selected for each expert. The results were placed in a horizontal bar chart. A bar chart was chosen because it would be easier to see the weight of each topic. The found expertises of the top expert of the 'politics' subreddit can be found in Figure 7.

In order to create the plots special characters in the expertises had to be escaped, which also generated a backward slash in front of every whitespace.

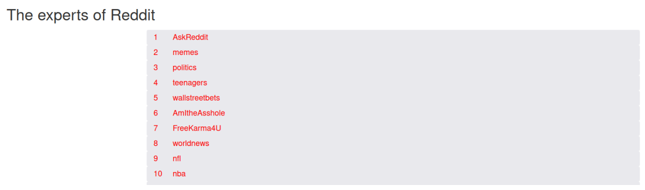
4.3 The output data

The output data contains the 1000 most popular subreddits, the top-20 expert users for each subreddit and the topics from the subreddits and the topics from the experts. The 1000 most popular subreddits are stored in a CSV file. This file includes the total amount of comments for each subreddit and the total score from submissions on that subreddit. All the experts are stored in a JSON file. The total score of the user and his expertises are also stored in the file.

Furthermore there are 1000 images with wordclouds, representing the topics of the subreddits. There are also 20000 images with the bar charts, one for each expert, representing the expertises of the experts.

4.3.1 *Visualization of the data*

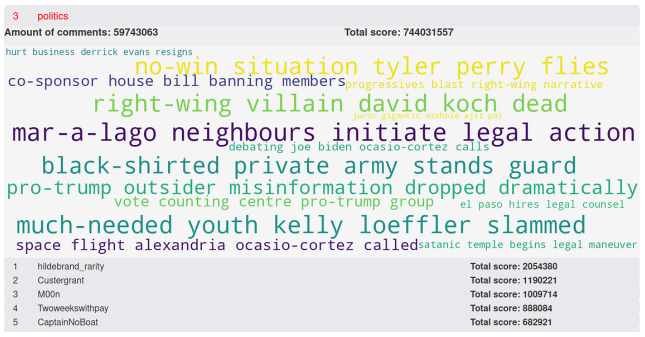
The output data has been displayed in a static HTML page. The website contains a list of the 1000 most popular subreddits, starting at the most popular subreddit. When you click on a subreddit it will show the wordcloud with the topics of the subreddit and it will show the top-20 experts for that subreddit. If you click on a expert you will see the expertises of the expert. Pictures of the website can be seen in Figure 8 and 9 The website has been generated using the Python Flask framework.



The experts of Reddit

1 AskReddit
2 memes
3 politics
4 teenagers
5 wallstreetbets
6 AmItheAsshole
7 FreeKarma4U
8 worldnews
9 nfl
10 nba

Figure 8: The most popular subreddits on the website



3 politics

Amount of comments: 59743063 **Total score: 744031557**

hurt business derrick evans resigns
no-win situation tyler perry flies
co-sponsor house bill banning members
progressives blast right-wing narrative
right-wing villain david koch dead
juror gigantic asshole ajit pai
mar-a-lago neighbours initiate legal action
debating joe biden ocasio-cortez calls
black-shirted private army stands guard
pro-trump outsider misinformation dropped dramatically
vote counting centre pro-trump group
el paso hires legal counsel
much-needed youth kelly loeffler slammed
space flight alexandria ocasio-cortez called
satanic temple begins legal maneuver

1 hildebrand_rarity **Total score: 2054380**
2 Custergrant **Total score: 1190221**
3 M00n **Total score: 1009714**
4 Twoweekswithpay **Total score: 888084**
5 CaptainNoBoat **Total score: 682921**

Figure 9: Opening the 'politics' subreddit on the website

5. CONCLUSIONS

In this paper we looked at finding the 1000 most popular subreddits and finding the top-20 expert users for each subreddit. The most popular subreddits were selected by taking the subreddits with the highest amount of comments. The experts for each subreddit were selected by summing the scores of the 500 comments with the highest score for each user and taking the users with the highest total score. The topics of the subreddits were determined by using RAKE on the titles of the submissions for each subreddit and a wordcloud was generated for each subreddit. The expertises of the experts were determined by using RAKE on the 500 most popular comments for each expert. The expertises were displayed in horizontal bar chart.

The challenge of working with big data was a very clear learning process in which the group was able to first-hand tackle the problems that come with, at the very least, reading these massive amounts of information, and to find ways of working with this data by parellizing and sampling and extrapolating the data. The fact that we had access to a shared cluster sometimes resulted in slower operation because of the division of the compute. This was solved by creating a dedicated cluster for our operations. It was also quite a challenge to read this data and transforming it into parquet files, considering that, if one of the tasks would fail, the whole command would have had to be run again, resulting in millions of files, which happened to the group at some point.

We are satisfied with our results. However, if we had the opportunity to continue doing this project, we think we could improve it in at least three aspects. First, a more comprehensive removal of elements that should not appear in the text. There are still some unnecessary elements that we did not think of before that appear in our results. Second, more careful selection of subreddit. Not all subreddits are suitable for extracting the topic or finding experts from them. For example, the subreddit we mentioned in 4.2.4, of which almost comments are the same sentence 'maybemaybemaybe'. At last, using a supervised learning method to extract the expertis. While extracting keywords from the submission titles gave decent results for the topics of the subreddits, it didn't work too well for extracting the expertises of the experts. Some results don't show the much information about the expertises of the experts. We think it could be better that we select some experts and attached the expertise tags to them manually as the training dataset to train a model, then apply this model to other experts.

6. WORK DIVISION

Max de Boer	Giancarlo Ianni	Zhe Liu
Writing the data to parquet	Data reading	Writing data to the parquet
Parallelizing expertise extraction	Initial data investigation	Selecting algorithms
Visualizing the results	Report Writing	Selecting top subreddits and experts
Writing the report	Presentation	Extracting topics and expertises
Error Handling		Report Writing and presentation

Table 5: The work division

References

[1] J Baruni and J Sathiaseelan. "Keyphrase Extraction from Document Using RAKE and TextRank Algorithms". In: *Int. J. Comput. Sci. Mob. Comput* 9 (2020), pp. 83–93.

[2] Shweta Ganiger and KMM Rajashekharaiah. "Comparative study on keyword extraction algorithms for single extractive document". In: *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE. 2018, pp. 1284–1287.

[3] *Homepage*. URL: `https://www.redditinc.com/`.

[4] Rada Mihalcea and Paul Tarau. "Textrank: Bringing order into text". In: *Proceedings of the 2004 conference on empirical methods in natural language processing*. 2004, pp. 404–411.

[5] Stuart Rose et al. "Automatic keyword extraction from individual documents". In: *Text mining: applications and theory* (2010), pp. 1–20.

[6] MG Thushara, Tadi Mownika, and Ritika Mangamuru. "A comparative study on different keyword extraction algorithms". In: *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE. 2019, pp. 969–973.