# Performance analysis of the Oráculo framework for data collection from Twitter

# Análise do desempenho do Framework Oráculo para coletas no Twitter

DOI:10.34117/bjdv6n12-549

Recebimento dos originais: 21/11/2020
Aceitação para publicação: 21/12/2020

**Hércules Batista de Oliveira**
Mestre em Educação - PPGED/UFVJM
Instituto Federal de Ciência, Educação e Tecnologia - IFNMG/Campus Diamantina
Fazenda Biribiri, Rod. MGT 367, KM 624, s/n°, Rodovia 367, Diamantina - MG
hercules.oliveira@ifnmg.edu.br

**Marcus Vinicius Carvalho Guelpeli**
Doutor em Computação - UFF / FCUL - Lisboa
Universidade Federal dos Vales do Jequitinhonha e Mucuri – UFVJM
Rod. MGT 367, 5000 - Alto da Jacuba, Diamantina - MG
marcus.guelpeli@ufvjm.edu.br

**ABSTRACT**
Online social networks are important social spaces for human interaction, with far-reaching applications in communication, entertainment, advertising, social campaigning and community empowerment. Shared data have become a research source for several studies seeking to analyze user interactions in these networks. Because of the large volume of data produced, text mining techniques are required for analyzing the collected data efficiently. One of the challenges of the text mining process is the lack of direct access to data from online social networks, which requires the use of specialized tools for collecting data. The present study conducts a performance analysis of Oráculo Application Development Framework as a tool for collecting and mining texts shared on the social network Twitter. In this framework, different algorithms and techniques were applied to circumvent the limitations imposed by the Twitter API. Performance tests were conducted comparing the Oráculo and DMI-TCAT algorithms. The results show that Oráculo presents superior performance in the number of tweets collected compared to DMI-TCAT considering the algorithms and scenarios analyzed.

**Keywords:** performance analysis, text collector, framework, Twitter.

**RESUMO**
As redes sociais online são espaços sociais importantes para a interação humana, com aplicações de longo alcance em comunicação, entretenimento, publicidade, campanhas sociais e fortalecimento da comunidade. Os dados compartilhados têm se tornado fonte de pesquisa para diversos estudos que buscam analisar as interações dos usuários nessas redes. Devido ao grande volume de dados produzidos, técnicas de mineração de texto são necessárias para analisar os dados coletados de forma eficiente. Um dos desafios do processo de mineração de texto é a falta de acesso direto aos dados das redes sociais online, o que requer o uso de ferramentas especializadas de coleta de dados. O presente estudo realiza uma análise de desempenho do Oráculo Application Development Framework como uma ferramenta de coleta e mineração de textos compartilhados na rede social Twitter. Neste framework, diferentes algoritmos e técnicas foram aplicados para contornar as limitações impostas pela API do Twitter. Testes de desempenho foram realizados comparando os algoritmos Oráculo e DMI-

TCAT. Os resultados mostram que o Oráculo apresenta desempenho superior no número de tweets coletados em relação ao DMI-TCAT considerando os algoritmos e cenários analisados.

**Palavras-chave:** análise de desempenho, coletor de texto, framework, Twitter.

## 1 INTRODUCTION

Online social networks occupy relevant space in the lives of their users. With far-reaching applications in communication, entertainment, advertising, social campaigning and community empowerment, social networks have billions of users throughout the world who spend a growing amount of time connected to them [1]. According to Recuero [2], online social networks represent a new and complex universe of communicative, social and discursive phenomena. By recording these social dynamics and registering their access, researchers are able to map and study interactions and conversations in a large scale.

These systems generally allow decentralized production of content, people interaction and mobilization, and create groups with common interests, thus facilitating structure and organization [3]. These characteristics, along with the large number of users, lead to expressive data creation and volume.

As a result, accurate computational techniques are necessary for analyzing huge piles of data from which to obtain useful knowledge. Existing techniques include text mining, which can be defined as the use of model-based methods for finding patterns, summarizing texts or making predictions, and thus extracting knowledge from texts [4]. Text mining encompasses the stages of collection, preprocessing, indexing, mining, visualization and analysis [5]. Since there is no direct access to online social data, specialized tools are required to conduct the collection process.

The microblogging and social networking service Twitter was chosen as the data source for this study, since its primary goal is sharing texts, or tweeting. According to Ausserhofer and Maireder [6], Twitter is a social network that broke barriers for participation and debate due to the public nature of communication developed in its space. The social relevance of Twitter has led to studies in different fields of knowledge, such as education [7, 8], social movements [9, 3, 10], politics and elections [11, 12], health [13, 14], among others.

Although there are tools for collecting and analyzing tweets that are licensed as free software, such as Digital Methods Initiative Twitter Capture and Analysis Toolset (DMI-TCAT) [15] and yourTwapperKeeper - yTk [16], these tools generally use an Application Programming Interface (API), which Twitter offers for collecting its data. In order to avoid abuses, the Twitter API imposes access limits and restrictions to the data that can be recovered [17]; thus, texts collected with these tools are restricted to the limitations imposed by the social network.

The objective of this study is to analyze the results of the development of a tool, called Oráculo Application Development Framework, for collecting and mining texts in online social networks applied to Twitter. Another related objective is to quantitatively describe the performance of the developed tool compared to another similar tool, verify the relationship between location and internet access and the results of Twitter data collection, compare the performance of different algorithms available in the Oráculo framework regarding the performance of the local and distributed data collection, and create a web interface that is friendly for this tool.

It is possible to identify several recent and relevant studies in academic literature that use a restricted number of tweets as samples in its discussions: 900 tweets [9], 5000 tweets per day [12], 54,611 tweets [11]. In contrast, studies that employ property and commercial tools for collecting their tweets, which is unavailable to every researcher, analyzed a considerably higher number of tweets: 6 million tweets [14] and 15 million tweets [13].

It is suggested that a collection tool that has the resources to circumvent the limitations imposed by the Twitter API would render it possible for these and other studies to use a wider sample. With a user-friendly interface, available in a web platform, this tool would assist and enhance research by users who are unfamiliar with the computing field as it allows them to collect data and perform analyses in a more practical manner.

In this context, this article presents a performance analysis of an innovative data collection tool for online social networks applied to Twitter. This tool has a distributed data collection structure and different algorithms and established techniques to circumvent the requisition limitations imposed by the Twitter API. This framework is available as free software for researchers from different fields of knowledge. It also presents a user-friendly and intuitive web interface, thereby assisting researchers who are unfamiliar with the computing field to perform text collection in Twitter and text mining in this collection. This article also presents a study of the relation between the collector's geographic location and the results achieved.

## 2 THEORETICAL REFERENCES

This section presents the theoretical assumptions underlying this research: social networks, Twitter, challenges of Twitter data collection, and data collection tools in this network.

### 2.1 SOCIAL NETWORKS

Benevenuto et al. [18] define online social networks as a web service that enables individuals to build public or semipublic profiles within a system, so as to articulate a list of other users (friends,

followers) with which they share connections, in order to view or interact with their connection lists and other lists made by other system users.

Among the available social networks, Twitter was chosen as the source of data for this research because its primary object is sharing texts, which can be mined for knowledge extraction. This choice was also relevant because Twitter provides an API for developers to have access to their texts and users.

In addition to these characteristics, moreover, most accounts are public and information circulates openly, with a lower incidence of algorithms that can restrict content displayed to other users [11].

## 2.2 TWITTER

Twitter is a short-text online social network featuring as a microblogging service. In this network, users share messages, called tweets, containing up to 280 characters that may include alphanumeric elements, photos and videos [19]. This network was launched in 2006 and currently has about 330 million monthly active users, of which 136 million users access Twitter on a daily basis [20] and produce, on average, 500 million tweets per day [21].

In order to be able to access the large volume of texts produced by Twitter, the social network offers developers a set of procedures and standards, or an Application Programming Interface (API), which enables access to tweets, user status and user data [17].

In order to curb abuse and avoid server overload, Twitter imposes limitations on data access through its API. Three different access levels are available in this API: Standard, Premium and Enterprise. The difference between these access levels is the number of monthly requests that can be made and especially the ability to backlog a search. The Standard version is able to view tweets posted within seven days, the Premium and Enterprise versions range from 30 days to full history, depending on the selected package. In addition to this restriction, the number of requests per second, per minute and within a 15-minute window is also limited [17].

Standard API level is free and available to all of the social network's users. The cost of accessing the paid versions of Twitter start at US$99.00 a month and reach thousands of dollars, depending on the number of requests made to the API [17].

### 2.2.1 Challenges of Twitter Data Collection

Bruns and Liang [16] point out that the challenges faced in developing tools for conducting Twitter data collection and analysis are related to three points: Twitter API limitations, scalability and timeliness.

Regarding the limitation of the Twitter API, the strict access control policy should be highlighted, with restrictive limitations on information retrieval through internet connection and access, originating from a specific IP address or access key. Recurring requests from the same requester face limitations on the number of queries over a given period of time. Another challenge regarding the Twitter API is the free version's seven-day limit to retrieve historical data [17]. By restricting the period and, consequently, the sample, surveys may become unfeasible, since they require a larger sample [16].

### 2.2.2 Tools for collecting Twitter data

There are open source initiatives and tools available for collecting Twitter data that researchers can apply in their own research areas in order to generate comparable data sets and replicable studies. Among them, the authors highlight DMI-TCAT and Scrapy for playing a particularly important role in the development of this study. DMI-TCAT was chosen as a benchmark for comparing the performance of the Oráculo framework, as it has been used by the research community since 2014. Scrapy is a web crawling framework that has been applied to one of Oráculo's available algorithms.

The Digital Methods Initiative Twitter Capture and Analysis Toolset (DMI-TCAT) is a tool designed to collect and analyze academically oriented tweets primarily in the fields of humanities and social sciences [15]. DMI-TCAT uses the Search API to access historical tweets and Streaming API for real-time tweets, and is thus restricted to the limits set by this API.

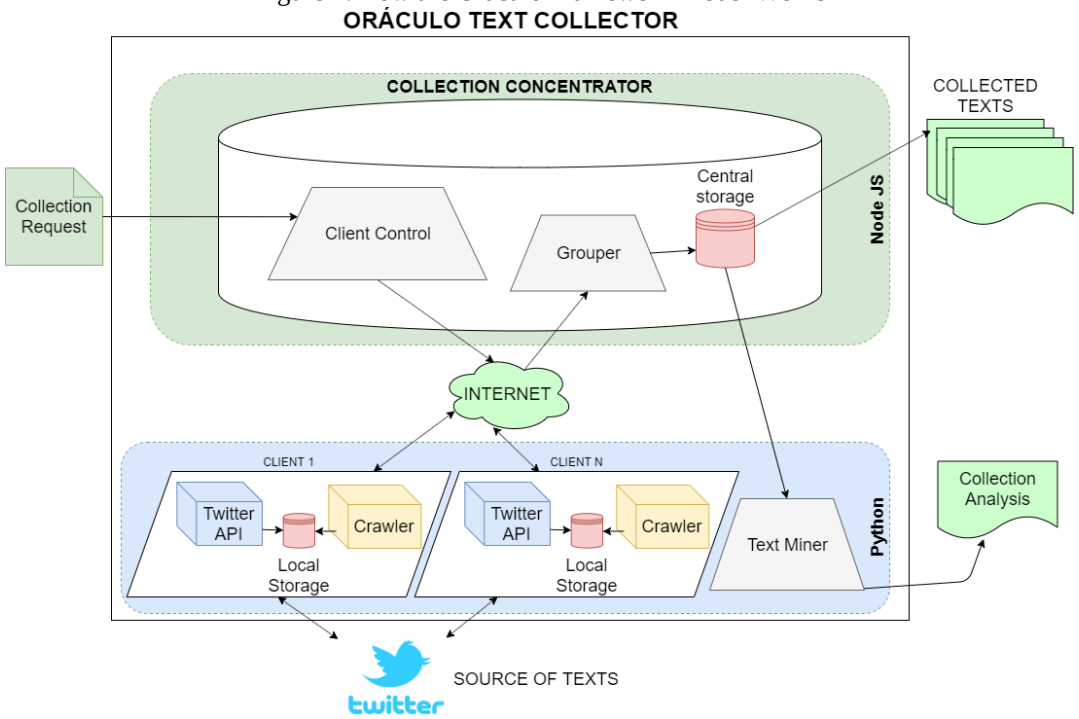
The Scrapy framework is a robust set of web scraping features. Developed in Python, this open source framework can be used to retrieve information via HTTP protocol from various data sources [22].

Using Scrapy for twitter text collection is described by Hernandez-Suarez [23] as a viable option to circumvent the limitations of the Twitter API, with superior results when compared to other applications using the API.

## 3 ORÁCULO APPLICATION FRAMEWORK

The Oráculo Application Framework is composed of different layers. This research was dedicated to developing the Oráculo Text Collection and Mining Layer framework. The other layers of the framework –Bots Filter and User Interface – are addressed in other studies by the Text Mining, Natural Language Processing and Machine Learning Research Group (Grupo de Pesquisa em Mineração de Textos, Processamento de Linguagem Natural e Aprendizagem de Máquina - MTPLNAM), from the Computer Science Department of the Federal University of Jequitinhonha and Mucuri Valleys - UFVJM. Figure 1 represents the Oráculo framework layer model that was developed and applied in the present research:

Figure 1: How the Oráculo Framework Model Works




Source: Created by the author.

The text collection and mining layer consists of the collection concentrator module and client module instances. The collection concentrator is responsible for receiving the collection request in its web interface and performing the corresponding negotiations to execute it. This module then uses the client control class to schedule a collection request for the configured client module instances.

In this request, the following information must be present: query term, algorithm used in the data collection, start and end dates of collection, type of data output and number of nodes used in the collection.

The requested collection can refer to real-time type texts for tweets posted after the collection begins, or retroactive to tweets posted before the collection began. For real-time collections, the only available option is to collect data through the Twitter API. In historical collections, the user can select the collection through a Crawler or the Twitter API.

For data output, the user can select individual JavaScript Object Notation - JSON files or from a MongoDB database collection.

The number of client nodes that can be used for data collection varies depending on the number of configured client instances. In the tests conducted by the present research, two nodes were used, one in Brazil and another in the United States. The user can then select between the local collection using a single node, the collection distributed between two nodes identically, and the complementary distributed collection. The latter option partitions the collection into two distinct periods that are collected and then joined to form a single collection.

The collection concentrator module was developed with the NodeJS language, which is a JavaScript language interpreter that executes its code on the server. One of the features of this language is building a single, asynchronous and non-blocking thread web interface; i.e., despite working on only one thread, the processing of requests will not be blocked while waiting for a request to finish in order to keep on processing the remaining system requests. These features provide scalability and a real possibility for parallelism in applications developed with NodeJS [24]. Thus, NodeJS was chosen for the development of the collection concentrator class to enable the management of multiple client requests, disk and database access without affecting system performance.

The client module consists of the collection class through the Twitter API, the collection class through a Crawler using the Scrapy framework, and local storage, which can be accomplished through JSON files or a MongoDB database. Upon receiving a request for data collection, the client module identifies the terms that will be collected, the type of collection, the time period and the algorithm that will be used. Depending on the algorithm selected, the client uses the corresponding class, whether related to the Twitter API or Crawler, and performs the collection, saving the corresponding data in Local Storage.

After the collection is completed, the client returns the collected information to the Collection Concentrator module, which in turn must wait for all clients to complete this procedure before processing the grouping and delivering the data to the user.

The client module can be installed and configured on multiple geographically distributed computers, thus creating multiple collection instances. These multiple instances aim to increase the number of collected tweets and circumvent access limitations through the Twitter API, for there will be different accesses through different internet connections and Twitter access keys, so that Twitter does not correlate these accesses. It can also prevent variation in the quality of the internet access link and possible network failures from interfering with the collection capacity at that time, since access redundancy makes the system more tolerant to failures.

The client module was developed using Python programming language. This language was chosen due to the extensive documentation and libraries for Twitter access. Another reason is that there is support for this language in various hosting sites around the world, which makes it easy to instantiate geographically distributed clients.

In association with the text collector, a text miner was developed to provide analysis of the collections made. This miner provides information on the number of tweets collected, the number of terms present in the collection, the frequency of the terms collected with graphs of this frequency, and the word cloud of the present terms. The language used to develop the text miner was Python, again

due to the number of libraries available in this language, so that it could be instantiated next to the client module in the different nodes where Oráculo is present.

### 3.1 ENVIRONMENT OF TESTING FRAMEWORKS

The text collections were performed on different computers, albeit with similar hardware configurations: four processing cores, two threads per core and with at least eight Gigabytes of RAM. However, during testing, the collection algorithms were found to use only one thread and consume minimal computational resources.

The nodes that performed the tests were accessed online from distinct and geographically distributed locations, each with 100 Megabits per second - Mbps, so that the network traffic generated by one collection would not interfere with the others. These nodes are distributed between Brazil and the United States of America - USA and were thus identified in the test results.

In the collections using the Twitter API, different access and identification keys of the application were used, once again, so that there would be no interference in the results of different collections due to the limitations imposed by the API.

### 3.2 TESTING FRAMEWORKS

In order to verify Oráculo's execution, performance tests were conducted for each available algorithm: Historical API, in different nodes and with complementary collections; Crawler, in different nodes and with complementary collections; and Realtime API on each node available in the framework.

These Oráculo framework tests were accompanied by tests from the DMI-TCAT, a Twitter collection tool with a web interface. Thus, the objective was to compare performance with a tool that is recognized and employed by several researchers.

The tests were performed in collections with a set time, every five, fifteen, thirty and sixty minutes, so as to verify the evolution in the number of tweets collected in each tool and algorithm over time. The collections began and ended on all algorithms and client nodes at the same time, by previously synchronizing the computer clocks used as nodes.

From the collections performed, the number of tweets collected within the given time frame, number of words in each collection and frequency of query term within the collection set were analyzed. This analysis was performed with the text miner of each tool: Oráculo framework and DMI-TCAT, respectively.

Due to the restriction imposed by the Twitter API of a maximum of 180 requests within a fifteen-minute time frame, this was the selected interval for their collection to be mined. Thus, we sought to equalize the comparison between available algorithms and tools used.

The query term used in all tests of this research was “bolsonaro”, surname of the current Brazilian president. This term was chosen to ensure that tweets were available throughout the collection period, as there has been considerable engagement with the President’s image on this platform since the 2018 presidential election [12].

## 4 RESULTS AND DISCUSSIONS

This section presents the results of the performance tests comparing Oráculo framework and DMI-TCAT.

### 4.1 QUANTITATIVE ANALYSIS

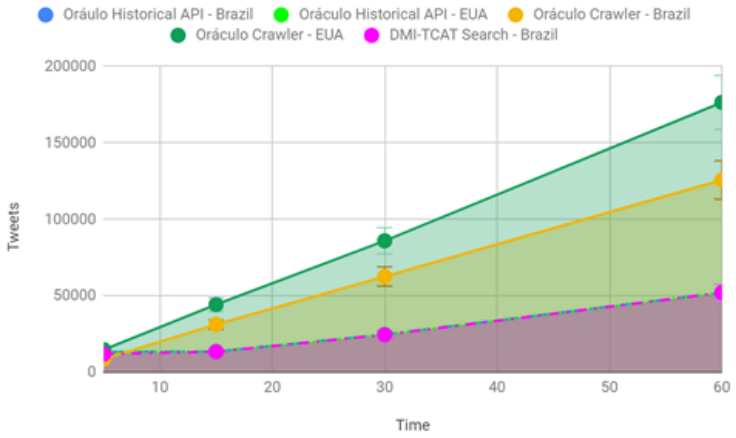
The tested data were collected between June 20 and 30, 2019, synchronized according to this study’s methodology. Table 1 presents the results for the historical data collection, which occurred with each algorithm available on Oráculo, and with the DMI-TCAT:

Table 1: Tweets collected in tests for historical algorithms. Source: Created by the author.

| Algorithm | 5 min | 15 min | 30 min | 60 min | Average tweets per minute |
|---|---|---|---|---|---|
| Oráculo Historical API - Brazil | 13304 | 13494 | 24685 | 52091 | 941.58 |
| Oráculo Historical API - EUA | 13304 | 13493 | 24684 | 52092 | 941.57 |
| Oráculo Crawler - Brazil | 8703 | 31238 | 62634 | 125734 | 2075.54 |
| Oráculo Crawler - EUA | 14767 | 44234 | 85956 | 176474 | 2922.10 |
| DMI-TCAT Search - Brazil | 11861 | 13457 | 24494 | 52012 | 925.67 |

The data collection was performed for each interval, and does not present associations with the previous interval. This way, it was possible to verify a trend in the stable evolution of number of tweets collected according to the increase in time interval. Figure 2 presents this trend in graph form:

Figure 2 - Graph showing the number of collected tweets in tests for historical algorithms




Source: Created by the author.

The authors noted that nodes using the Twitter API-based algorithm achieved similar performance in both Oráculo and DMI-TCAT. Due to the limited number of twitter requests – every 15 minutes, the results for 5 and 15 minutes are close, since the algorithms consume these requests quickly and wait for the next 15-minute window to make new requests.

Although they use the same algorithm, nodes in Brazil and the United States running the Crawler option obtained considerably different results: the node in the United States was, on average, 40% more efficient compared to Brazil. Because this algorithm accesses the Twitter search page through a HTTP protocol, the limitation for data return is only in network communication.

As both internet access links on the nodes were identical, it was necessary to analyze the average response latency of the Twitter server. At this point, it was observed that the average latency of the Brazilian node was 122 (one hundred and twenty-two) milliseconds and average node latency located in the United States was 5 (five) milliseconds. This difference in response time explains the difference in performance in this type of data collection.

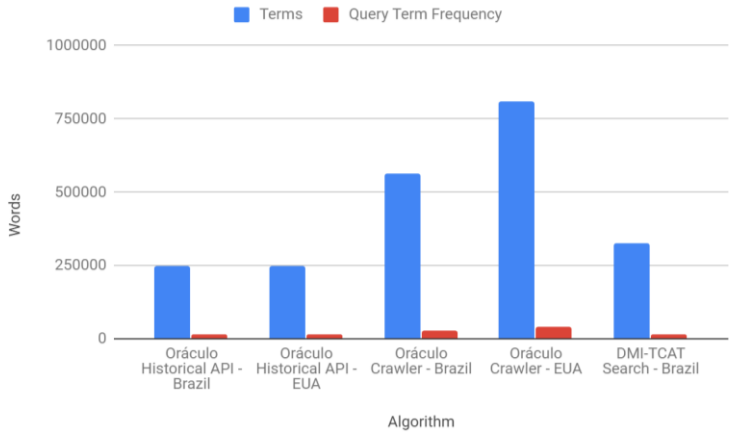
The historical data collections, performed in a 15-minute interval, were analyzed with the Oráculo and DMI-TCAT text miner, considering the number of terms present, frequency of query term used, and the percentage of incidence that this frequency represents in total number of terms. Table 2 and Figure 3 present these results:

Table 2: Analysis of tweets in tests for historical algorithms.

| Algorithm | Number of terms | Frequency of consulted term | Frequency (%) |
|---|---|---|---|
| Oráculo Historical API - Brazil | 249438 | 14169 | 5.68% |
| Oráculo Historical API - EUA | 249427 | 14168 | 5.68% |
| Oráculo Crawler - Brazil | 561501 | 28322 | 5.04% |
| Oráculo Crawler - EUA | 808468 | 40341 | 4.99% |
| DMI-TCAT Search - Brazil | 326392 | 13945 | 4.27% |

Source: Created by the author

Figure 3: Analysis of non-distributed historical data collection




Source: Created by the author

When analyzing data from Table 2, it can be noted that, although the number of tweets is different between algorithms, the incidence of the query term within the collected data remains similar. Differences in the DMI-TCAT method, collection of stopwords, and text mining algorithms may explain Oráculo's advantage.

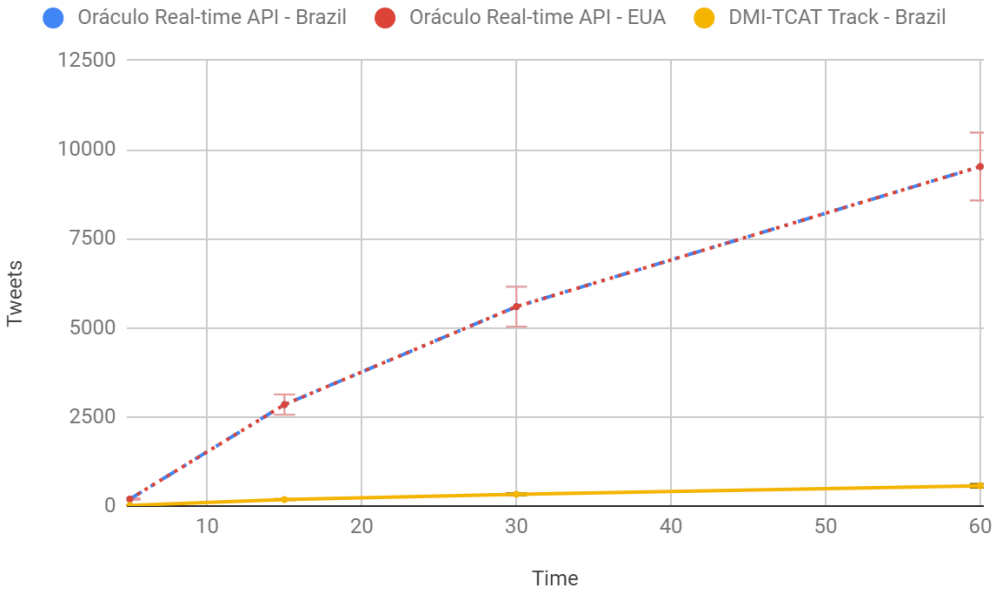
For collections performed with the real-time algorithm, nodes were retested with Oráculo and DMI-TCAT. Table 3 and Figure 4 show the results of this collection:

Table 3: Tweets collected in realtime algorithm tests

| Algorithm | 5 min | 15 min | 30 min | 60 min | Average Tweets per minute |
|---|---|---|---|---|---|
| Oráculo API Realtime - Brazil | 195 | 2846 | 5594 | 9524 | 165.0818182 |
| Oráculo API Realtime - USA | 195 | 2845 | 5593 | 9525 | 165.0727273 |
| DMI-TCAT Track - Brazil | 17 | 181 | 329 | 567 | 9.945454545 |

Source: Created by the author

Figure 4: Graph with number of tweets collected in algorithm tests




Source: Created by the author

In this situation there were very different performance results between the Oráculo and DMI-TCAT nodes. The graph in Figure 4 shows that Oráculo nodes in Brazil and in the United States obtained identical results; however, the DMI-TCAT result is lower. Because, in all these nodes, the Twitter API is the basis of these collections, only the algorithms and methods used to explain this difference remain. It should be noted that the programming language adopted by DMI-TCAT differs from the Oráculo framework, as they possess different libraries and features.

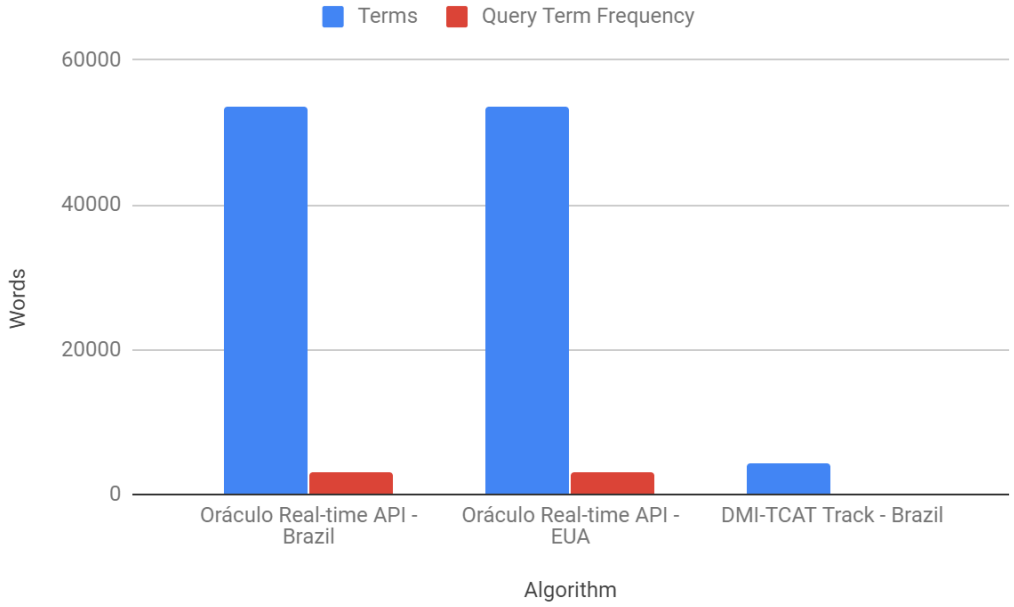
Real-time collections, performed within a 15-minute interval, were also analyzed using the Oráculo and DMI-TCAT data miner. Table 4 and Figure 5 present the results of this analysis:

Table 4: Analysis of the tweets collected in Real-time algorithm tests

| Algorithm | Number of terms | Frequency of consulted term | Frequency (%) |
|---|---|---|---|
| Oráculo Real-time API - Brazil | 53615 | 3100 | 5.78% |
| Oráculo Real-time API - EUA | 53584 | 3098 | 5.78% |
| DMI-TCAT Track - Brazil | 4213 | 195 | 4.63% |

Source: Created by the author

Figure 5: Analysis of Real-time Collection




Source: Created by the author

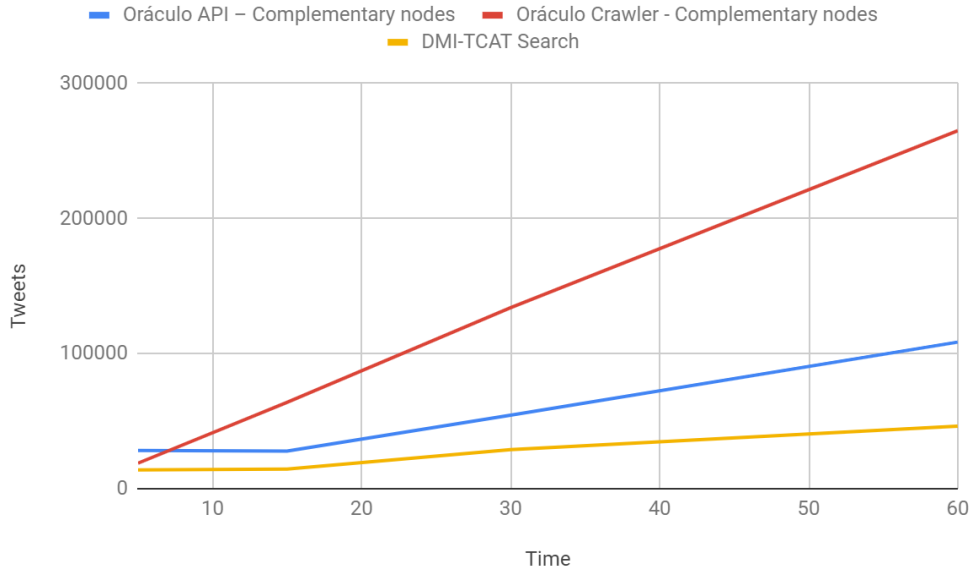
When analyzing the results presented in Table 2, it was found that the superior performance of the Oráculo application framework in the number of collected tweets also implied an incidence of the query term in the analyzed set, although this difference may be related to the difference of the text mining algorithm. Oráculo enables collection distributed across different geographically dispersed nodes. By employing this feature, Oráculo separates the collection into different partitions according to the collection date, and each node assumes a partition to collect. At the end of the collection phase, the system gathers all collected tweets into a larger and thus more comprehensive collection. Although this feature is not available in DMI-TCAT, it has been compared to serve as a reference to Oráculo performance. Table 5 and Figure 6 present the results of this test for two nodes:

Table 5: Tweets collected in tests for distributed historical algorithms

| Algorithm | 5 min | 15 min | 30 min | 60 min | Average tweets per minute |
|---|---|---|---|---|---|
| Oráculo API – Complementary nodes | 28281 | 27805 | 54361 | 108506 | 1990.481818 |
| Oráculo Crawler - Complementary nodes | 18870 | 63885 | 134005 | 265085 | 4380.409091 |
| DMI-TCAT Search | 13881 | 14568 | 28935 | 46224 | 941.8909091 |

Source: Created by the author.

Figure 6: Graph with number of tweets collected in distributed historical algorithms




Source: Created by the author

It is observed that, even when using twice the number of nodes, the result of distributed collection is not necessarily double the collection made by only one node. As the distributed collection is performed using different periods of posted tweets, the number of tweets available for collection varies.

When analyzing Figure 6, it is worth highlighting Crawler's superior API performance, each with two nodes and with Crawler's performance being 120% superior to the Oráculo algorithm using the API.

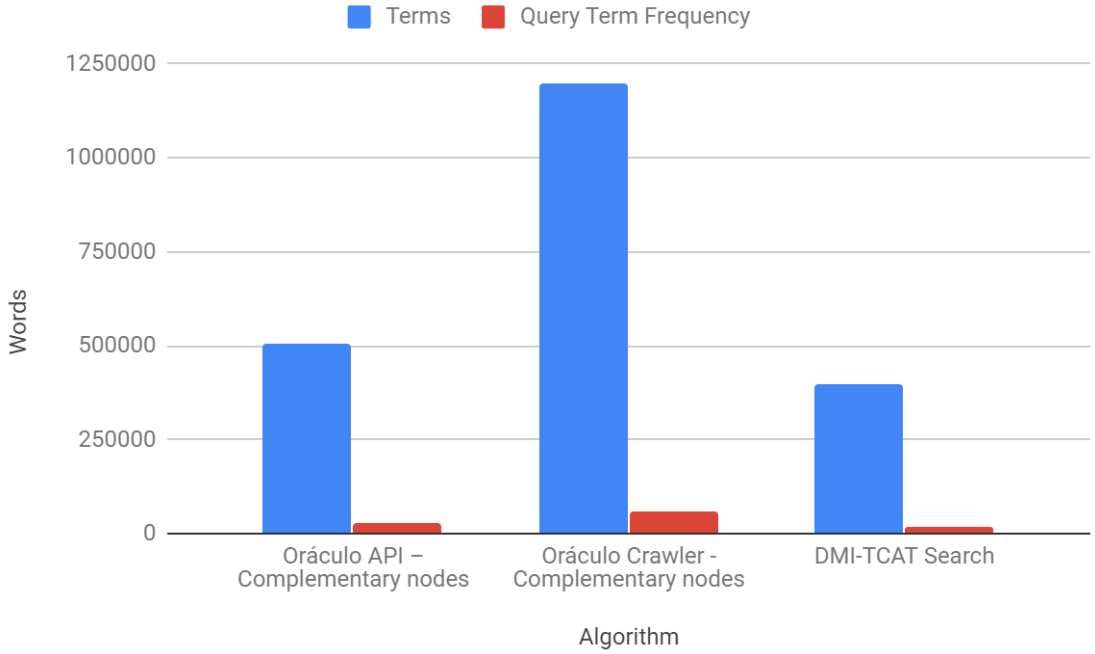
The 15-minute distributed collections with Oráculo and DMI-TCAT as reference were also analyzed with their respective text miners. Table 6 and Figure 7 present the results of this analysis:

Table 6: Analyses of collected tweets in distributed historical algorithm tests

| Algorithm | Number of terms | Frequency of consulted term | Frequency (%) |
|---|---|---|---|
| Oráculo API – Complementary nodes | 505337 | 29374 | 5.81% |
| Oráculo Crawler - Complementary nodes | 1200650 | 56780 | 4.73% |
| DMI-TCAT Search | 395595 | 15268 | 3.86% |

Source: Created by the author

Figure 7: Analysis of distributed historical collections




Source: Created by the author

When analyzing the results of Table 6 and Figure 8, a better result is identified when using Oráculo. Since complementary collection divides the collection into different time frames, thus collecting different tweets among Oráculo algorithms, the difference in incidence can be explained. Because DMI-TCAT was limited to a single node, it did not partition its collection into different dates, and its result reflected this limitation.

## 5 CONCLUSIONS

Although it may appear that Crawler is always the best option for data collection and that API-based algorithms are not interesting, it is necessary to analyze the results achieved from both algorithms. Although both returned the same tweets, and Oráculo output standardizes its output in JSON or MongoDB, the main difference between the two is in the metadata allied to the tweets.

In algorithms using the Twitter API, the return of metadata allied to texts is composed of 34 data, some of them composed. These include data such as geographic location, language and information about the user who posted the tweet. While much of this information is not always present, as it is optional, it may be relevant for different searches.

The data set returned with the Crawler-based algorithm is smaller, as these results are limited to the visual information available in Twitter's search page. Regarding tweets, the following are collected by Crawler: number of retweets, user id, URL, text, user name, date and time of post, medias, tweet ID, number of responses and number of times it was liked. Regarding users, Crawler is able to

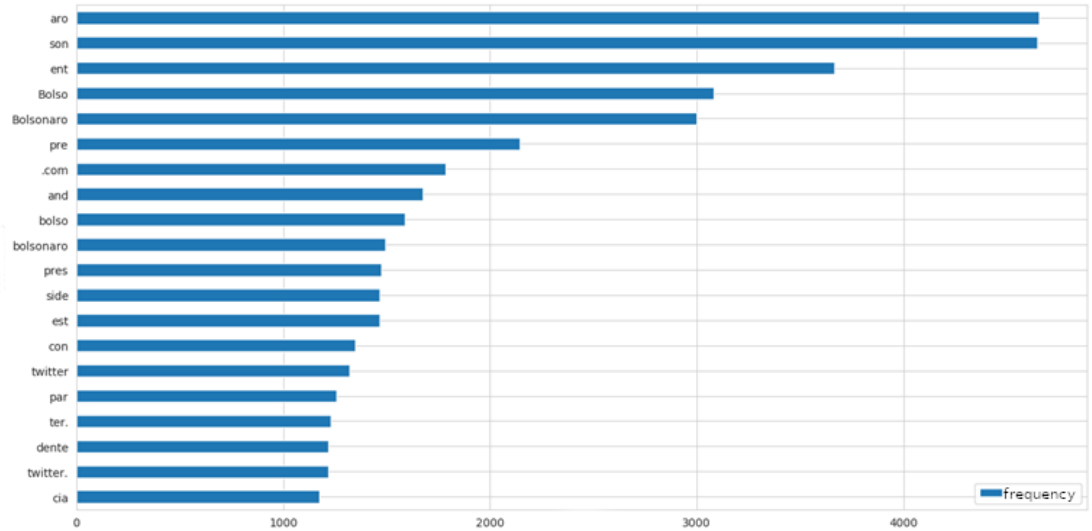
collect the following data: name, user, avatar and ID. Although relevant and always present, these data can be insufficient for conducting certain queries.

As for the ability to collect tweets retroactively, the Crawler-based collection algorithm has been able to return tweets from Twitter's entire historical base since its foundation on March 21, 2006. Therefore, because it is not limited to the last seven days, like the Twitter API-based algorithms are, it offers researchers access to a much larger search base. Thus, there is no single algorithm that is able to meet the needs of all researchers using Twitter as a data source for their research. When using Oráculo, researchers must be aware of the limitations and benefits of each algorithm and select the one that best suits their needs.

When dealing with textual data, one must keep in mind that, in order to understand implicit information quickly and simply, it is necessary to find ways that can represent it to convey some knowledge [25]. Histograms and word clouds can be used in the evaluation process of documents containing unstructured texts to obtain hidden knowledge in the texts [26].

The Oráculo framework relies on text mining tools to analyze the collected texts. These tools include the removal of stopwords, number of terms in the collection, calculation of term frequency, frequency graphs and word cloud about the collection. Figure 8 presents an example of a graph generated by Oráculo with text collection specifically performed for the development of this research.

Figure 8: Graph showing frequency of terms.




Source: Created by the author

Word clouds are commonly used for the qualitative analysis of data. These word clouds are built by using different letter sizes and fonts according to how often the words appear in the text [27]. Figure 9 presents an example of a word cloud generated by the Oráculo from the same text collection.

Figure 9: Word cloud generated by Oráculo


Source: Created by the author

The Oráculo application framework conducted the process of text collection and data mining from Twitter with satisfactory performance. The web crawling and distribution techniques and algorithms obtained results that were superior to those by DMI-TCAT tools as they were able to circumvent the limit in number of requests imposed by the Twitter API.

The results of this research showed that Twitter does not differentiate between the user's geographic location for the collection of tweets, because the collections conducted through the Twitter API in the United States obtained a similar result to the collection made in Brazil.

As a contribution of this research, the Oráculo framework is notable for being available as free software, so that researchers from different fields of knowledge can employ it in their research. With a user-friendly and intuitive web interface, researchers who are not familiar with computer science can more easily collect text and perform text mining on Twitter.

In addition, after comparing results between single-node and distributed collections and evaluating the strengths and weaknesses of each of the available algorithms, future Oráculo framework users are now able to effectively plan their collections.

This research, the quantitative analysis of the data was limited to comparing the performance of the number of tweets collected within a certain period of time between different algorithms available in Oráculo and another tool. For future research, it is possible to conduct a qualitative analysis of the collected texts, thus attesting to the qualitative efficiency of Oráculo algorithms.

The authors intend to continue the development of the Oráculo framework by adding new text mining tools. Once made available in an integrated manner, researchers did not have to resort to other tools and perform data output conversions to perform their analysis of the collected texts. Another future goal is to adapt the use of this framework to other online social networks, thus meeting the need for another range of researchers who conduct their research on these networks.

## REFERENCES

[1] SILVIUS, A. J.; KAVALIAUSKAITE, Ruta. Value of online social networks from the perspective of the user. Journal of International Technology and Information Management, v. 23, n. 2, p. 1, 2014.

[2] RECUERO, Raquel. Contribuições da Análise de Redes Sociais para o estudo das redes sociais na Internet: o caso da hashtag# Tamojuntodilma e# CalaabocaDilma. Fronteiras-estudos midiáticos, v. 16, n. 2, p. 60-77, 2014.

[3] FRANÇA, T.; OLIVEIRA, J.. Análise de sentimento de tweets relacionados aos protestos que ocorreram no Brasil entre Junho e Agosto 2013. Brazilian workshop on social network analysis and mining, 2014.

[4] TAN, Ah-Hwee et al. Text mining: The state of the art and the challenges. In: Proceedings of the PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases. sn, 1999. p. 65-70.

[5] MATHIAK, Brigitte; ECKSTEIN, Silke. Five steps to text mining in biomedical literature. In: Proceedings of the second European workshop on data mining and text mining in bioinformatics. 2004.

[6] AUSSERHOFER, Julian; MAIREDER, Axel. National politics on Twitter: Structures and topics of a networked public sphere. Information, Communication & Society, v. 16, n. 3, p. 291-314, 2013.

[7] TANG, Ying; HEW, Khe Foon. Using Twitter for education: Beneficial or simply a waste of time?. Computers & Education, v. 106, p. 97-118, 2017.

[8] KIM, Yongsung; HWANG, Eenjun; RHO, Seungmin. Twitter news-in-education platform for social, collaborative, and flipped learning. The Journal of Supercomputing, v. 74, n. 8, p. 3564-3582, 2018.

[9] MARADEI, Anelisa. TWITTER COMO ESFERA PÚBLICA EM MOMENTOS DE PROTESTO: Estudo da comunicação pela rede social nos movimentos de 2013, 2015 e 2016 no Brasil. Tese de Doutorado. Universidade Metodista de São Paulo, Universidade da Beira Interior, 2018.

[10] BRUNS, Axel; BURGESS, Jean E. The use of Twitter hashtags in the formation of ad hoc publics. In: Proceedings of the 6th European Consortium for Political Research (ECPR) General Conference 2011. 2011.

[11] RECUERO, Raquel da Cunha; ZAGO, Gabriela da Silva; SOARES, Felipe Bonow. Mídia social e filtros-bolha nas conversações políticas no twitter. Associação Nacional de Programas de Pós-Graduação em Comunicação. Encontro Anual. Anais. São Paulo: Faculdade Cásper Líbero, 2017

[12] GABARDO, Ademir Cristiano et al. Como Mensurar a Importância, Influência e a Relevância de Usuários do Twitter? Uma análise da interação dos candidatos à presidência do Brasil nas eleições de 2018. arXiv preprint arXiv:1902.11197, 2019.

[13] PRUSS, Dasha et al. Zika discourse in the Americas: A multilingual topic analysis of Twitter. In: PloS one, v. 14, n. 5, p. e0216922, 2019.

[14] STEFANIDIS, Anthony et al. Zika in Twitter: temporal variations of locations, actors, and concepts. In: JMIR public health and surveillance, v. 3, n. 2, p. e22, 2017.

[15] BORRA, Erik; RIEDER, Bernhard. Programmed method: developing a toolset for capturing and analyzing tweets. Aslib Journal of Information Management, v. 66, n. 3, p. 262-278, 2014.

[16] BRUNS, Axel; LIANG, Yuxian Eugene. Tools and methods for capturing Twitter data during natural disasters. First Monday, v. 17, n. 4, 2012.

[17] TWITTER. Docs. Disponível: <https://developer.twitter.com/en/docs> Acesso em: 01 jul 2019b.

[18] BENEVENUTO, Fabrício; ALMEIDA, Jussara M.; SILVA, Altigran S. Explorando redes sociais online: Da coleta e análise de grandes bases de dados às aplicações. Porto Alegre: Sociedade Brasileira de Computação, 2011.

[19] MURTHY, Dhiraj. Twitter. Polity Press, 2013.

[20] TWITTER. Q1 2019 Shareholder Letter. Disponível em: <https://s22.q4cdn.com/826641620/files/doc_financials/2019/q1/Q1-2019-Shareholder-Letter.pdf> Acesso em: 01 jul 2019a.

[21] SAEED, Zafar et al. What's Happening Around the World? A Survey and Framework on Event Detection Techniques on Twitter. Journal of Grid Computing, p. 1-34, 2019.

[22] KOUZIS-LOUKAS, Dimitrios. Learning Scrapy. Editora Packt Publishing, 2016.

[23] HERNANDEZ-SUAREZ, Aldo. et al. A Web Scraping Methodology for Bypassing Twitter API Restrictions. arXiv preprint arXiv:1803.09875, 2018.

[24] MORAES, William Bruno. Construindo aplicações com NodeJS. Novatec Editora, 2018.

[25] SARGIANI, Vagner. Identificação de padrões em textos de mídias sociais utilizando redes neurais e visualização de dados. Dissertação de Mestrado. Universidade Presbiteriana Mackenzie, 2018.

[26] BRUNO, G. Text mining and sentiment extraction in central bank documents. In: 2016 IEEE International Conference on Big Data (Big Data) . IEEE, 2016. p. 1700-1708. ISBN 978-1-4673-9005-7. Disponível em: < http://ieeexplore.ieee.org/document/7840784/> Acesso em: 01 jul 2019

[27] VILELA, Rosana Brandão; RIBEIRO, Adenize; BATISTA, Nildo Alves. Os desafios do mestrado profissional em ensino na saúde: uso da nuvem de palavras no apoio à pesquisa qualitative.