Parallel news clustering and topic modeling approaches

Parallel news clustering and topic modeling approaches

A S Shomanov[1] and M E Mansurova[2]

[1] Nazarbayev University
[2] Al-Farabi Kazakh National University

e-mail: adai.shomanov@nu.edu.kz, mansurova.madina@gmail.com

Abstract. At the current age there is an urgent need in developing massively scalable and efficient tools to Big Data processing. Even the smallest companies nowadays inevitably require more and more resources for data processing routines that could enhance decision making and reliably predict and simulate different scenarios. In the current paper we present our combined work on different massively scalable approaches for the task of clustering and topic modeling of the dataset, collected by crawling Kazakhstan news websites. In particular, we propose Apache Spark parallel solutions to news clustering and topic modeling problems and, additionally, we describe results of implementing document clustering using developed partitioned global address space Mapreduce system. In our work we describe our experience in solving these problems and investigate the efficiency and scalability of the proposed solutions.

1. Introduction

News clustering provides a tool to find a grouping of the news documents into a set of relevant categories. These categories, for example, can be further explored and studied in order to extract the most common words and phrases used in these categories. Another popular approach to analyze news documents is a topic modeling. In a topic modeling the words in news documents are assigned a certain probability of belonging to certain topics. In other words, in document clustering the output is a set of clusters with a given documents belonging to a particular cluster centroid, whereas topic modeling finds a set of topics and the list of the most relevant words belonging to each topic. The algorithm that was chosen for the task of classifying news into categories is the K-Means clustering algorithm, which groups data points into a certain number of clusters depending on the location of the point relative to the "centroids". This algorithm was chosen because it works well with unmarked data (such as text), however, it is quite resource-intensive and requires significant computational costs. In this regard, we used an approach using parallel computing. We have chosen the Apache Spark framework based on in-memory computing paradigm that allows to solve clustering problem in parallel. In Apache Spark workload is usually divided between parallel processes that are orchestrated by the master process. The master process governs data distribution, workload balancing and other important administrative tasks in order to keep track of machines and processes running in a cluster environment. K-means is one of the basic unsupervised learning algorithms that gives solution to a clustering problem, given that the number of clusters is approximately known in advance. Therefore, the procedure follows the method of classifying a given data set into a specific fixed number of clusters (k clusters). The basic idea is to determine k centroids, one for each cluster. These centroids must be initially located in the right way, because different locations lead to different results. Thus, the

best choice is to place them as far apart as possible. This idea underlies the Kmeans ++ algorithm [1, 2], which we use for our task of classifying news. The initial step of k-means algorithm is to take each point belonging to a given dataset and associate that point with the nearest centroid. As a next step it is required to recalculate k new centroids as the average of points belonging to a particular cluster obtained as a result of the previous step. After k new centroids have been identified, it is necessary to assign points in the data set to a particular cluster center. This process is carried out iteratively until a certain convergence condition is reached or upon completion of a given number of iteration steps. The K-Means algorithm can thus be used to identify similar categories of objects in complex and unmarked datasets.

2. Dataset and methods

Our dataset was crawled from different online news sources in Kazakhstani section of the internet. This dataset includes 19583 news reports. In the dataset records are stored in a json format. To process such a quantity of data, efficient algorithms and data processing methods are needed. We used the Apache Spark framework for this task, which is based on parallel data processing. To build a K-Means model from the dataset, it is needed to load this dataset into a DataFrame. DataFrame allows to store data in a distributed form across several cluster nodes. Thus, all transformations on the data can be carried out through operations supported by the DataFrame.

The diagram of the data model shows that the data set consists of the following fields:

- URL – url of the data source
- id – identification number
- body – text of the news message
- className – class name
- commentInfo – information about comments left by users
- createDate – date of creation of the news message
- date – date of news release
- lang – the selected language of the news message
- lastChange – date of the last change
- newsSource – name of the news source
- notificationId – news notification identifier
- region – region or area
- tags – relevant tags for the news message
- title – name of the news message
- type – type of news message

A. Data preprocessing

Stop words removal. Stop words are words that, as a rule, should be excluded from input because these words appear frequently and do not carry any meaningful information. The stop word removal procedure was implemented using the MlLib library, which is part of the Apache Spark library set. For example, in the Table I we presented an example of the translated from Russian into English sentence that was converted to an array form of words that do not contain any stop words or punctuation marks.

Table 1. Stop words removal example.

Before stop words removal	After stop words removal
In the coming days, weakening of the ice formation, fluctuations in water levels will be observed on separate rivers in the south and southeast of the republic. In Astana, the daytime is mostly cloudy, -10. In Almaty, it is cloudy in the afternoon, +4.	[coming, days, weakening, ice, formation, fluctuations, water, levels, observed, separate, rivers, south, southeast, republic, Astana, daytime, mostly cloudy, -10, Almaty, daytime, cloudy, + 4]

From this example, one can see that all prepositions and punctuation marks were removed after stop words removal procedure.

TF-IDF procedure. TF-IDF represents term frequency – inverse document frequency measure to assign weights to terms in a collection of documents. Each word receive a score that consists of two parts TF and IDF. TF counts number of times a term appears in a document divided by total number of terms in that document. And IDF is computed as a natural logarithm over total number of documents divided by count of documents containing a specific term for which IDF is measured. As a result of TF-IDF procedure, we obtained news messages converted to a vectorized representation. News message were stored in memory in the form of sparse vectors, where all nonzero values are specified by a pair of index and word itself. This representation is more effective than the uncompressed representation, since basically the text of the message contains much less words than the size of the entire input dictionary.

Stemming. In linguistic morphology and information retrieval, stemming is the process of shortening words to get the stem or root word. We used the Porter algorithm to bring all the words in the news messages to their root form. This process is necessary for the preprocessing of textual information due to the fact that each term can occur in various forms, however, the semantics of this term remains unchanged. For this task, we wrote Apache Spark code for preprocessing news messages to reduce various word forms to their single root word.

3. Results

B. News clustering using Kmeans++ algorithm

After data was preprocessed, we used the Kmeans algorithm for vectorized news messages in a TF-IDF representation. The input for the Kmeans algorithm was the number of clusters k, as well as a distributed DataFrame object that stored all news messages in a distributed form. Using Apache Spark transformations, that corresponds to a set of Mapreduce-like operations, the Kmeans algorithm iteratively selects new cluster centroids, while trying to decrease the intra-cluster distance between the points belonging to the cluster and the centroid of that cluster.

Thus, after a certain number of iterations, the algorithm reaches a certain local minimum. In order to minimize the probability of reaching local minimums and increase the probability of being in a global minimum, we aused a modified Kmeans++ algorithm based on [3, 4]. The approach described in these works is based on the fact that centroids should initially be selected in such a way that their location is as distant as possible from each other. The algorithm randomly selects only the first cluster center. Each subsequent cluster center is selected with a probability proportional to the distance from the previously selected centroids. This algorithm allows one to achieve an O (log K) approximation to the optimal cluster assignment. For example, as a result of clustering, the following news messages were grouped into one group:

- Nazarbayev approved the concept of strengthening and developing of Kazakhstani identity and unity
- The President of the Republic of Kazakhstan approved the Concept for the development of the ANC until 2025
- Article of the head of state “Plan of the Nation – The Path to the Kazakhstan Dream”
- And another category of news reports includes, for example:
- Athlete Victoria Zyabkina: I dream to jump 7 meters!
- The Weightlifting Federation of the Republic of Kazakhstan will not appeal against the decision of the Montreal Laboratory
- Universiade 2017 facilities are being delivered ahead of schedule

C. Topic modeling

We used the LDA [5] (Latent Dirichlet Allocation) approach to select news topics that are most suitable for our dataset. The main idea of the LDA is that each document is represented by a specific

set of topics, and each topic is specific to this document with a given probability. To solve this problem using LDA, we also used the MLLib Apache Spark library, which allows to distribute calculations to parallel tasks according to the Mapreduce-like calculation model, which relies on the scheme of an directed acyclic graph (DAG). Each transformation is a node in this graph, so it is possible to restore the entire computational process from this graph, which also leads to more superior fault tolerance.

The input for the LDA algorithm was the number of topics k, as well as a distributed DataFrame object that stores all news messages in a distributed form. News messages are also stored in a sparse vector representation using TF-IDF representation.

Table 2. Top words for each topic.

Topic 1 words	Topic 2 words	Topic 3 words	Topic 4 words	Topic 5 words
wind	driver	regions	sport	very
in the areas	accident	at times	national team	which
Kazakhstan	assigned	year	Kazakhstan	who
in the afternoon	deputy	city	Olympic	therefore
lighting	graduated	RK	republic	this
mist	toyota	Almaty	from the republic	time

As can be seen from the Table II, the most significant words from the Topic 5 contain mostly stop words that needs to be removed in order to obtain a more relevant topic distribution. It should be noted that words given in the tables are translations from Kazakh and Russian languages into English. In Table III we can see results of topic assignment after stop words removal. It is clear that the selected relevant words are less noisy and show more relevancy and descriptive power towards representing a coherent picture of topic distribution. We used a list of 559 words (for the Russian language), as well as a list of stop words for the Kazakh language, which is part of the nltk (Natural Language Toolkit) library.

Table 3. Top words for each topic.

Topic 1 words	Topic 2 words	Topic 3 words	Topic 4 words	Topic 5 words
republic	Kazakhstan	driver	Kazakhstan	weather
Kazakhstan	areas	accident	governmental	worsening
year	tenge	resulted	work	types
sport	year	happened	new	earthquake
Almaty	development	Almaty	president	movement
region	thousands	police	year	approvement

D. PGAS Mapreduce approach to clustering of documents

We developed a clustering algorithm written in Mapreduce system that was implemented on top of the shared global address space model. PGAS-based Mapreduce [6] has been modified and rewritten to work with arbitrary keys and values. The previous implementation supported only a limited number of data types. For the basic Mapreduce implementation mechanism based on the shared global address space model, a common hashmap data structure was chosen. Hashmap has a number of features that make it suitable for the Mapreduce system. Operations with hashmap are performed with the asymptotic complexity of O (1), which provides quick search of the desired key. In our approach, intermediate key / value pairs are stored in an affine hashmap structure located in the shared part of the

global shared memory (See. Figure 1). Key-value pairs are assigned to reduce threads after the shuffle collectively collects and groups keys according to hashmap structures.

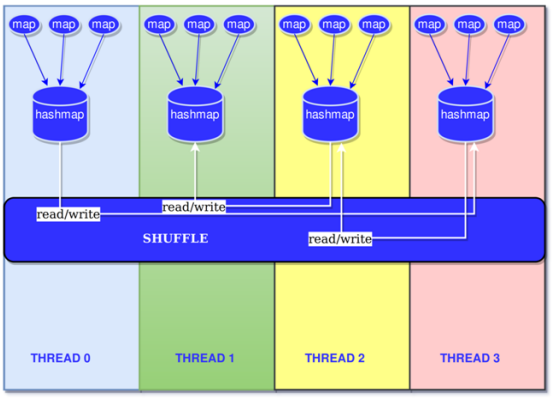

Figure 1. PGAS Mapreduce scheme

The PGAS Mapreduce system relies on efficient execution of bulk and fine-grained operations with memory. In this article, we propose a new implementation of the shuffle procedure, which uses collective operations to exchange key-value pairs between individual threads. The collective exchange process is similar to the collective operation scheme in the MPI environment. Elements stored in hashmap and associated with a particular key are grouped in an orderly manner, so that the values are transferred from one thread to another according to the tree topology (See Figure 2). This approach allows reducing the number of fine-grained memory operations by an order of magnitude. The second modification of our previous PGAS Mapreduce implementation is the addition of support for arbitrary data types for key-value pairs. The implementation is based on the use of void pointers.

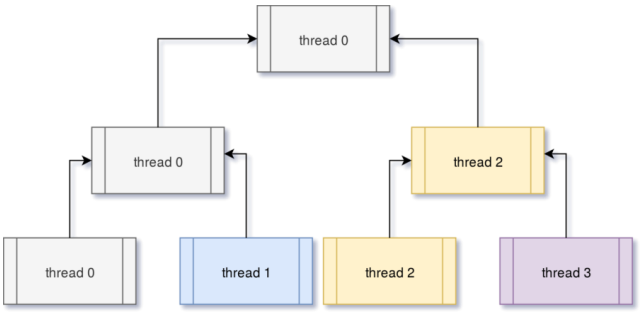

Figure 2. Collective data exchange

The Mapreduce-based parallel clustering algorithm is based on the separation of the workload so that map processes are responsible for assigning objects to certain cluster centers in order to form the corresponding key-value pair consisting of the center of the cluster in combination with the data point. Then, all data points that have been assigned to the same cluster center are grouped at the shuffle stage. The reduce stage is responsible for changing the centers of clusters in accordance with the average sum of all data points in the cluster. The map processes are thus responsible for writing the generated key-value pairs to the local part of the hashmap global structure. After the map processes complete their execution, the shuffle procedure collectively combines all the generated keys to distribute all the values associated with specific keys into the corresponding threads that were selected according to the load balancing algorithm. The reduce processes in this case retrieve data from the global hashmap in accordance with the collective exchange algorithm described above. The parallel algorithm described above was used to cluster the set of documents. The goal is to group similar documents by the description of the documents in the data set. The data set was taken from the UCI

machine learning repository to test our algorithm [7]. The data set consists of 300,000 documents, 102660 unique words and a total of 69679427 words. Each document and word is encoded with a unique identifier. The input data consists of the number of lines represented by the following tuples: document identifier, word identifier, word frequency. Our first step was to convert the above representation of words into a sparse matrix form. For this task, we created a separate Mapreduce task. The result of this Mapreduce pre-processing was a vectorized representation of tuples consisting of a document identifier as a key and a list of related word frequencies. In our experiments, we used a virtual machine with 64 virtual cores and 240 GB of memory from the Google Cloud Platform. The experimental setup consisted of the Berkeley UPC runtime version 2.28.0, the Berkeley UPC-to-C translator, version 2.28.0. Figure 3 depicts scalability of the proposed parallel clustering solution. Results show good scalability. As we can see from the graph, as the number of cores increase, the time to execute the task is proportionately decrease.

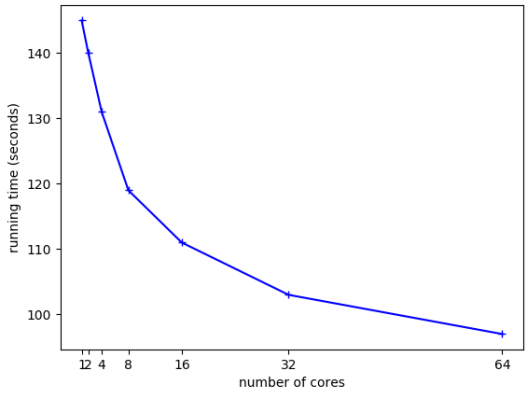

Figure 3. Scalability of the clustering algorithm

4. Conclusion

In this paper different parallel approaches to news clustering and topics modeling were presented. Results showed that developed parallel algorithms possess good scalability and efficiency and can be used to solve the described problems with high quality of obtained solutions.

References

[1] Pasi Fränti Sami Sieranoja 2019 How much can k-means be improved by using better initialization and repeats? Pattern Recognition **93** pp 95–112.

[2] Jordan Yoder & Carey E. Priebe 2017 Semi-supervised k-means++ *Journal of Statistical Computation and Simulation* **87:13** 2597–2608 DOI: 10.1080/00949655.2017.1327588.

[3] Arthur D and Vassilvitskii S 2007 k-means++: The advantages of careful seeding *In Proceedings: ACM-SIAM SODA* (*Symposium on Discrete Algorithms*) pp 1027–1035

[4] Bahmani B, Moseley B, Vattani A, Kumar R and Vassilvitskii S 2012 Scalable k-means++ *Proc. VLDB Endow.* **5** (7) pp 622–633 DOI=http://dx.doi.org/10.14778/2180912.2180915

[5] Blei D M, Ng AY and Jordan M I 2003 Latent dirichlet allocation *J. Mach. Learn. Res*. **3** pp 993–1022

[6] Shomanov A, Akhmed-Zaki D and Mansurova M 2017 PGAS Approach to Implement Mapreduce Framework Based on UPC Language *Parallel Computing Technologies. PaCT 2017. Lecture Notes in Computer Science*, vol 10421 Springer, Cham pp 342–350

[7] Dua D and Graff C 2019 UCI Machine Learning Repository [http://archive.ics.uci.edu/ml] (Irvine, CA: University of California, School of Information and Computer Science)