The Usage of Apache Spark for Collection and Analysis of Social Networking Statistics

Ihor Tovpinets
The Department of System Design
Faculty of Electronics and Computer Technologies
Ivan Franko National University of Lviv
Lviv, Ukraine
itovpinets@gmail.com

Roksolana Kovtko
The Department of System Design
Faculty of Electronics and Computer Technologies
Ivan Franko National University of Lviv
Lviv, Ukraine
rsulymko@ukr.net

Volodymyr Yuzevych
The Department of System Design
Faculty of Electronics and Computer Technologies
Ivan Franko National University of Lviv
Karpenko Physico-mechanical Institute of the NAS of Ukraine
Lviv, Ukraine
yuzevych@ukr.net

Andrii Prodyvus
The Department of System Design
Faculty of Electronics and Computer Technologies
Ivan Franko National University of Lviv
Lviv, Ukraine
andrijy.prodyvus@lnu.edu.ua

***Abstract*—This paper investigates the new Facebook API for Instagram. During the work we implemented the live data stream from the Instagram social network and displayed the hashtags depending on their popularity at a specific moment or during some period. We also investigated the relationship between hashtags, depending on the time of data collection and analysis. As a result of the investigation, we developed grouping of related hashtags (based on the given hashtag), which are not placed directly in the same post but connected through the other posts with the intermediate hashtags, which also belong to a certain semantic group.**

Keywords—Scala, Spark, Big Data, life-streaming, streaming, spark-streaming, Social network, Graph API

I. INTRODUCTION

The processing and analysis of a large amount of data generated on the Internet become strategic tasks, in the conditions of market competition intensification and information space oversaturation. Business analysts can successfully adjust the strategy of entering a business, due to the discovery of patterns between the data collected through digital marketing. Currently, there is a transition from the usual marketing segmentation, which is based on selective, not always objective data, to customer service, that depends on the customer's requirements.

Big Data technologies allow you to convert and analyze a lot of heterogeneous and unstructured data. These technologies are only gaining popularity in the Ukrainian business environment due to the high cost of software and high requirements for skilled professionals, but the number of companies that implement them is growing steadily.

The relevance of the work is determined by the increasing load on the data processing systems. Scalable and distributed systems are used to solve this class of problems. Apache Spark is one of the tools to build such systems. It allows receiving and working with real-time data, which highly affects the decision-making dynamics depending on the analyzed data changes. Social networks are quite popular, and therefore, data collection and real-time analysis can be considered reliable and relevant to modern social problems and requirements in various spheres.

The purpose of this work is to create a system that provides the ability to collect and analyze the data of the Instagram social network, explore the relation between hashtags based on live user publications.

II. REVIEW OF TECHNOLOGIES

Many products are designed to build distributed real-time systems nowadays. One of the most known open-source products is Apache Spark Streaming.

All available systems can be divided into 3 major classes [1]:

1. Application software systems designed to work on only one personal computer or workstation. These include word processors, spreadsheets, graphics systems, etc.
2. Built-in systems designed to work on one processor or an integrated processor group. These include household control systems, various devices, and devices for technical use.
3. Distributed systems in which the software is installed on a group of parallel working processors, connected through the network. These include publishing systems, systems of collective use.

In distributed computing we can distinguish 3 strategies for data management [2]:

1. centralized;
2. replicated;
3. distributed.

Apache Spark is a universal and high-performance clustered computing platform [3]. The framework was created to cover the widest range of workloads that first required the creation of separate distributed systems, including batch processing applications, cyclic algorithms, interactive queries, and thread processing.

The elastic distributed data sets (hereinafter referred to as RDDs) represents a set of data, that performs fast and easy access to application interface data on Python, Scala, and Java.

By default, each transformed RDD can be recalculated every time a new action is performed on it. However, RDD can also be stored for a long time in memory. Using this method of storing or caching, Spark will store the necessary elements on the cluster (Fig. 1) and the user will be able to perform requests much faster.

978-1-7281-3561-8/19/$31.00 ©2019 IEEE

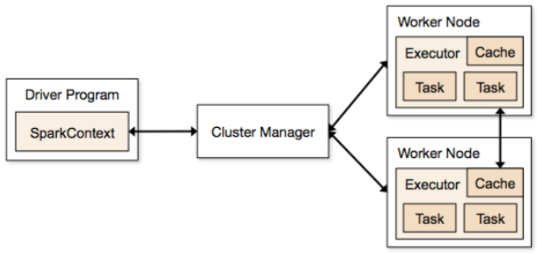


Fig. 1. The Apache Spark cluster architecture

The main reasons for the importance of the Apache Spark framework:

• the speed at all levels of the frame;

• multifunctionality;

• frameMark is a modification of the MapReduce computing model.

The framework's flexibility is that programs can be run under the control of various distributed systems [4]:

• Stand-alone mode, where the user can deploy Spark infrastructure and manage all cluster resources and tasks.

• Yarn is a computing platform that is part of the Hadoop system. In this case, the program runs on the Hadoop cluster, running this platform.

• Mesos is an alternative cluster resource management system.

• Local mode created for development and debugging.

Apache Spark and Hadoop are open-source cluster computing platforms designed to solve various problems. Hadoop provides a distributed data infrastructure, where all data is stored in many nodes inside the cluster. This means that the user of this platform should not worry about buying expensive equipment. Freemark Hadoop also indexes and tracks data that facilitates processing and analytics as well as automates these processes. On the other hand, Apache Spark operates on such distributed data sets but does not require the creation of a distributed data warehouse.

Hadoop is a whole system that includes not only a data storage component but also a distributed cluster for data processing - YARN, Map Reduce. Processing data using Map Reduce technology is acceptable for the majority of static data sets, but the analysis of data obtained in real-time is not provided here, for such operations Apache Spark is better suited. In batch processing data task, the advantage of this framework over Hadoop reaches 10 times, and when processing data that is in memory - 100 times [6].

Hadoop is more resistant to various types of crashes since after each operation data is distributed, recorded and copied between a specific number of nodes. Each file is split into blocks and can be copied to different machines, which in turn provides reliability. In case of falling off one of the nodes, data is restored from another. Spark can use HDFS, so the stability will be provided by Hadoop tools.

III. Project For Collection And Analysis Data From The Social Network Development Process

Before you will write the project code directly, you need to configure all the necessary components. The development was done using such technology set:

• JDK (Java Development Kit) - a set of libraries and tools for writing and compiling Java code.

• Intellij IDEA - a development environment that integrates the necessary development components

• SBT is an open-source programming tool for projects that are using Scala and Java programming languages. This tool is similar to Maven or Gradle.

The specified tools are installed from the official repositories.

To access the latest version of Apache Spark, you need to download it from the official git repository https://github.com/apache/spark. After that, you need to execute the corresponding Maven command, to collect the source code of the framework.

Once all the necessary steps are taken to prepare the work environment, the required libraries and tools become available in an integrated development environment.

Additionally, you must configure the environment variables by specifying the variables: HADOOP_HOME, SBT_HOME with a link to the corresponding folders. Properly for Instagram, you need to specify INSTAGRAM_AUTH_TOKEN and INSTAGRAM_BU_ID that are used directly in the project itself.

Here INSTAGRAM_AUTH_TOKEN is the authorization token added to each request accordingly to the Facebook requirements for the Instagram API;

INSTAGRAM_BU_ID is the Business Profile user ID that is added to each request accordingly to the Facebook requirements for the Instagram API.

At first, you need to declare an authorization class, since each request to the Graph Facebook API (hereinafter - API) requires authorization.

```
case class InstagramAuth (accessToken: String, apiHost: String = "graph.facebook.com")
```

accessToken is the authorization token that is read from the environment variables that were specified earlier.

```
accessToken = System.getenv ("INSTAGRAM AUTH TOKEN")
```

Implementation of the Instagram Client class, which is intended to provide basic search configurations.

```
    abstract class InstagramClient(auth: InstagramAuth)
extends Serializable with Logger {
    def loadNewInstagrams():
Iterable[RecentMediaItem] = {
        try {
          loadNewInstagramsPaginated()
        } catch {
          ..} }
```

The loadNewInstagrams and loadNewInstagramsPaginated methods carry out a simple search and search with depth degeneracy given by the DEPTH_SEARCH variable, respectively.

The actual class that will be used as a tag search client.

```
    class InstagramTagClient(tag: String, userId:String,
auth: InstagramAuth) extends InstagramClient(auth) {

     override protected def fetchInstagramResponse(url:
Option[String] = None): String = { val fetch =
url.getOrElse(s"https://${auth.apiHost}/$tag/recent_med
ia?user_id=$userId&access_token=${auth.accessToken}
&fields=id,media_type,comments_count,like_count,capti
on,media_url,premalink")
```

Here, fetchInstagramResponse is an overridden method from InstagramClient, which describes how to get data, namely the URL of the API with the necessary data and parameters.

Next, you need to create a class that is derived from PollingReceiver with a given data type, in our case it is RecentMediaItem.

```
    class InstagramReceiver(
     …
    )extends PollingReceiver [RecentMediaItem]
(pollingSchedule, pollingWorkers, storageLevel) with
Logger {
      @volatile private var savedLastId = "-1"
      @volatile private var newId = "-1"
      override protected def poll(): Unit = {
        var reachedLast = false
        client
          .loadNewInstagrams()
          .filter(x => {
            reachedLast = reachedLast ||
x.id.equals(savedLastId)
            !reachedLast})
          .foreach(x => {
            if (newId.equals("")) {newId = x.id}
            store(x)
          })savedLastId = newId
      }}
```

InstagramReceiver is the class with an implemented poll method for filtering and storing received data groups in memory.

Let's consider the functions used to count the number of tags, and the number of pairs.

```
    val tagWords = rddMedia.flatMap(x =>
x.caption.getOrElse("").split("[.]|[\\s]"))
          .filter(_.startsWith("#"))
          .map(x => (x, 1))
          .reduceByKey(_ + _)
    var tagsPairs = rddMedia.flatMap(x =>
x.caption.getOrElse("")
                  .split("[.]|[\\s]")
                  .filter(_.startsWith("#"))
                  .combinations(2))
          .map(x => {var x1 = x(1); var x2 = x(0)
            if (x(0).compareTo(x(1)) > 0) {x1 = x(0); x2 =
x(1)}
            ((x1 , x2), 1)
          })
          .reduceByKey(_ + _)
```

In both cases, the split and filtering functions are used to separate tags from the user's post. The flatMap function allows you to work with unstructured data collections and aggregate this data into the original collection. We use combinations to get all possible pairs of hashtags in a post. The map function transforms data into key-value object. We can aggregate the data by a certain parameter, using the reduceByKey function.

IV. Demonstration Of The Project

For the correct execution, you must first configure the environment variables (INSTAGRAM_ AUTH_ TOKEN and INSTAGRAM_BU_ID), then:

1) Enter the hashtag for monitoring and data analysis. For example: tag = "sun" (Fig. 2).

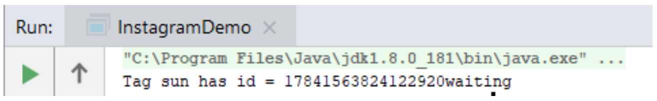
2) Specify the time characteristics that determine the project execution:

• timeForPooling (ms), the frequency with which the data is collected in real-time.

• numOfIterations, the number of iterations that determines the total duration of the monitoring (may be set in seconds).

• rddIterdelta (ms), the time of calculation of received data, the duration must not exceed timeForPooling * (numOfIterations-1).

3) Run the project …

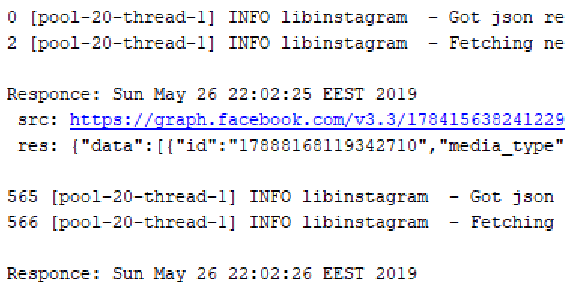


```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Tag sun has id = 17841563824122920waiting
```

Fig 2. Project starting message. «sun» as a hashtag with his id.

At this point, the project starts to automatically track all Instagram posts that contain the specified hashtag in real-time.

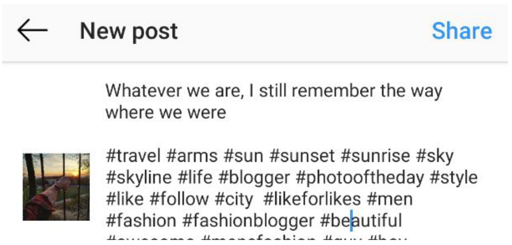
Dynamically, we receive messages about data appeared on the network, that is so-called LifeStream - a stream of data that arrives in real-time (Fig. 3).



```
0 [pool-20-thread-1] INFO libinstagram  - Got json re
2 [pool-20-thread-1] INFO libinstagram  - Fetching ne

Responce: Sun May 26 22:02:25 EEST 2019
 src: https://graph.facebook.com/v3.3/178415638241229
 res: {"data":[{"id":"17888168119342710","media_type"

565 [pool-20-thread-1] INFO libinstagram  - Got json
566 [pool-20-thread-1] INFO libinstagram  - Fetching

Responce: Sun May 26 22:02:26 EEST 2019
```

Fig. 3. LifeStream with the hashtag data

During monitoring, we will post to the Instagram social network to verify the accuracy of real-time data (Fig. 4).



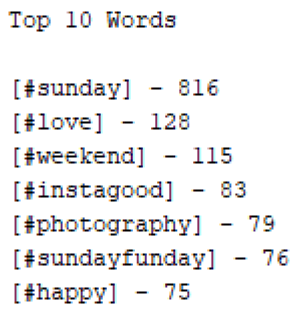
Whatever we are, I still remember the way where we were

#travel #arms #sun #sunset #sunrise #sky #skyline #life #blogger #photooftheday #style #like #follow #city #likeforlikes #men #fashion #fashionblogger #beautiful

Fig. 4. The publication in the social network Instagram

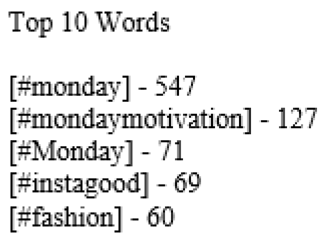
In the next iteration, LifeStream will be able to see a post that is relevant to our publication.

Consider the #today tag. The parameter rddIterdelta is configured as timeForPooling * numOfIterations - 1. We will get aggregated data of the hashtag's appearance frequency (one of which is the #today tag). Please note that the screen entries and execution results that are shown in Fig 5.1 were taken on May 26, 2019 (Sunday), and on Fig. 5.2, the results of the workshops on May 27, 2019 (Monday).



```
Top 10 Words

[#sunday] - 816
[#love] - 128
[#weekend] - 115
[#instagood] - 83
[#photography] - 79
[#sundayfunday] - 76
[#happy] - 75
```

Fig. 5.1. Results for the hashtag #today on Sunday

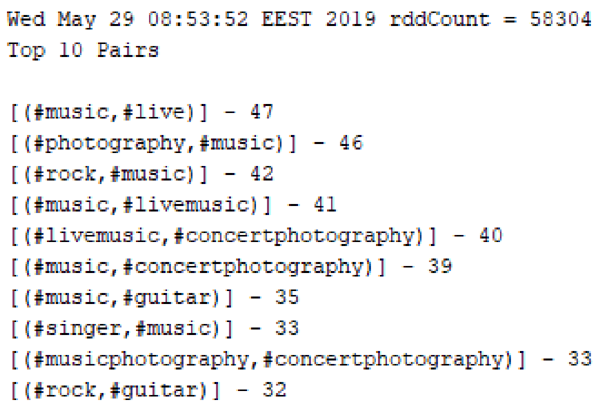


```
Top 10 Words

[#monday] - 547
[#mondaymotivation] - 127
[#Monday] - 71
[#instagood] - 69
[#fashion] - 60
```

Fig. 5.2. Results for the hashtag #today on Monday.

After analyzing received data, we saw that during the process of data collection and analysis (on Sunday), in combination with the hashtag #today, often there were hashtags #Sunday #weekend #love and so on. The next time the project was launched on Monday, May 27, 2019, the results were received as expected. These words are lexical. This project allows the program level to "understand" that these sets of letters are truly homogeneous concepts at a certain level.

Additionally, we implement an algorithm in the project, which allows obtaining the connection tree of related hashtags in the deep. That is, in addition to finding pairs for the main hashtag, implemented the search for kinship by secondary hashtags, appeared in pairs with the given ones. This allows you to monitor the emergence of new hashtags that have never been seen before but belong to the tracked topic. It is also possible to add the conditions for filtering the resulting queries and look at the related hashtags, for example to the #concert hashtag.



```
Wed May 29 08:53:52 EEST 2019 rddCount = 58304
Top 10 Pairs

[(#music,#live)] - 47
[(#photography,#music)] - 46
[(#rock,#music)] - 42
[(#music,#livemusic)] - 41
[(#livemusic,#concertphotography)] - 40
[(#music,#concertphotography)] - 39
[(#music,#guitar)] - 35
[(#singer,#music)] - 33
[(#musicphotography,#concertphotography)] - 33
[(#rock,#guitar)] - 32
```

Fig. 6. The couples related to the #concert

The hashtags (Fig. 6) related to the preconfigured (#concert) and have a close semantic value are represented by the frequency of occurrence:

- #music and #life
- #rock and #music
- #livemusic and #concertphotography.

This, in turn, will allow further separation and clustering similar hashtags, even if they did not appear in any post with the given one.

V. Conclusions

We analyze the usage of the Apache Spark framework for solving distributed data computing tasks and their analysis in real-time. Investigate the approaches and main application fields of the framework, and consider the typical problems solved by Apache Spark. Describe the main components of the framework: Streaming, Spark SQL.

The conclusion of the detailed analysis of Apache Spark and Hadoop is that these frameworks are not competitive for working with large volumes of data. Both frameworks are the main Apache products for working with large volumes of data and are usually used together. But you must understand the basic difference before using them: Hadoop is tied to the hard drive using the MapReduce paradigm, while Spark is more flexible but requires much more RAM.

This article describes the process of setting up a development environment. Demonstrates in detail the main features of the Spark framework for processing and analyzing the data received from the social network Instagram in real-time.

Consequently, as a result of the development application software, the following tasks were performed:

- Explored new API from Facebook for Instagram;
- Provided access from the project to the private part of the API;
- Created a stream of data from the social network Instagram in real-time;
- Implementing of displaying hashtags depending on their popularity at a specific moment or period is foreseen;
- Investigated the relationship between hashtags (considering the hashtag under which the monitoring is conducted), depending on the time of data collection and analysis;
- Implemented grouping of related hashtags (based on a given hashtag) that is not found directly in the same post but connected through a link of intermediate hashtags, which also belong to a certain semantic group.

The results of the project can be used both in marketing (to search for popular and related groups of hashtags, as well as hashtags that are gaining popularity in a certain area), but also for computer systems that is training to identify groups of words that have a common semantics that has a fairly wide application in the field of artificial intelligence.

References

[1] G. Agha. Actors: A Model of Concurrent Computation in Distributed Systems. Cambridge: MIT Press, 1985.

[2] S. Gulati, S. Kumar, Apache Spark 2.x for Java Developers. Mumbai: Packt Publishing, 2017.

[3] S. Ryza, U. Laserson, S. Owen and J. Wills, Advanced Analytics with Spark. Boston: O'Reilly, 2016.

[4] P. Zecevic, M. Bonaci, Mastering Apache Spark. London: Manning, 2015.

[5] The Scala Programming Language. [Online]. Available: https://www.scala-lang.org/

[6] Hadoop vs. Spark: The New Age of Big Data. [Online]. Available: https://www.datamation.com/data-center/hadoop-vs.-spark-the-new-age-of-big-data.html