ISSN 2347 - 3983
Volume 6, No.1 January 2018
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter01612018.pdf
https://doi.org/10.30534/ijeter/2018/01612018

Real-time Streaming Data Analysis using Spark

Kyeongjoo Kim[1], Jihyun Song[2], Minsoo Lee[3]
[1]Dept. of Computer Science and Engineering, Ewha Womans University, Seoul, Korea,
Email:kjkimkr@ewhain.net
[2] Dept. of Computer Science and Engineering, Ewha Womans University, Seoul, Korea,
Email:ssongji7583@ewhain.net
[3] Dept. of Computer Science and Engineering, Ewha Womans University, Seoul, Korea,
Email:mlee@ewha.ac.kr

ABSTRACT

Streaming data generates a lot of information in real-time. Various types of streaming data exist, and analyzing such streaming data in real-time is a very important issue. Streaming data such as IoT sensor data, SNS Twitter data, stock data could contain very sensitive information that need to be analyzed in real-time. Our approach to analyze streaming data is based on real-time twitter data. We used Spark streaming provided by Apache Spark API for processing streaming data. We analyzed hashtags of real time twitter streaming data to find some related information about specific interesting keywords. In this paper, we did two kinds of analysis. First, searching the Top 10 hashtags related to a keyword. Second, analyzing how many words used related with happiness represented as integer values.

Key words: Hashtag, Sentimental analysis, Spark streaming, Twitter

1. INTRODUCTION

TWITTER is one of the biggest platform in Social network systems (SNS). And their instant messages (i.e. tweets) are published every second. Users tend to express their real feelings freely in Twitter, which makes it an ideal source for capturing the opinions towards various interesting topics, such as brands, products or celebrities, etc. Hashtags, starting with a symbol “#” ahead of keywords or phrases that users want to emphasize are widely used in tweets. In this paper, we did hashtags analysis based on real time Twitter data using Apache Spark streaming. Specifically we used the iphone8 keyword as it is a popular keyword. We analyzed hashtags about iphone8 in two ways. The first way is analyzing Top 10 hashtags represented with “#iphone8” and another way is how related the tweet with iphone8 hashtag is to happiness using the happiness word table.

2. RELATED RESEARCH

2.1 Spark streaming

Spark Streaming enables scalable, high-throughput, fault-tolerant stream processing of live data streams. It is an extension of the core Spark API and can be used to stream live data and perform real-time processing. Spark Streaming provides a single platform for ingesting data in order to perform fast and live processing in Apache Spark [1].

Data Streaming is a technique to transfer data so that it can be processed as a steady and continuous stream. Streaming technologies are becoming increasingly important and we can use Spark Streaming to stream real-time data from various sources like Twitter, Stock Markets, IoT sensors and Geographical Systems and perform powerful analytics to help businesses. The fundamental stream unit is DStream which is basically a series of RDDs to process the real-time data. The major features of Spark Streaming are scaling, speed, fault tolerance, integration, and business analysis [2].

Spark Streaming workflow has four high-level stages. The first is to stream data for real-time streaming from various sources like Kafka, Flume or Parquet. The second type of source is for static/batch streaming which includes HBase, MySQL, PostgreSQL, Mongo DB and Cassandra. Afterwards Spark can be used to perform Machine Learning on the data via its MLlib API[3]. Further, Spark SQL can be used to perform further operations on this data. Finally, the streaming output can be stored into various data storage systems such as HBase, Cassandra, Kafka, HDFS and local file systems. Spark Streaming Workflow contains four components, Streaming Context, DStream, Caching, and accumulators, broadcast variables and checkpoints [4]. Streaming Context consumes a stream of data in Spark and is the main entry point for Spark functionality. Discretized Stream (DStream) is the basic abstraction provided by Spark Streaming. Accumulators are variables that are only added through an associative and commutative operation and are used to implement counters or sums. Broadcast variables allow the programmer to keep a read-only variable cached on each machine rather than shipping a copy of it with tasks, which makes it useful to give a copy of a large input dataset in an efficient way. .

2.2 Twitter analysis

Text mining uses natural language processing techniques and analytical methods on text data in order to derive useful information. Text mining is gaining a lot of attention these last years, due to an exponential increase in social media services such as Twitter[5]. Twitter data constitutes a rich source that can be used for capturing information about various topics. This data can be used in to find trends, measure brand sentiment, and gather feedback.

In this paper, we chose Twitter tweet as streaming text. We need an authentication framework to get twitter streaming data. To create services which act on behalf of users' accounts and make it really secure and easy to develop, we need three things, which are the Twitter application, REST API and access to the user account.

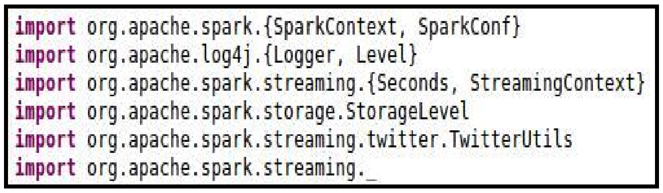
To put the pieces together into a working mechanism, we need an authentication framework. As a Twitter standard, the REST API identifies Twitter applications and users using OAuth. OAuth is an open protocol to allow secure authorization in a simple and standard method from web, mobile and desktop applications. This framework allows users to grant you permission to act on their behalf without sharing the account password. After the user has given permission, OAuth will return you a token, and this token itself grants access to make requests on behalf of the user [6].

3. APPROACH FOR SPARK STREAMING DATA ANALYSIS

In this part, we explain our real-time streaming data analysis approach used in detail. First, we need to add some external jar files to develop our model. Our model is based on real time streaming data. So we need to connect http network. Okhttp3 and Gson jar files consider this problem. And Twitter4j jar file make it possible to load Twitter data and use their function$_{x}$.

3.1 Spark streaming implementation

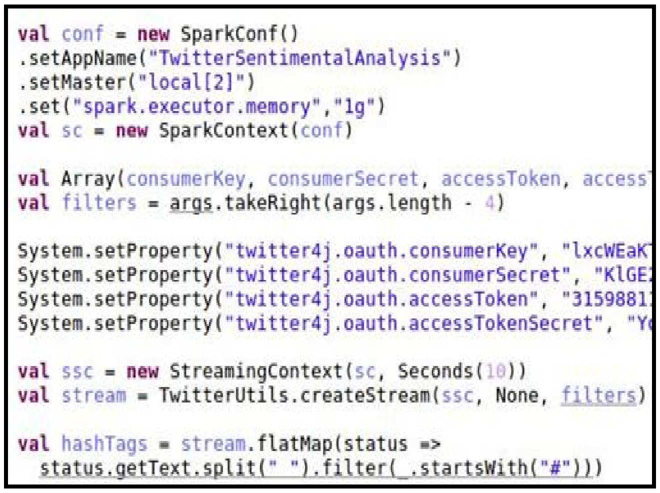
Figure 1 represents 'import' part of whole codes to make it executed. Spark Context, SparkConf is necessary to execute Spark to get host address and access to the Spark. Logger and Level is determined log level of the code. And the rest of this park is about Spark streaming and Twitter streaming data. It has a package called Twitter.utils, which contains of all the built-in functions to stream data from Twitter



```
import org.apache.spark.{SparkContext, SparkConf}
import org.apache.log4j.{Logger, Level}
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.storage.StorageLevel
import org.apache.spark.streaming.twitter.TwitterUtils
import org.apache.spark.streaming._
```

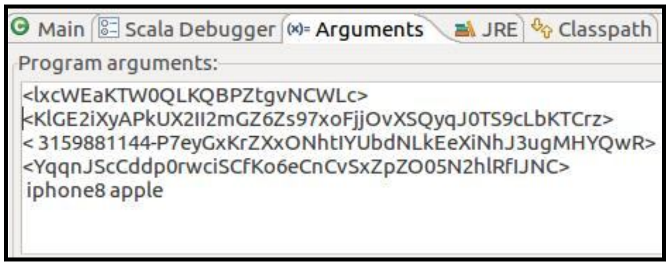
Figure 1: Import part of Scala code using spark streaming API and Twitter API.

In main function (Figure 2), there should be connection to spark context and stream context. It also gets the filter words from arguments and split by blank and attach a symbol "#" ahead of the word. This program will gain the Twitter authentication key and filter words from arguments when it needs to be entered in the run stage (Figure 3). In this paper, we use 'iphone8' as a filter word and then, our program loads streaming data every 10 seconds.



```
val conf = new SparkConf()
.setAppName("TwitterSentimentalAnalysis")
.setMaster("local[2]")
.set("spark.executor.memory","1g")
val sc = new SparkContext(conf)

val Array(consumerKey, consumerSecret, accessToken, access
val filters = args.takeRight(args.length - 4)

System.setProperty("twitter4j.oauth.consumerKey", "lxcWEaK
System.setProperty("twitter4j.oauth.consumerSecret", "KlGE
System.setProperty("twitter4j.oauth.accessToken", "3159881
System.setProperty("twitter4j.oauth.accessTokenSecret", "Y

val ssc = new StreamingContext(sc, Seconds(10))
val stream = TwitterUtils.createStream(ssc, None, filters)

val hashTags = stream.flatMap(status =>
  status.getText.split(" ").filter(_.startsWith("#")))
```

Figure. 2: Code to make connection and get filter words.



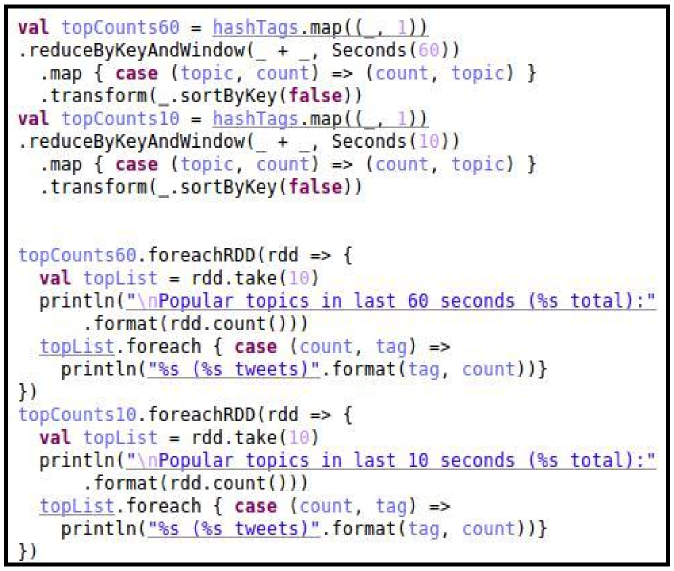
```
Program arguments:
<lxcWEaKTW0QLKQBPZtgvNCWLc>
<KlGE2iXyAPkUX2Il2mGZ6Zs97xoFjjOvXSQyqJ0TS9cLbKTCrz>
< 3159881144-P7eyGxKrZXxONhtIYUbdNLkEeXiNhJ3ugMHYQwR>
<YqqnJScCddp0rwciSCfKo6eCnCvSxZpZO05N2hlRfIJNC>
 iphone8 apple
```

Figure 3: Arguments for consumer key, consumer secret, access token, access token secret, and filter words.

After that, the following figures show the main part of our approach. The first case is about searching other hashtags written with our filter hashtag (ex. #iphone8) (Figure 4). The other case is about analyzing happiness in tweets with our filter hashtag. The happiness is based on AFINN happiness table (Figure 7).

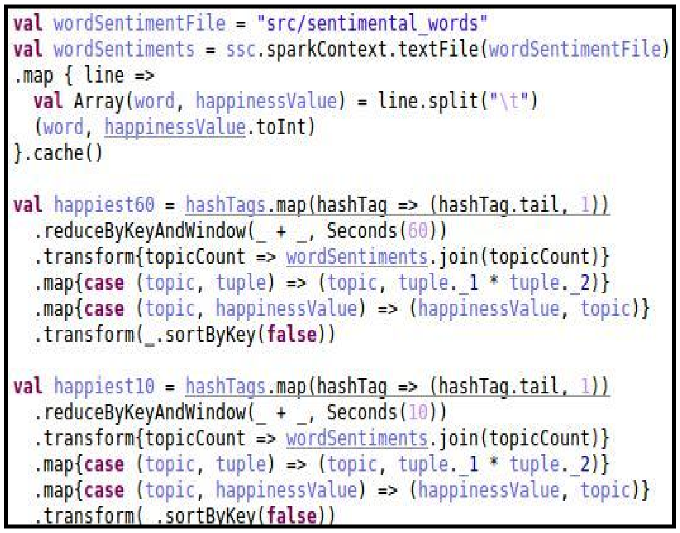
In the first case, there are two variables in Top 10 part. One is searching hashtags by 60 seconds and the other is by 10 seconds. That functions show the list of popular hashtags used with "#iphone8". The code basically does a word count of the hashtags over the previous 60/10 secs as specified in the ‘reduceByKeyAndWindow’ and sorts them in descending order. ‘ReduceByKeyAndWindow’ is used in case we have to apply transformations on data that is accumulated in the previous stream intervals.

In the second case, it is about measuring happiness of the filter word. In this part, we used happiness table as a criteria. It was split by a tab because word-integer value pairs separated by a tab and stored in a cache to reduce I/O overhead. After that, we made two variables: happiest60 and happiest10. The happiest60 loads Twitter streaming data every 60 seconds and the happiest10 make it every 10 seconds as RDD data. Those functions found tweets written with our filter hashtag and calculate the happiness value based on “sentimental_words” table. (Figure 5) After calculating, it shows results every 60 and 10 seconds. (Figure 6).



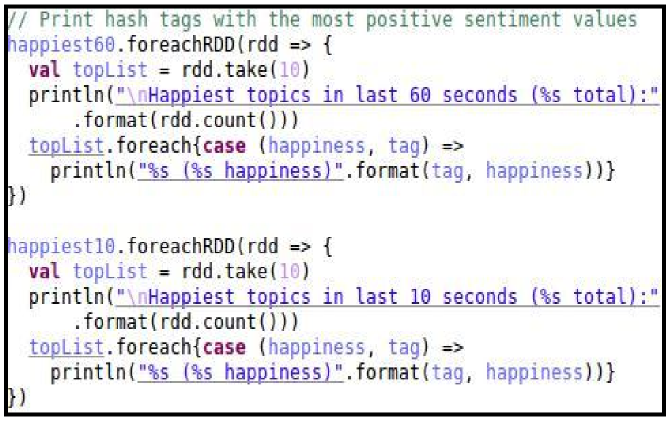
```
val topCounts60 = hashTags.map((_, 1))
.reduceByKeyAndWindow(_ + _, Seconds(60))
  .map { case (topic, count) => (count, topic) }
  .transform(_.sortByKey(false))
val topCounts10 = hashTags.map((_, 1))
.reduceByKeyAndWindow(_ + _, Seconds(10))
  .map { case (topic, count) => (count, topic) }
  .transform(_.sortByKey(false))


topCounts60.foreachRDD(rdd => {
  val topList = rdd.take(10)
  println("\nPopular topics in last 60 seconds (%s total):"
      .format(rdd.count()))
  topList.foreach { case (count, tag) =>
    println("%s (%s tweets)".format(tag, count))}
})
topCounts10.foreachRDD(rdd => {
  val topList = rdd.take(10)
  println("\nPopular topics in last 10 seconds (%s total):"
      .format(rdd.count()))
  topList.foreach { case (count, tag) =>
    println("%s (%s tweets)".format(tag, count))}
})
```

Figure 4: Search popular hashtags used with filter hashtag every 60 and 10 seconds.



```
val wordSentimentFile = "src/sentimental_words"
val wordSentiments = ssc.sparkContext.textFile(wordSentimentFile)
.map { line =>
  val Array(word, happinessValue) = line.split("\t")
  (word, happinessValue.toInt)
}.cache()

val happiest60 = hashTags.map(hashTag => (hashTag.tail, 1))
  .reduceByKeyAndWindow(_ + _, Seconds(60))
  .transform{topicCount => wordSentiments.join(topicCount)}
  .map{case (topic, tuple) => (topic, tuple._1 * tuple._2)}
  .map{case (topic, happinessValue) => (happinessValue, topic)}
  .transform(_.sortByKey(false))

val happiest10 = hashTags.map(hashTag => (hashTag.tail, 1))
  .reduceByKeyAndWindow(_ + _, Seconds(10))
  .transform{topicCount => wordSentiments.join(topicCount)}
  .map{case (topic, tuple) => (topic, tuple._1 * tuple._2)}
  .map{case (topic, happinessValue) => (happinessValue, topic)}
  .transform(_.sortByKey(false))
```

Figure 5: Measuring happiness of the filter word.



```
// Print hash tags with the most positive sentiment values
happiest60.foreachRDD(rdd => {
  val topList = rdd.take(10)
  println("\nHappiest topics in last 60 seconds (%s total):"
      .format(rdd.count()))
  topList.foreach{case (happiness, tag) =>
    println("%s (%s happiness)".format(tag, happiness))}
})

happiest10.foreachRDD(rdd => {
  val topList = rdd.take(10)
  println("\nHappiest topics in last 10 seconds (%s total):"
      .format(rdd.count()))
  topList.foreach{case (happiness, tag) =>
    println("%s (%s happiness)".format(tag, happiness))}
})
```

Figure 6: Print result in console window.

3.2 Workflow

The Spark streaming workflow is processed[1]. The loading of the streaming data source from Twitter tweets is done and such data is processed using Spark streaming. We are not using a storage system in this paper because our goal is limited to analyzing the tweets not stored.

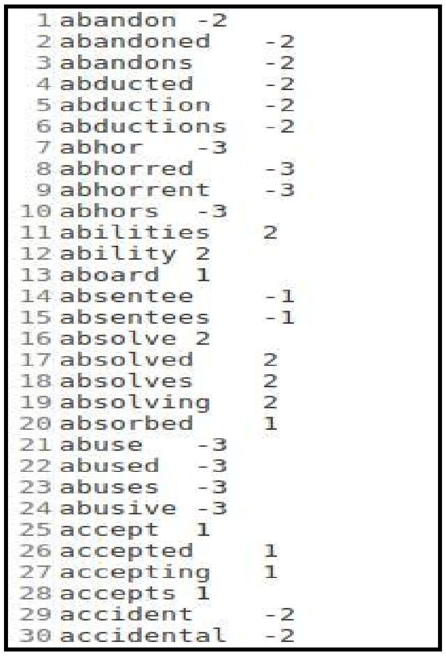
4. EXPERIMENTAL RESULTS

4.1 Experimental setup

We experimented Linux environment, using Ubuntu 16.04 LTS, Spark 2.2.0, and Scala 2.11 version. And we used Scala IDE (Eclipse oxygen) to develop Scala programming.

4.2 Dataset

We used Twitter streaming data from Twitter API. First, we need to make a new application in twitter application web site. After that, we are able to get authentication secret key from the website. Using those secret keys and tokens, we could gather Twitter streaming data in real time. And we used happiness word table from AFINN (Figure. 8). AFINN is an affective lexicon by Finn Årup Nielsen. The word list has been used for sentiment analysis and is developed in the Responsible Business in the Blogosphere project [7]. AFINN is a list of English words rated for valence with an integer between minus five (negative) and plus five (positive). The words have been manually labeled by Finn Årup Nielsen in 2009-2011. The file is tab-separated. There is Newest version with 2477 words and phrases [8].

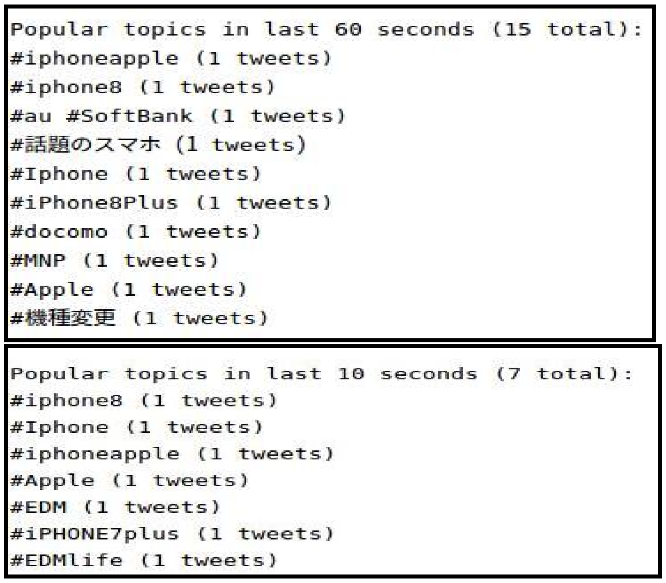


abandon	-2
abandoned	-2
abandons	-2
abducted	-2
abduction	-2
abductions	-2
abhor	-3
abhorred	-3
abhorrent	-3
abhors	-3
abilities	2
ability	2
aboard	1
absentee	-1
absentees	-1
absolve	2
absolved	2
absolves	2
absolving	2
absorbed	1
abuse	-3
abused	-3
abuses	-3
abusive	-3
accept	1
accepted	1
accepting	1
accepts	1
accident	-2
accidental	-2

Figure 7: AFINN Happiness word table..

4.3 Results

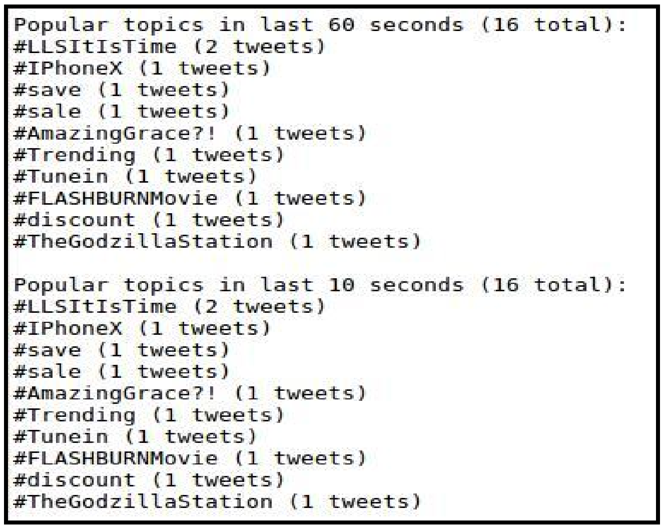
We gathered streaming data for 12 hours in each way. And we shows some useful part of the whole results because there are lots of output data produced by every 60 and 10 seconds.



```
Popular topics in last 60 seconds (15 total):
#iphoneapple (1 tweets)
#iphone8 (1 tweets)
#au #SoftBank (1 tweets)
#話題のスマホ (1 tweets)
#Iphone (1 tweets)
#iPhone8Plus (1 tweets)
#docomo (1 tweets)
#MNP (1 tweets)
#Apple (1 tweets)
#機種変更 (1 tweets)
```

```
Popular topics in last 10 seconds (7 total):
#iphone8 (1 tweets)
#Iphone (1 tweets)
#iphoneapple (1 tweets)
#Apple (1 tweets)
#EDM (1 tweets)
#iPHONE7plus (1 tweets)
#EDMlife (1 tweets)
```

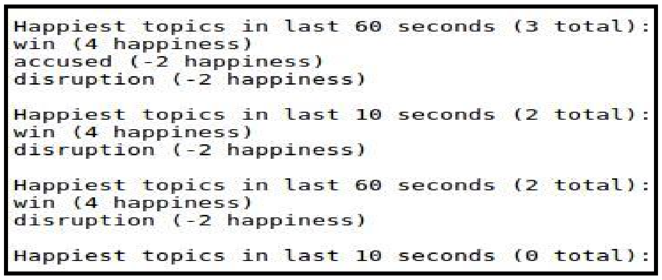
Figure 8: Result of popular Top 10 hashtags (1)



```
Popular topics in last 60 seconds (16 total):
#LLSItIsTime (2 tweets)
#IPhoneX (1 tweets)
#save (1 tweets)
#sale (1 tweets)
#AmazingGrace?! (1 tweets)
#Trending (1 tweets)
#Tunein (1 tweets)
#FLASHBURNMovie (1 tweets)
#discount (1 tweets)
#TheGodzillaStation (1 tweets)

Popular topics in last 10 seconds (16 total):
#LLSItIsTime (2 tweets)
#IPhoneX (1 tweets)
#save (1 tweets)
#sale (1 tweets)
#AmazingGrace?! (1 tweets)
#Trending (1 tweets)
#Tunein (1 tweets)
#FLASHBURNMovie (1 tweets)
#discount (1 tweets)
#TheGodzillaStation (1 tweets)
```

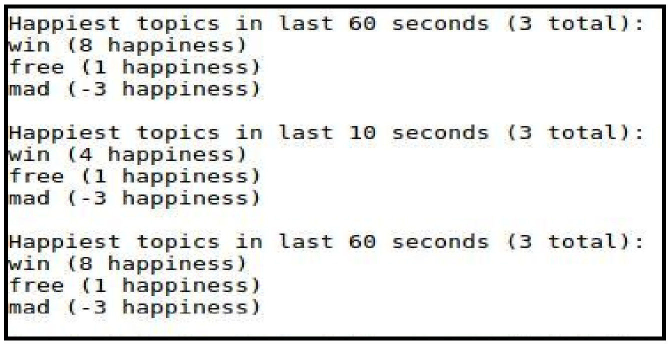
Figure 9: Result of popular Top 10 hashtags (2)

Figure 8 and Figure 9 are results of popular Top 10 hashtags written with “iphone8”



```
Happiest topics in last 60 seconds (3 total):
win (4 happiness)
accused (-2 happiness)
disruption (-2 happiness)

Happiest topics in last 10 seconds (2 total):
win (4 happiness)
disruption (-2 happiness)

Happiest topics in last 60 seconds (2 total):
win (4 happiness)
disruption (-2 happiness)

Happiest topics in last 10 seconds (0 total):
```

Figure. 10: Result of happiness value hashtags (1).



```
Happiest topics in last 60 seconds (3 total):
win (8 happiness)
free (1 happiness)
mad (-3 happiness)

Happiest topics in last 10 seconds (3 total):
win (4 happiness)
free (1 happiness)
mad (-3 happiness)

Happiest topics in last 60 seconds (3 total):
win (8 happiness)
free (1 happiness)
mad (-3 happiness)
```

Figure 11: Result of happiness value hashtags (2)

Figure 10 and Figure 11 are results of happiness value as a measure in tweets written with “iphone8”.

5. CONCLUSION

In this paper, we searched related hashtags from streaming data with our filter word “iphone8” and analyzed happiness value based on happiness table as a criteria. For future work, it would be better to connect to sentiment analysis, dividing positive and negative with hashtag “iphone8” to analyze public’s interest for marketing purpose. Furthermore, it isn’t limited about iphone8 but it can be other newly released product to determine whether the consumer will or will not buy the product. Twitter is one of the biggest platform for social network systems. If we can gather more information merged from other social network systems, we can create a novel sentiment analysis system for products.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2017R1D1A1B03034691).

REFERENCES

1. Apache Spark https://spark.apache.org/streaming/
2. Apache Spark - Spark Programming guide https://spark.apache.org/docs/2.2.0/rdd-programming-guide.html
3. Meng, Xiangrui, et al. "Mllib: Machine learning in apache spark." *The Journal of Machine Learning Research* 17.1 (2016): 1235-1241.
4. https://www.edureka.co/blog/spark-streaming/
5. Twitter API https://apps.twitter.com/
6. A. K. Jose, N. Bhatia, S. Krishna, "Twitter Sentiment Analysis", *National Institute of Technology Calicut*, 2010.
7. Aliza Sarlan, Chayanit Nadam, Shuib Basri, “Twitter sentiment analysis”, Information Technology and Multimedia (ICIMU), 2014 International Conference on https://doi.org/10.1109/ICIMU.2014.7066632
8. Happiness word table http://neuro.imm.dtu.dk/wiki/AFINN