

Report for lab assignment 6

Ting Xia ID: 30

1. Question:

Spark and Smartphone/Watch Application

Implement a smart application with big data analytics related to your project showing the collaboration between Spark and Smart Apps. Implement Twitter Streaming and perform word count on it and publish the results and showcase it in your Smart Phone/Watch Application.

Description:

Three steps:

1). Collect tweets using twitter streaming

```
System.setProperty("twitter4j.oauth.consumerKey", "R2v2WMKrF7UGipifRcMk0yjT1")
System.setProperty("twitter4j.oauth.consumerSecret", "InkVkJfUsJPQyA17GzGks9")
System.setProperty("twitter4j.oauth.accessToken", "3630687739-9y2qw6YKOMgeApm")
System.setProperty("twitter4j.oauth.accessTokenSecret", "IBjoDz21BTBaXwnJ13jy")

//Create a spark configuration with a custom name and master
// For more master configuration see https://spark.apache.org/docs/1.2.0/sub
val sparkConf = new SparkConf().setAppName("STweetsApp").setMaster("local[*]")
//Create a Streaming Context with 2 second window
val ssc = new StreamingContext(sparkConf, Seconds(2))
//Using the streaming context, open a twitter stream (By the way you can also
//Stream generates a series of random tweets
val stream = TwitterUtils.createStream(ssc, None, filters)
stream.print()
//Map : Retrieving text
val getText = stream.flatMap(status => status.getText.split(" "))
```

2). Do MapReduce on the collected data.

```
//Finding the top word used on 6 second window
val topCounts3 = getText.map(_._1).reduceByKeyAndWindow(_ + _, Seconds(6))
  .map{case (word, count) => (count, word)}
  .transform(_._sortByKey(false))
```

3). Store results in a string and send it to Android phone.

Just get the top 20 results

```
// Print popular words
topCounts3.foreachRDD(rdd => {
    val topList = rdd.take(20)
    println("\nPopular words used in last 6 seconds (%s total):".format(rdd.count()))
    topList.foreach{case (count, word) => println("%s (%s times)".format(word, count))}

    var s:String="Popular words used in last 6 seconds (%s total): \nWords:Count \n"
    topList.foreach{case(count,word)=>{
        s+=word+" : "+count+"\n"
    }}
    SocketClient.sendCommandToRobot(s)
})
```

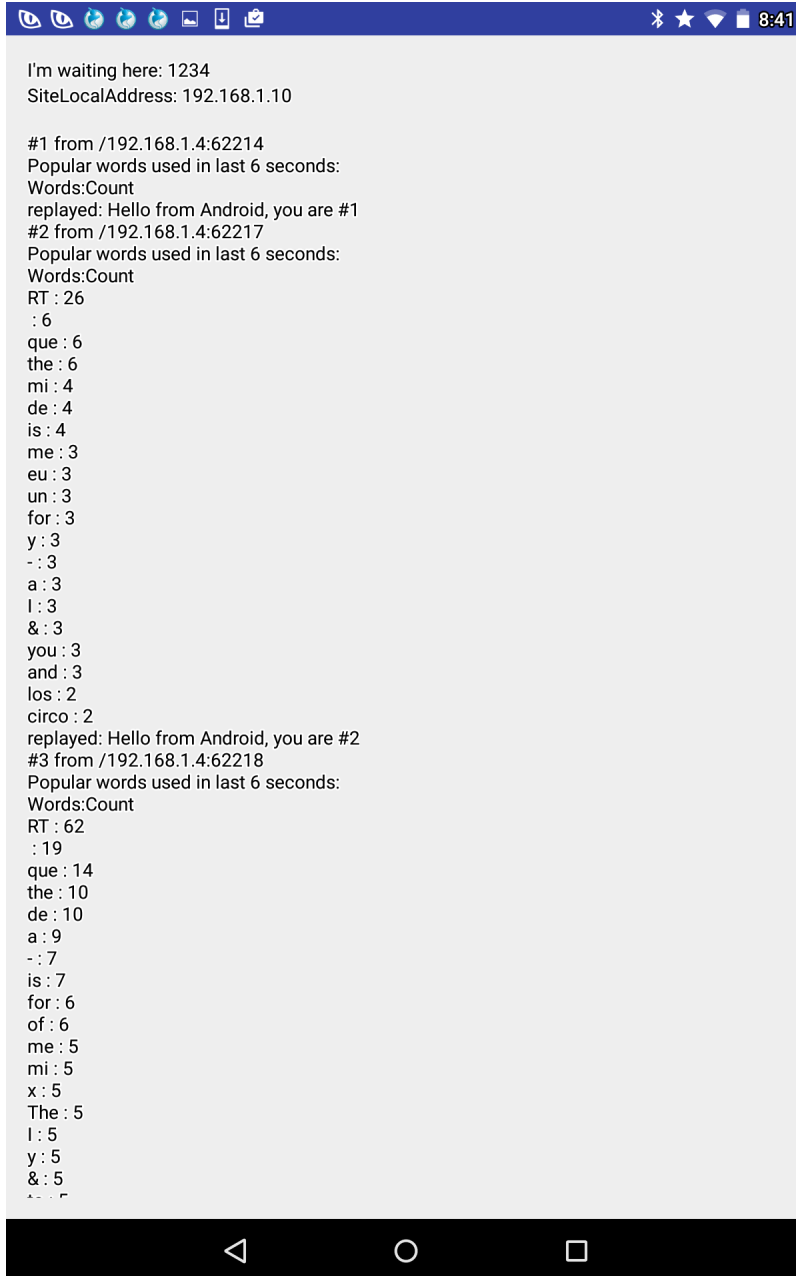
Screenshots:

Run log:

```
16/02/27 20:41:10 INFO DAGScheduler: ResultStage 23 (count)
16/02/27 20:41:16 INFO DAGScheduler: Job 9 finished: count

Popular words used in last 6 seconds (415 total):
RT (26 times)
(6 times)
que (6 times)
the (6 times)
mi (4 times)
de (4 times)
is (4 times)
me (3 times)
eu (3 times)
un (3 times)
for (3 times)
y (3 times)
- (3 times)
a (3 times)
I (3 times)
& (3 times)
you (3 times)
and (3 times)
los (2 times)
circo (2 times)
16/02/27 20:41:17 INFO MemoryStore: Block input-0-1456627:
-----
```

Android phone:



2. Question:

Spark ML Lib Application

Perform a machine learning algorithm with the Twitter Streaming data to categorize each Tweet

1) Training datasets: Collect different categories of Tweets related to your project.

(Categories can be based on HashTags / Subjects etc.)

2) Test data: the upcoming twitter stream.

Description:

Three steps:

1) Get the training data from twitter using object: *GetTrainingData*.

In order to do the category analysis, here I define three different hashtag categories: food, sport, and others. Others means hashtag is not food or sport. Tweets belong to the three different categories will be stored in different folders: hashtag.food, hashtag.sport, hashtag.other. (Only the non-empty data will be stored)

```
//Receive training data. Select three categories: food, sports, and other

val trainingFoodStream = stream.filter(_._2.getHashtagEntities.mkString.contains("food")).map(_._2.getText)
val trainingSportStream = stream.filter(_._2.getHashtagEntities.mkString.contains("sport")).map(_._2.getText)
val trainingOthersStream = stream.filter(!_._2.getHashtagEntities.mkString.contains("food")).filter(!_._2.getHashtagEntities.mkString.contains("sport")).map(_._2.getText)

val trainingFood = trainingFoodStream.foreachRDD(rdd =>
{
  val count = rdd.count()
  if (count > 0){
    rdd.repartition(1).saveAsTextFile("data/training/hashtag.food")
  }
})
val trainingSport = trainingSportStream.foreachRDD(rdd =>
{
  val count = rdd.count()
  if (count > 0){
    rdd.repartition(1).saveAsTextFile("data/training/hashtag.sport")
  }
})
val trainingOthers = trainingOthersStream.foreachRDD((rdd,time) =>
{
  val count = rdd.count()
  if (count > 0){
    rdd.repartition(1).saveAsTextFile("data/training/hashtag.other")
    val temp = count
  }
})
```

2) Get the testing data from twitter.

Testing data will be stored when the program is running.

```
val stream = TwitterUtils.createStream(ssc, None, filters)
/ stream.print()

//Testing data
val testing = stream.filter(!_.getHashtagEntities.isEmpty).map(_.getText)

val testingData = testing.foreachRDD(rdd =>
{ val count = rdd.count()
  if (count > 0){
    rdd.saveAsTextFile("data/testing/")
  }
})

ssc.start()
ssc.awaitTerminationOrTimeout(10)
```

3) Do machine learning algorithm to category the upcoming testing tweets.

a) Train the training data

b) Do the prediction on the testing data.

```
//Train tweets datasets
val sc = ssc.sparkContext
val stopWords = sc.broadcast(loadStopWords("/stopwords.txt")).value
val labelToNumeric = createLabelMap("data/training/")
println(labelToNumeric)
var model: NaiveBayesModel = null

val training = sc.wholeTextFiles("data/training/*")
.map(rawText => createLabeledDocument(rawText, labelToNumeric, stopWords))
val X_train = tfidfTransformer(training)
X_train.foreach(vv => println(vv))

model = NaiveBayes.train(X_train, lambda = 1.0)

val lines=sc.wholeTextFiles("data/testing/*")
val data = lines.map(line => {

  val test = createLabeledDocumentTest(line._2, labelToNumeric, stopWords)
  println(test.body)
  test
})

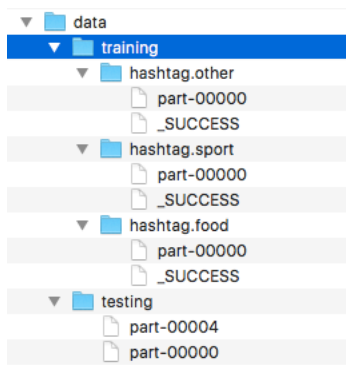
val X_test = tfidfTransformerTest(sc, data)

val predictionAndLabel = model.predict(X_test)
println("PREDICTION")
predictionAndLabel.foreach(x => {
  labelToNumeric.foreach { y => if (y._2 == x) {
    println(y._1)
  }
}
})
```

2) and 3) are done in one object.

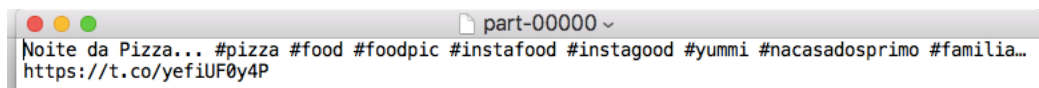
Screenshots:

Folder structure

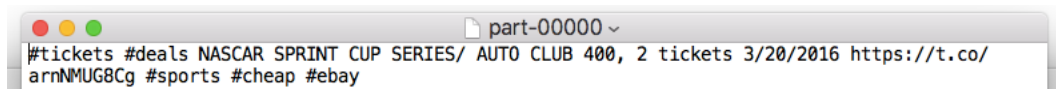


Example of training data:

Food

A screenshot of a text file named 'part-00000'. The file contains the following text: 'Noite da Pizza... #pizza #food #foodpic #instafood #instagood #yummi #nacasadosprimo #familia... https://t.co/yefiUF0y4P'.

Sport

A screenshot of a text file named 'part-00000'. The file contains the following text: '#tickets #deals NASCAR SPRINT CUP SERIES/ AUTO CLUB 400, 2 tickets 3/20/2016 https://t.co/arnNMUG8Cg #sports #cheap #ebay'.

Other

```
part-00000 ~
RT @ethiopianpeter: If it comes between @realDonaldTrump vs @HillaryClinton for president, I will chose not to vote #SouthCarolinaPrimary
I'm trying to binge watch all of Fuller House since we're getting rid of Netflix tomorrow but the wifi decided to die.😞
RT @la_iguanatv: Muere niña víctima del hambre en Colombia https://t.co/nMAXU9S0L https://t.co/sEZtw8Wwix
*
RT @EthanDolan: So close to meeting you all😞😞😞😞
RT @SyahmirulAmir: Drama melayu..asal budak u je kehulu hilir pegang buku kat tangan..dia xtau ke real life student g class kertas pon mint...
Miauuu
RT @EthanDolan: So close to meeting you all😞😞😞😞
RT @sorelatabIe: When your phone autocorrects to its ducking lit https://t.co/pLbFMaIMQv
@MEI_KMYD
おはよお!
RT @EthanDolan: So close to meeting you all😞😞😞😞
RT @oh_jessJessJESS: RT @iDntWearCondoms Ladies u just won 10 MILL... Your man in jail, bail 6 mill... How much do u have? https://t.co/Dhe...
I have really bad taste in guys sometimes
@sweet_mingal KKKKKKKK ja perdoou?
RT @kplove8226: These Things that Mean More to Men Than "I Love You" https://t.co/WuKJ1zyQ9j
https://t.co/BnmAtVPVFn
En un rato a lo de Rodri
RT @HmadiFatma: #ihabamir
#ايهابمغربيحتىالنخاع
مغربي حر بحبه وأخلاقه جعل علم المغرب
يرفع في الجزائر والجزائري يرفع في المغرب
🇵🇸🇵🇸🇵🇸 https://t.co/...
```

Example of testing data

```
part-00001 ~
*LIVE NOW*
*Husband & WIFEY! play RL*
https://t.co/TSEKWPRpkE
#RocketLeague #twitch @dirkened @RocketLeague https://t.co/Va06xixmq0
```

```

PREDICTION
16/02/27 21:01:07 INFO MemoryStore: Block input-0-1456628467000 stored as bytes in memory
16/02/27 21:01:07 INFO BlockManagerInfo: Added input-0-1456628467000 in memory on local
16/02/27 21:01:07 WARN BlockManager: Block input-0-1456628467000 replicated to only 0
16/02/27 21:01:07 INFO BlockGenerator: Pushed block input-0-1456628467000
16/02/27 21:01:07 INFO SparkContext: Starting job: foreach at TwitterCategoryAnalysis.sc
16/02/27 21:01:07 INFO DAGScheduler: Got job 31 (foreach at TwitterCategoryAnalysis.sc)
16/02/27 21:01:07 INFO DAGScheduler: Final stage: ResultStage 30 (foreach at TwitterCategoryAnalysis.sc)
16/02/27 21:01:07 INFO DAGScheduler: Parents of final stage: List()
16/02/27 21:01:07 INFO DAGScheduler: Missing parents: List()
16/02/27 21:01:07 INFO DAGScheduler: Submitting ResultStage 30 (MapPartitionsRDD[74] at
16/02/27 21:01:07 INFO MemoryStore: Block broadcast_36 stored as values in memory (estimated
16/02/27 21:01:07 INFO MemoryStore: Block broadcast_36_piece0 stored as bytes in memory (estimated
16/02/27 21:01:07 INFO BlockManagerInfo: Added broadcast_36_piece0 in memory on localhost
16/02/27 21:01:07 INFO SparkContext: Created broadcast 36 from broadcast at DAGScheduler
16/02/27 21:01:07 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 30 (MapPartitionsRDD[74] at
16/02/27 21:01:07 INFO TaskSchedulerImpl: Adding task set 30.0 with 2 tasks
16/02/27 21:01:07 INFO TaskSetManager: Starting task 0.0 in stage 30.0 (TID 131, localhost)
16/02/27 21:01:07 INFO TaskSetManager: Starting task 1.0 in stage 30.0 (TID 132, localhost)
16/02/27 21:01:07 INFO Executor: Running task 0.0 in stage 30.0 (TID 131)
16/02/27 21:01:07 INFO Executor: Running task 1.0 in stage 30.0 (TID 132)
16/02/27 21:01:07 INFO BlockManager: Found block rdd_65_1 locally
16/02/27 21:01:07 INFO BlockManager: Found block rdd_65_1 locally
16/02/27 21:01:07 INFO BlockManager: Found block rdd_65_0 locally
16/02/27 21:01:07 INFO BlockManager: Found block rdd_65_0 locally
16/02/27 21:01:07 INFO JniLoader: successfully loaded /var/folders/_n/x33qbl154b132v0p
16/02/27 21:01:07 INFO Executor: Finished task 1.0 in stage 30.0 (TID 132). 2044 bytes result size for task 1.0 in stage 30.0
16/02/27 21:01:07 INFO Executor: Finished task 0.0 in stage 30.0 (TID 131). 2044 bytes result size for task 0.0 in stage 30.0
16/02/27 21:01:07 INFO TaskSetManager: Finished task 1.0 in stage 30.0 (TID 132) in 82 ms
16/02/27 21:01:07 INFO TaskSetManager: Finished task 0.0 in stage 30.0 (TID 131) in 84 ms
16/02/27 21:01:07 INFO TaskSchedulerImpl: Removed TaskSet 30.0, whose tasks have all completed
16/02/27 21:01:07 INFO DAGScheduler: ResultStage 30 (foreach at TwitterCategoryAnalysis.sc) finished
16/02/27 21:01:07 INFO DAGScheduler: Job 31 finished: foreach at TwitterCategoryAnalysis.sc
hashtag.food
hashtag.other
hashtag.food
hashtag.other
hashtag.food
hashtag.food
hashtag.food
hashtag.food
hashtag.food
16/02/27 21:01:07 INFO StreamingContext: Invoking stop(stopGracefully=false) from shutdown

```