Final Presentation

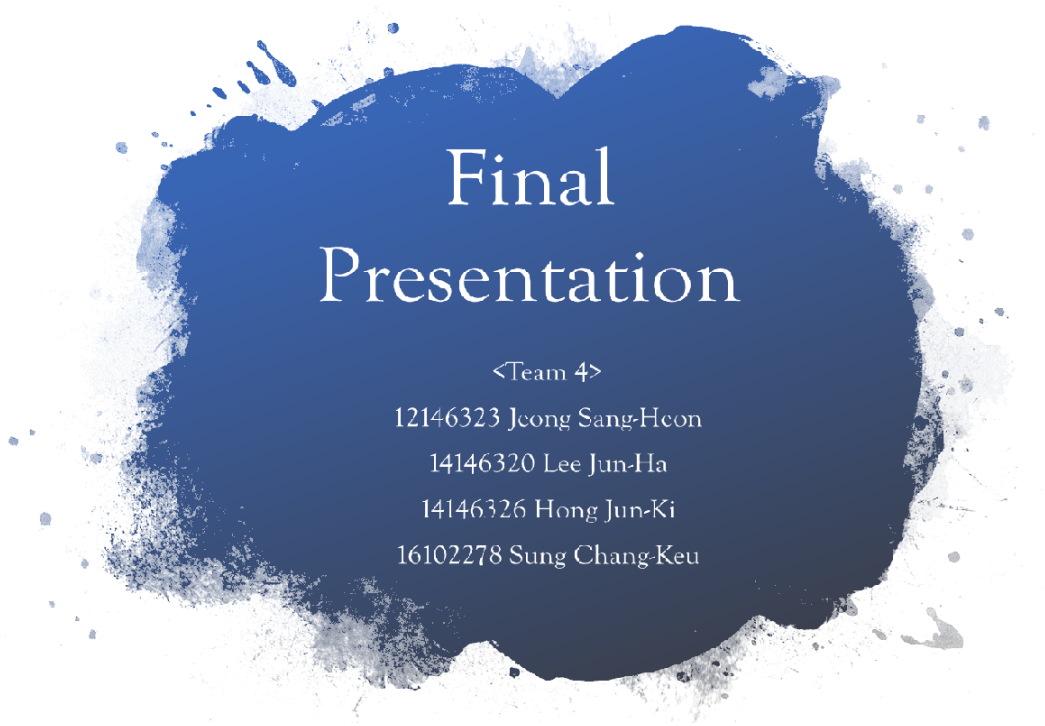
<Team 4>

12146323 Jeong Sang-Heon

14146320 Lee Jun-Ha

14146326 Hong Jun-Ki

16102278 Sung Chang-Keu



Contents

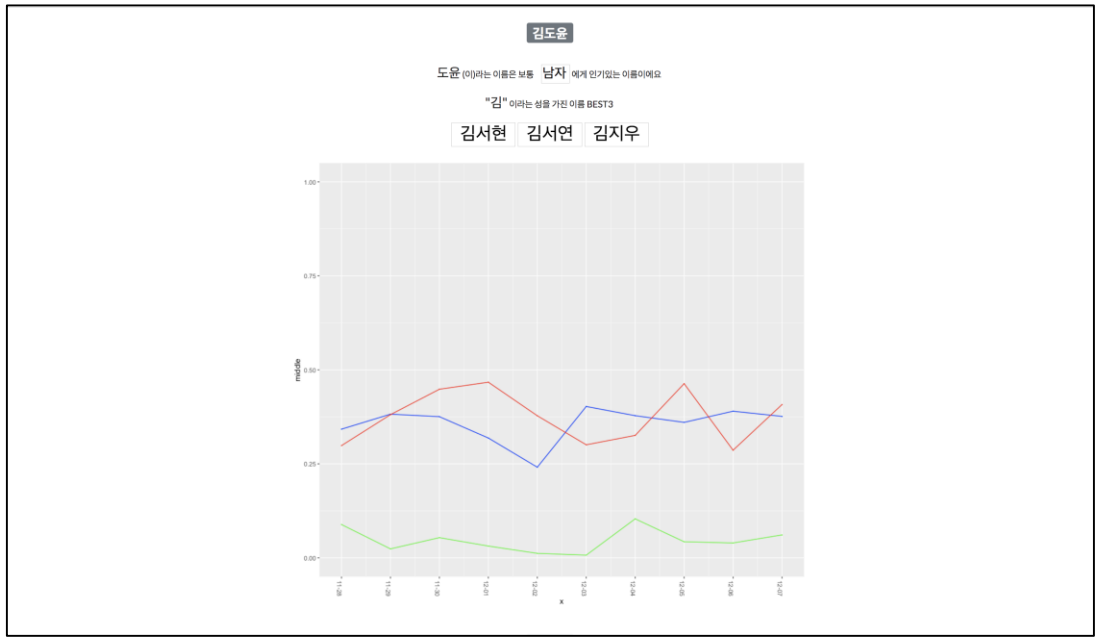
1. The result and methods of implementation
2. The used techniques
3. The overall steps of testing
4. The steps of analyzing the problems and debugging

1. The result and methods of implementation

1.1 Result of implementation

1.1.1 User Interface

- Tweets & Statistic survey of name
- Based on web service
- Search specific name → sentimental score graph will be shown
- Show the names of the top three people who have same last name.



1.1.2 Compare with Requirement specification

Functional Requirement	System	Sentiment Database	Select the appropriate information from the DB and print it to the user
	User	Web Page	Users search through web pages to get the information they want
Non-functional Requirement	System	Language	System language type
		Accessibility	The users can use service without difficulty and don't need guide to implement
		Update Cycle	Update system's modification
		Data-backup Cycle	Backup the system data
		Information Accuracy	Whether the system can provide users with validated data
		Cope with Errors	How to deal with errors
		Data Access	Correct access for the data
	User	Search the Preferences	Users can check the preferences for smartphone

<Requirement specification – SRS>

1. The result and methods of implementation

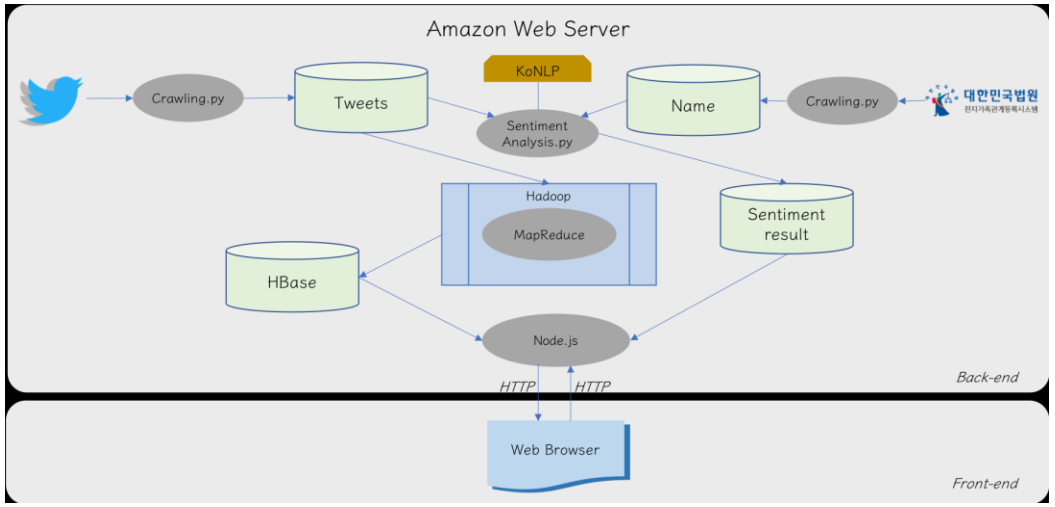
1.1.2 Compare with Requirement specification

Requirements	Score	Detailed evaluation
Sentiment database	4	Database deliver accurate information to certain input
Web page	5	User can view all the information provided on the web page
Language	1	Only Korean is supported
Accessibility	5	Web pages are easy to use because they are simple
Update Cycle	3	Since the maximum range of data crawl using Twitter is 10 days, the data must be updated once every 10 days. - Increased cost
Data backup Cycle	3	
Information accuracy	2	Insufficient validation of result
Cope with errors	4	Can handle the error which is caused by user input
Data access	5	Provide results using only information about the name user enter
Search the preferences	4	Shows the user's preference for the name they enter in scores and graphs

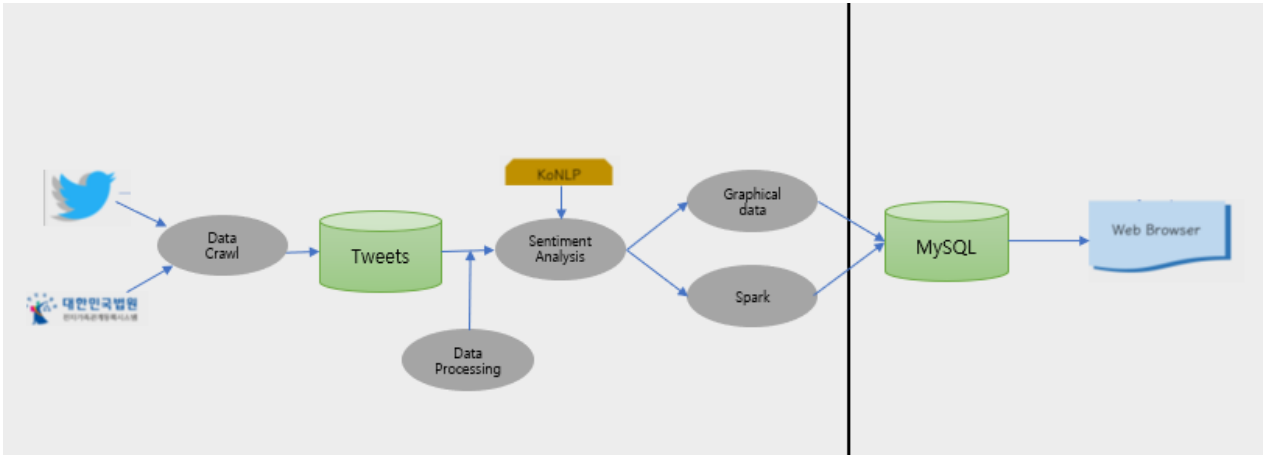
1. The result and methods of implementation

1.2 Method of implementation

1.2.1 Initial service flow



1.2.2 Final service flow



Methods	Language	Implementation	Input	Output
Crawl	Python	Crawl tweets from web site	(tweets) txt	json
Data Preprocessing	Python	JSON type transformation	json	json*
Sentiment Analysis	Java Script	Utilize sentiment analyzer library	json	(score) json
Graph generating	R	Generate time-series graph	json	jpg
Spark processing	pyspark	Transform DataFrame to calculate key-value pairs	DataFrame	txt
Database	MySQL	ETL graph and score data	txt / jpg dir	txt / jpg dir

2. The used techniques

2.1 The techniques for crawling the data – Tweepy package

Tweets mentioning the name

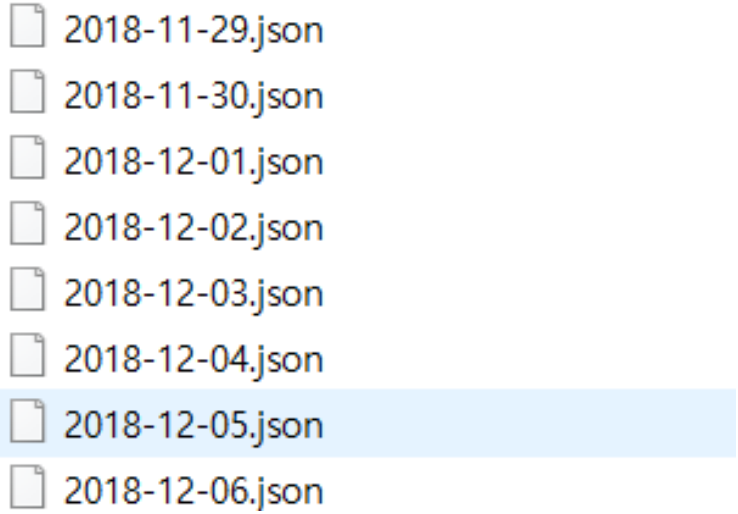
```python
def get_tweets(keyword, num_limit):
    i=0

    os.mkdir("\\Users\\hong\\Desktop\\크롤링데이터\\"+keyword)

    for tweet in tweepy.Cursor(api.search, q=keyword, since='2018-12-08', unti
        StatusObject = tweet._json
        dict1 = {
                'id': StatusObject['id_str'],
                'permalink':"",
                'username':StatusObject['user']['name'],
                'text': StatusObject['text'],
                'date':StatusObject['created_at'],
                'retweets': StatusObject['retweet_count'],
                'favorites': StatusObject['favorite_count'],
                'mentions':StatusObject['entities']['user_mentions'],
                'hashtags':StatusObject['entities']['hashtags'],
                'geo':StatusObject['geo']
                }
        print(str(i)+':'+str(dict1))

        ####날짜 형태 변환하기####
        unformatted = StatusObject['created_at']
        # Use re to get rid of the milliseconds.
        remove_ms = lambda x:re.sub("\+\d+\s","",x)
        # Make the string into a datetime object.
        mk_dt = lambda x:datetime.strptime(remove_ms(x), "%a %b %d %H:%M:%S %Y
        # Format your datetime object.
        my_form = lambda x:"{:%Y-%m-%d}".format(mk_dt(x))
```

Codes for crawling the tweets



Crawled data saved as JSON files

2. The used techniques

2.2 The techniques for processing the data – Tweepy package


```
[
  "id": "1068567487413813248",
  "permalink": "",
  "username": "상요",
  "text": "RT @eungalawyer: 1반남자들이(기바람 선동으로) 여학생 대상으로 인기투표해서 신경안쓰는척하지만
  "date": "Fri Nov 30 18:08:36 +0000 2018",
  "retweets": 29,
  "favorites": 0,
  "mentions": [
   {
    "screen_name": "eungalawyer",
    "name": "응가변호사",
    "id": 1025015594771349504,
    "id_str": "1025015594771349504",
    "indices": [
     3,
     15
    ]
   }
  ],
  "hashtags": [],
  "geo": null
},{
  "id": "1068551330048892928",
  "permalink": "",
  "username": "강현우",
  "text": "(그걸 또 들은건지 해맑게 웃으며 재현이 끌어안아) 당연히 강현우는 유재현꺼죠~ 아, 귀여워!",
  "date": "Fri Nov 30 17:04:24 +0000 2018",
  "retweets": 0,
  "favorites": 0,
  "mentions": [],
  "hashtags": [],
  "geo": null
},{
  "id": "1068490776345174017",
  "permalink": "",
  "username": "유재현",
  "text": "흔해도 예쁜 이름이 있잖아요. 난 마음에 드는데 강현우",
  "date": "Fri Nov 30 13:03:47 +0000 2018",
```


```
path = '/Users/hong/Desktop/크롤링데이터/'+names
extension = 'json'
os.chdir(path)
result = [i for i in glob.glob('*.{}'.format(extension))]

for file in result:
    open_file = open(file,'r', encoding = 'UTF8')
    read_file = open_file.read()
    new_content = "[" + read_file + "]"
    new_content2 = new_content.replace(",]","]")
    write_file = open(file,'w')
    write_file.write(new_content2)
    write_file.close()
```

Pre-process JSON format data to implement Morpheme Analyze Server & Twitter Sentiment Analysis without errors securely

```
pip3 install konlpy
ksa-server localhost 7000
```

(Morpheme analyzer server)

```
pip3 install konlpy
pip3 install JPype1
ksa-client localhost 7000 2017-01-01 2017-02-01 ./sourceDir ./targetDir
```

(Twitter sentiment analysis)

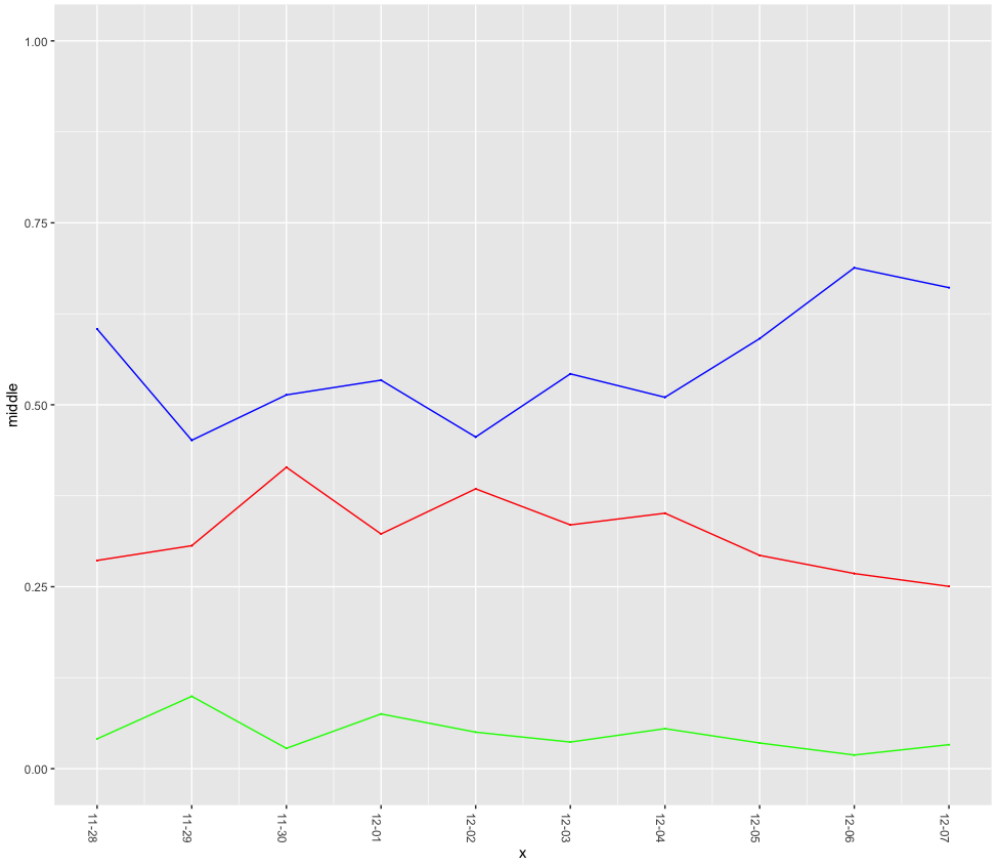

```
[{"id":"1067740711427104768",
    "permalink":"",
    "username":"라오",
    "text":"도윤이 생각난김에 비설 올려야지 에버노트에 올린건 아닌데 이거 생각하고 짠
    "date":"Wed Nov 28 11:23:17 +0000 2018",
    "retweets":0,
    "favorites":4,
    "mentions":[],
    "hashtags":[],
    "geo":null,
    "raw":"도윤이 생각난김에 비설 올려야지 에버노트에 올린건 아닌데 이거 생각하고 짠
    "polarity":{"com":0.15002740612142465,
    "pos":0.4907180830727546,
    "neg":0.25983683296288046,
    "neut":0.03922011399439713,
    "none":0.060197563848543204},
    "intensity":{
        "high":0.19258227442857145,
        "low":0.05744768228571429,
        "meium":0.6888359331428572,
        "none":0.06113411014285715},
    "expressive":{
        "dir-action":0.0015509855716501408,
        "dir-explicit":0.49228916149889845,
        "dir-speech":0.265855048752265,
        "indirect":0.18314091745473443,
        "writing-device":0.057163886722451984}},
{"id":"1067591229850845184",
    "permalink":"",
    "username":"조키하보고지구부순 사보",
    "text":"김도윤 심장에 스며든다,,,,ㅜ진자 미쳐ㅜ ㅅㅂ이런 미세먼지 오지는날에도 힘
    "date":"Wed Nov 28 01:29:18 +0000 2018",
```


(The results of sentiment analysis which are shown as positive/negative/neutral value, will be used for making the graph by R, and displaying the most preferred names)

2. The used techniques

2.2 The techniques for processing the data – R script

```r
ggplot.sentiment <- function (dataset) {
    # 감성그래프
    names = as.Date(unlist(list.map(dataset, return(data))))
    pos = unlist(list.map(dataset, return(polarity$pos)))
    neg = unlist(list.map(dataset, return(polarity$neg)))
    neut = unlist(list.map(dataset, return(polarity$neut)))
    df = data.frame(
        x = names,
        middle = pos*0+0.5,
        pos = pos,
        pos.label = 'Positive',
        neg = neg,
        neg.label = 'Negative',
        neut = neut,
        neut.label = 'Neutral'
    )
    color.pos = "blue"
    color.neg = "red"
    color.neut = "green"
    size.dot = 0
    gp = ggplot(df, aes(x, middle)) ## + geom_line()
    gp + ylim(0, 0.5) +
        geom_point(aes(x, pos), color = color.pos, size = size.dot) + geom_line(aes(x, pos), color = color.pos) +
        geom_point(aes(x, neg), color = color.neg, size = size.dot) + geom_line(aes(x, neg), color = color.neg) +
        geom_point(aes(x, neut), color = color.neut, size = size.dot) + geom_line(aes(x, neut), color = color.neut) +
        scale_x_date(date_labels = "%m-%d", date_breaks = "1 day") +
        theme(axis.text.x = element_text(angle = 270, hjust = 1))
        ## theme(axis.text.x = element_blank(),
    ##       axis.ticks = element_blank())
}
```



Visualize the result of sentiment analysis to graph by R

2. The used techniques

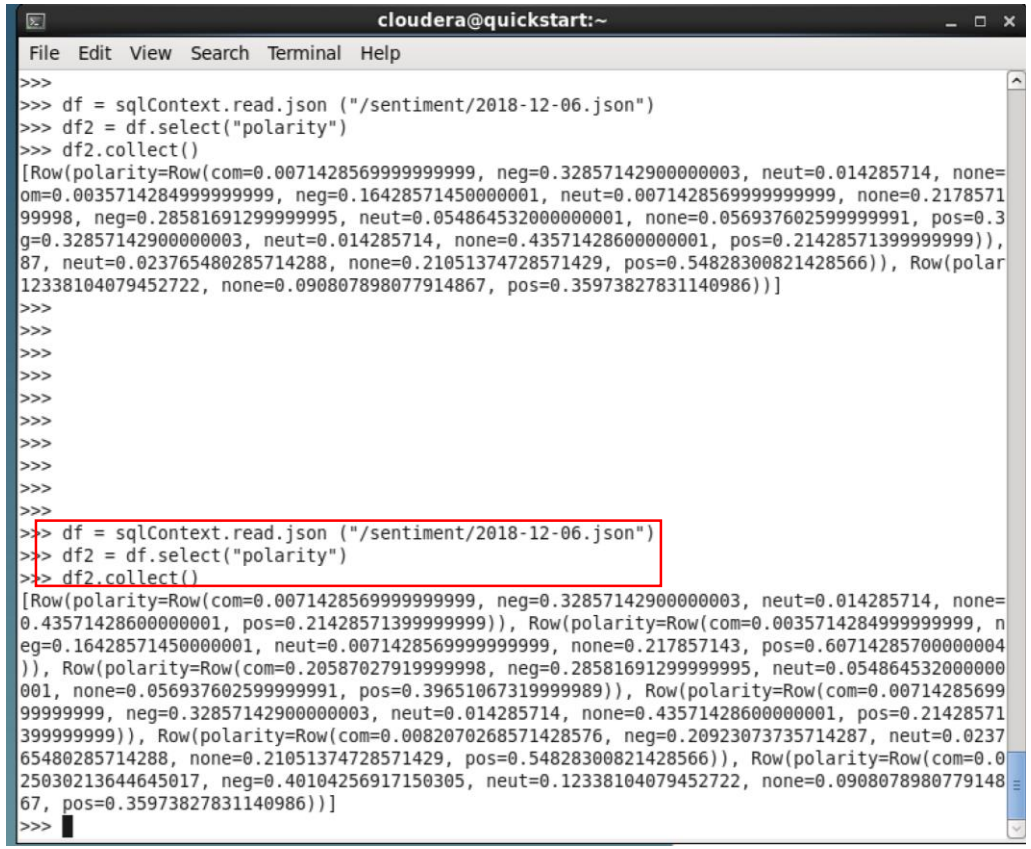
2.2 The techniques for processing the data – Spark

<JSON object>

/ sentiment / **2018-12-06.json**


```
[{"id":"1070710713209896960","permalink":"","username":"Today","text":"김 다원[도윤] /
KR / 27Y / 185cm ","date":"Thu Dec 06 16:05:01 +0000 2018","retweets":0,"favorites":0,"m
entions":[],"hashtags":[],"geo":null,"raw":"김 다원[도윤] / KR / 27Y / 185cm https://t.co/
iSJoYSdiuI","polarity":{"com":0.007142857,"pos":0.214285714,"neg":0.328571429,"neut":0.0
14285714,"none":0.435714286},"intensity":{"high":0.142857143,"low":0.114285714,"meium":0
.307142857,"none":0.435714286},"expressive":{"dir-action":0,"dir-explicit":0.18571428581
42857,"dir-speech":0.29999999969999996,"indirect":0.0857142859142857,"writing-device":0.
42857142857142855}},{"id":"1070708466321260545","permalink":"","username":"n인 트친소 맞괄
 시운","text":"김도윤 오빠만 ","date":"Thu Dec 06 15:56:05 +0000 2018","retweets":0,"favori
tes":1,"mentions":[],"hashtags":[],"geo":null,"raw":"김도윤 오빠만 https://t.co/vGzNPi41G3
","polarity":{"com":0.0035714285,"pos":0.607142857,"neg":0.1642857145,"neut":0.007142857
,"none":0.217857143},"intensity":{"high":0.0714285715,"low":0.057142857,"meium":0.653571
4285,"none":0.217857143},"expressive":{"dir-action":0,"dir-explicit":0.09285714295357143
,"dir-speech":0.649999999675,"indirect":0.042857142978571426,"writing-device":0.21428571
439285712}},{"id":"1070683636238700545","permalink":"","username":"해나","text":"@N00Y0D
제가 원피스가 참 안 어울리더라고요 그 김에 다이어트 해서 저거 입을게용. 입구 도윤님이랑 데이트 가야징
~~~~ ♥ ","date":"Thu Dec 06 14:17:25 +0000 2018","retweets":0,"favorites":0,"mentions":[
{"screen_name":"N00Y0D","name":"도윤","id":1032808823856160800,"id_str":"1032808823856160
768","indices":[0,7]}],"hashtags":[],"geo":null,"raw":"@N00Y0D 제가 원피스가 참 안 어울리더
라고요 그 김에 다이어트 해서 저거 입을게용. 입구 도윤님이랑 데이트 가야징~~~~ ♥ https://t.co/tppG
aPDugv","polarity":{"com":0.20587027919999998,"pos":0.3965106731999999,"neg":0.285816912
99999995,"neut":0.054864532,"none":0.05693760259999999},"intensity":{"high":0.3176313629
3647376,"low":0.046715927790656815,"meium":0.5746305416850739,"none":0.06102216758779556
```


<Data parse in type of data frame>



2. The used techniques

2.2 The techniques for processing the data – Spark

<Data parse in type of Spark Context>

```
>
> df = sqlContext.read.json("/sentiment/*")
> df2 = df.select("polarity")
> df2.coalesce(1).rdd.saveAsTextFile("/sentiment/result.txt")
>
```

/ sentiment / result.txt / **part-00000**

```
Row(polarity=Row(com=0.15002740612142465, neg=0.25983683296288046, neut=0.03922011399439
7128, none=0.060197563848543204, pos=0.49071808307275461))
Row(polarity=Row(com=0.12971109175000001, neg=0.30588705012500006, neut=0.196922183375,
none=0.228433507625, pos=0.13904616712499995))
Row(polarity=Row(com=0.025864422012932212, neg=0.32989081016494542, neut=0.0294358505147
17924, none=0.21785714310892859, pos=0.39695177419847588))
Row(polarity=Row(com=0.15002740612142465, neg=0.25983683296288046, neut=0.03922011399439
7128, none=0.060197563848543204, pos=0.49071808307275461))
Row(polarity=Row(com=0.12971109175000001, neg=0.30588705012500006, neut=0.196922183375,
none=0.228433507625, pos=0.13904616712499995))
Row(polarity=Row(com=0.025864422012932212, neg=0.32989081016494542, neut=0.0294358505147
17924, none=0.21785714310892859, pos=0.39695177419847588))
Row(polarity=Row(com=0.14464285703616073, neg=0.42142857160535718, neut=0.02142857125535
7142, none=0.10892857152723215, pos=0.30357142857589281))
Row(polarity=Row(com=0.0066244239999999994, neg=0.29531490025000001, neut=0.024539170500
000002, none=0.19925115199999999, pos=0.47427035325))
Row(polarity=Row(com=0.0037274219999999998, neg=0.246174056, neut=0.016650246200000003,
none=0.22622331700000001, pos=0.50722495879999996))
Row(polarity=Row(com=0.008832565333333332, neg=0.56041986700000013, neut=0.0327188940000
00005, none=0.15455709166666667, pos=0.24347158199999999))
Row(polarity=Row(com=0.14464285703616073, neg=0.42142857160535718, neut=0.02142857125535
7142, none=0.10892857152723215, pos=0.30357142857589281))
```

```
>>> rdd = sc.textFile("/sentiment/result.txt/*")
>>> rdd.take(1)
[u'Row(polarity=Row(com=0.15002740612142465, neg=0.25983683296288046, neut=0.039
220113994397128, none=0.060197563848543204, pos=0.49071808307275461))']
>>>
>>> rdd1 = rdd.map(lambda line:line.split(",")[1])\
...            .map(lambda line:line.split("="))\
...            .map(lambda x: (x[0], float(x[1])))
>>> rdd1.take(1)
[(u' neg', 0.25983683296288046)]
>>> neg_sum = rdd1.reduceByKey(lambda v1,v2: v1+v2)
>>> print neg_sum
PythonRDD[87] at RDD at PythonRDD.scala:43
>>> neg_sum.collect()
[(u' neg', 20.807473277488825)]
>>>
```

(negative_sum)

```
>>> rdd2 = rdd.map(lambda line:line.split(",")[2])\
...            .map(lambda line:line.split("="))\
...            .map(lambda x: (x[0], float(x[1])))
>>> rdd2.take(1)
[(u' neut', 0.039220113994397128)]
>>> neut_sum = rdd2.reduceByKey(lambda v1,v2: v1+v2)
>>> neut_sum.collect()
[(u' neut', 2.8746634063707814)]
>>>
```

(neutral_sum)

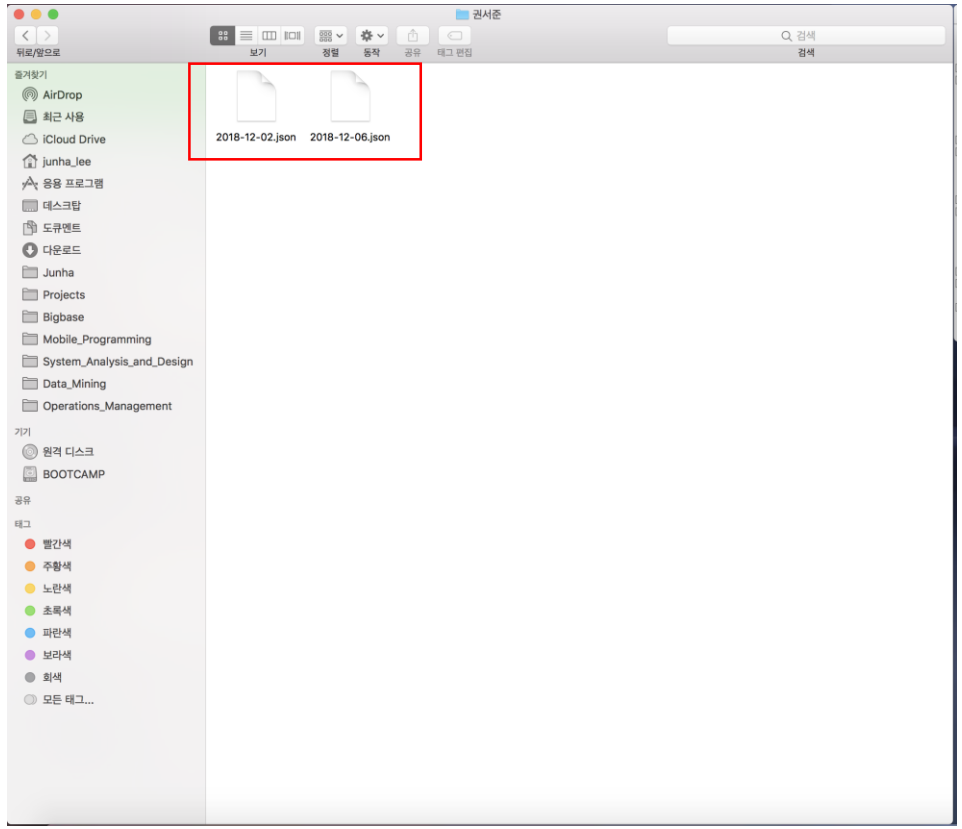
```
>>> rdd3 = rdd.map(lambda line:line.split(",")[4])\
...            .map(lambda line:line.split("="))\
...            .map(lambda x: (x[0], x[1]))
>>> rdd4 = rdd3.map(lambda (v1,v2): (v1,float( v2.split("))")[0]) ))
>>> pos_sum = rdd4.reduceByKey(lambda v1,v2: v1+v2)
>>> pos_sum.collect()
[(u' pos', 19.955320971382207)]
>>>
```

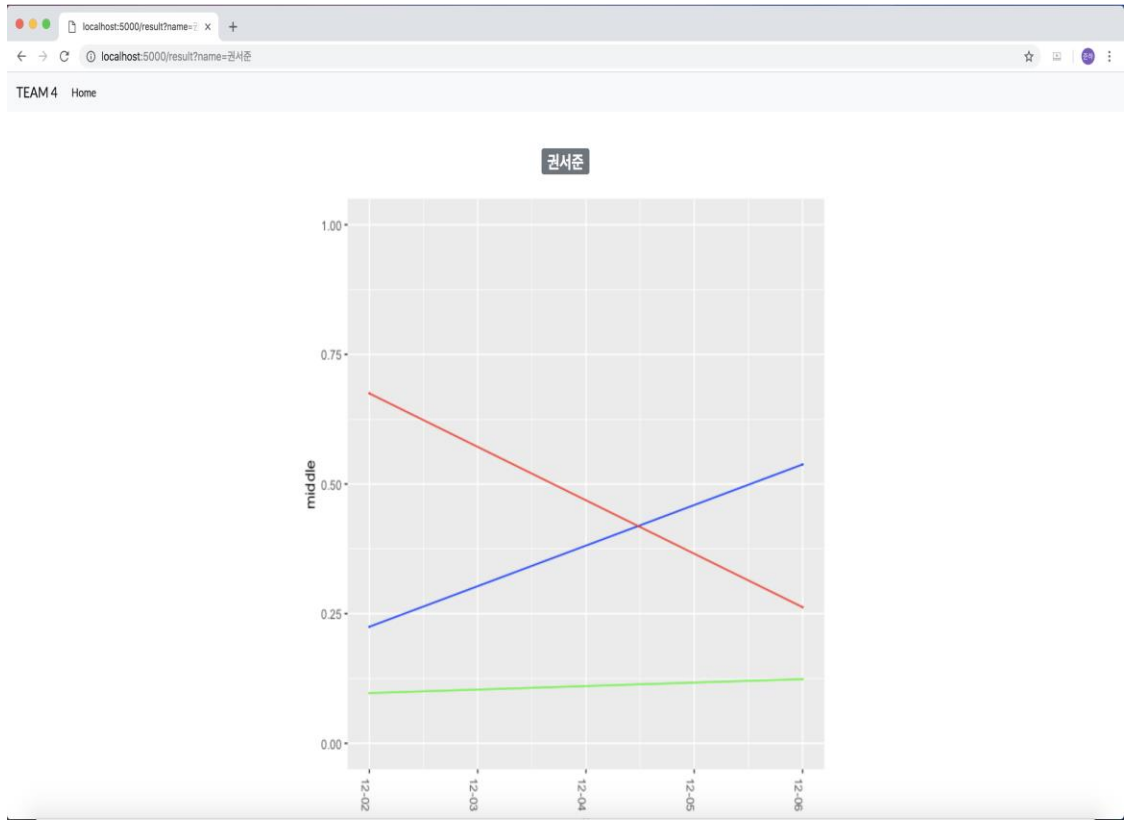
(positive_sum)

3. The overall steps of testing

Integration testing

1) Test the case if the name which does not have enough data is inputted



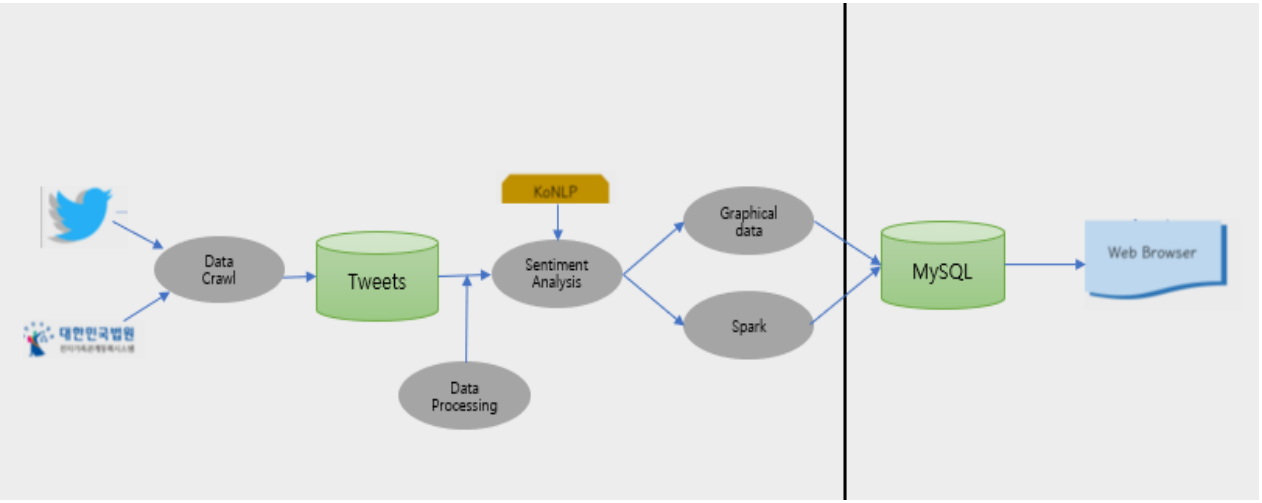


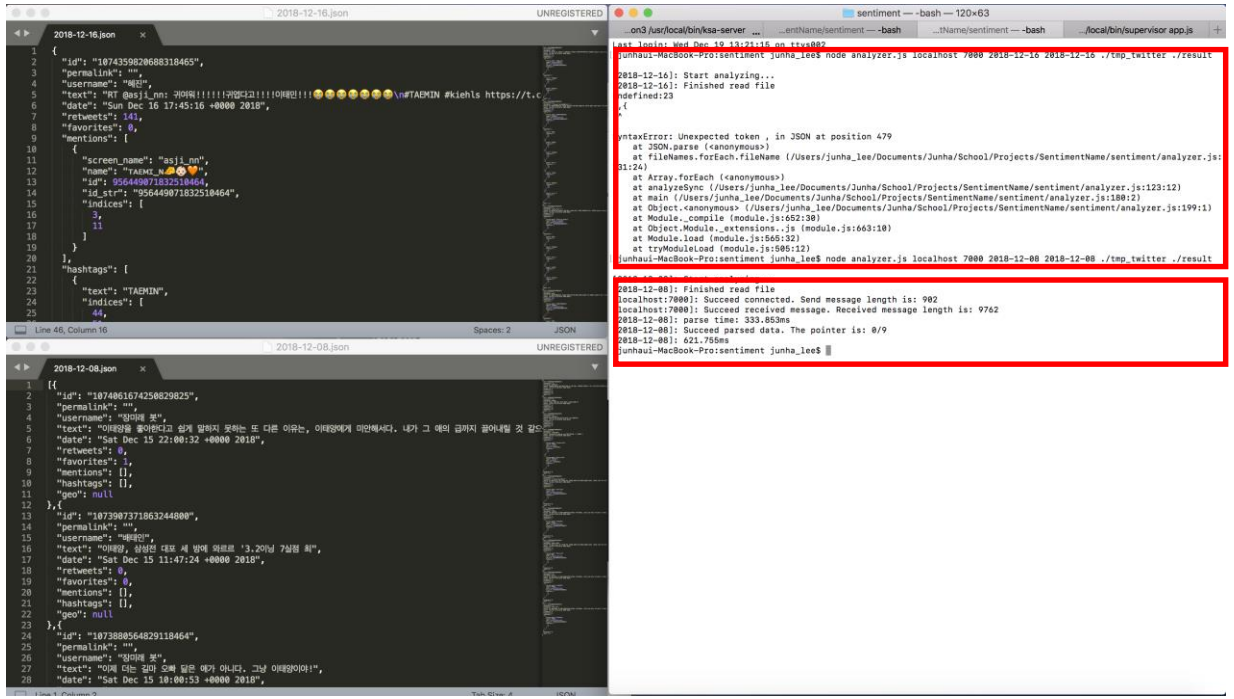
→ Some of names don't have enough data to derive meaningful result

3. The overall steps of testing

Unit testing

2) Test whether the entire process is managed and implemented well as intended





Although all the flow process is constructed, the entire functions should be checked several times to make sure that every steps are under the control and to avoid possible errors

3. The overall steps of testing

Unit testing

3) Check whether the retweets also have similar value with regular tweets in sentiment analysis

Sentiment Analysis of the Correlation between Regular Tweets and Retweets

Jundong Chen[1], He Li[2,3], Zeju Wu[2], and Md Shafaeat Hossain[4]

[1]Department of Math and Computer Science, Dickinson State University, ND 58601, USA
[2]School of Communication and Electronic Engineering, Qingdao University of Technology, China
[3]College of Mechanical Engineering, Qingdao University of Technology, China
[4]Computer Science Department, Southern Connecticut State University, New Haven, CT 06515, USA
Jundong.Chen@dickinsonstate.edu, qdhehe@qut.edu.cn, wuzeju@qut.edu.cn, HossainM3@SouthernCT.edu

***Abstract*—In this paper, we study the influence from the sentiment of regular tweets on retweeting. We propose a method to calculate the sentiment score for each tweet and each Twitter user. This method enables us to place the tweets and retweets into the same time period to explore the sentiment factor. We adopt the correlation coefficient between the sentiment scores of regular tweets and those of retweets to measure the influence. We categorize the Twitter users in three different ways to investigate three factors, which are the number of followers, betweenness centrality and the types of accounts. Community detection and machine learning are integrated into our approach. We find that the difference for correlation coefficients exists between different levels of the number of followers, and different types of users. Our method sheds a light on better predicting the dynamics of tweets diffusion by including the sentiment factor into the prediction model.**

***Index Terms*—sentiment analysis, social network, machine learning, Twitter classification**

I. Introduction

Twitter is a popular platform for sharing information. The shared information spreads in a cascading way through followers and followers of followers, etc. Understanding tweets diffusion is very important for anyone who wants to spread their information in an efficient way, such as online advertisers to conduct targeted marketing campaigns, and policy makers to efficiently propagate their policies [1]. One of the most important mechanisms for information diffusion on Twitter is retweeting.

Researchers have been striving to discover the pattern of retweeting, and a step further to predict retweeting. Retweet prediction aims to solve the problem that which tweets are more likely to be retweeted by which Twitter users. Petrovic et al. [2] used a machine learning approach to predict which tweet is retweeted and when is retweeted. Luo et al. [3] developed a prediction model based on time series to better understand the underlying mechanism of retweeting behaviors. Bradlow et al. [4] designed a probabilistic model to forecast the total number of retweets based on a Bayesian approach. However, up to now, the approaches proposed only help us to understand retweeting and barely provide a precise prediction model.

To better predict retweeting, it is necessary to investigate and analyze various factors which impact retweeting. Boyd et al. [5] analyzed why people retweet and what people retweet by elaborating on some examples and case studies about retweets. Suh et al. [6] identified factors that are significantly associated with retweet rate. Those factors are some of the content features and contextual features. They also built a linear model to predict the retweet rate from those features.

Researchers also try to include sentiment as a factor for tweets diffusion. Ferrara et al. conducted a quantitative analysis of the effect of sentiment on information diffusion [7]. They attempted to answer some questions, such as whether positive tweets or negative ones spread faster, what types of emotions of tweets are more popular, etc. To the best of our knowledge, not much work has been done to investigate the sentiment as a factor for retweeting.

We develop an approach to study the impact from sentiment on retweeting, which is not investigated by Ferrara et al. [7]. First we design a Twitter crawler to collect tweet contents. We calculate the sentiment score for each tweet for each Twitter account. Then we find the correlation for the sentiment scores between original tweets and retweets. We categorize all the Twitter accounts in three different ways. The first way is based on the number of followers. The second way is based on the betweenness centrality. The third one is based on different types of accounts. The correlation coefficients are calculated under the three different ways of categorization.

The structure of this paper is as follows. We first discuss related work and introduce our work in the introduction section. Then, in Section II we explain the process of data collection and why it is impossible to analyze retweeting by first drawing a complete route for the propagation of a tweet. Our methodology for the sentiment analysis is provided in Section III. We calculate and compare the correlation

978-1-5386-1465-5/17/$31.00 ©2017 IEEE

(Thesis about correlation between tweets-retweets)

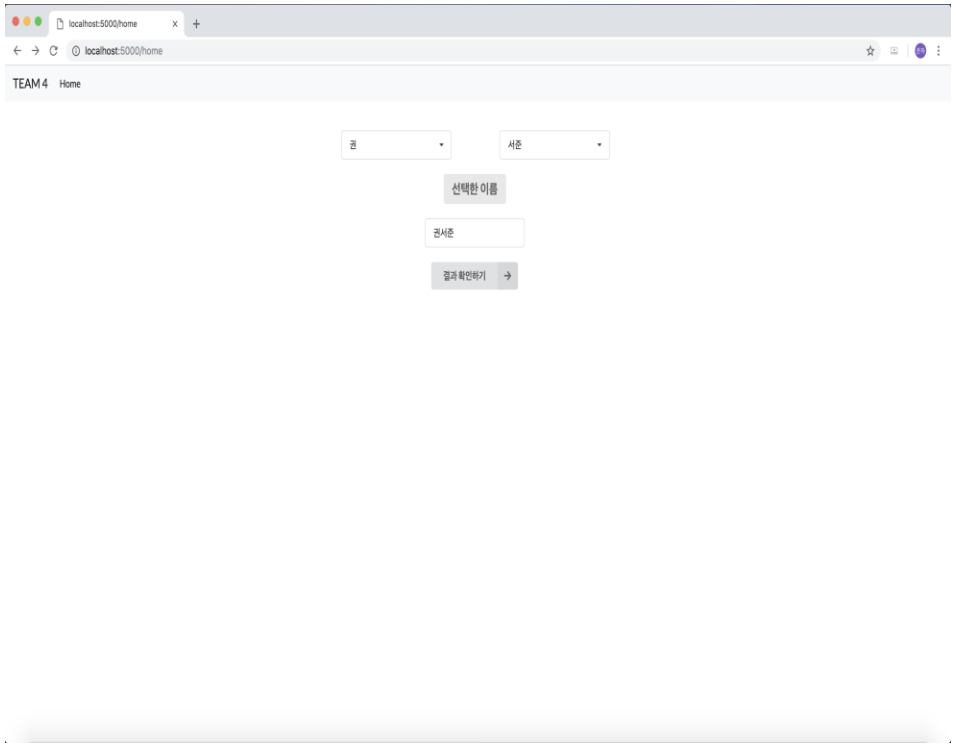
After checking the dataset, we find that only 8.4% of all the users have negative average sentiment scores for regular tweets, and 9.0% for **retweets**. That means most of the users have positive sentiment scores for their average one-day tweets.

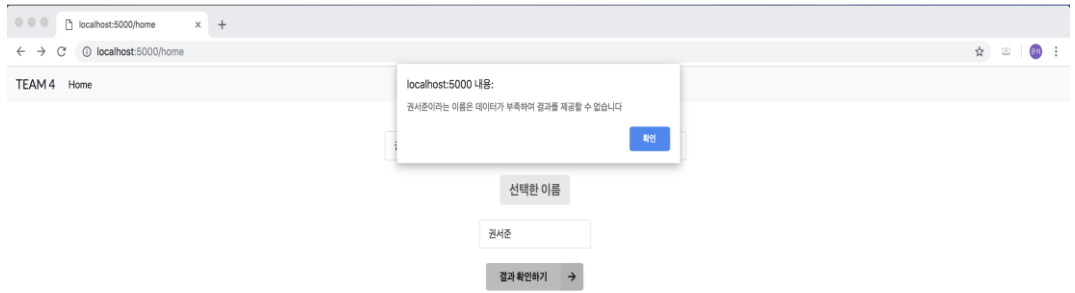
In this research, it is said that retweets(9.0%) show similar average sentiment scores with regular tweets(8.4%)

4. The steps of analyzing the problems and debugging

4.1 Case 1 – The amount of saved data is not enough to show meaningful results

Integration testing



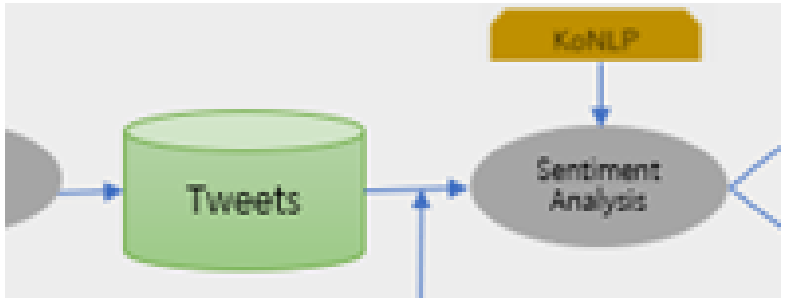


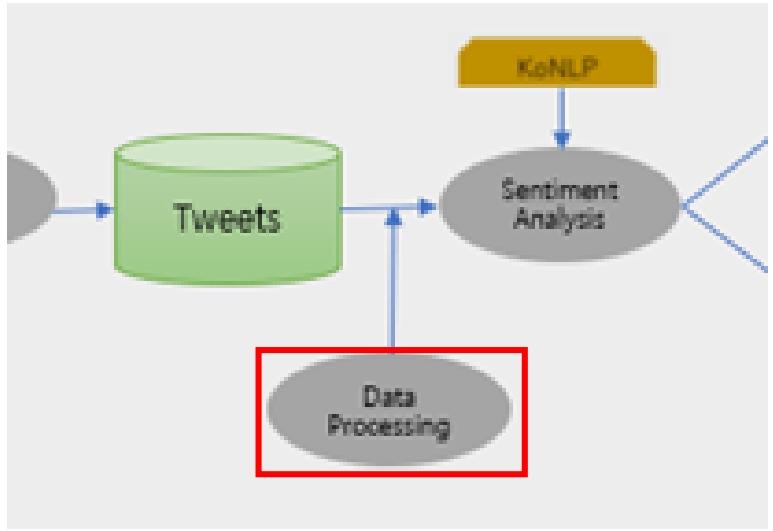
If the name with lack of data is inputted,
web would show pop-up message to notify that the search is unavailable

4. The steps of analyzing the problems and debugging

4.2 Case 2 – Cannot read the previous JSON format while implementing sentiment analysis

Unit testing





Couldn’t read the JSON format to implement sentiment analysis

Add the steps for processing data before implementing sentiment analysis

4. The steps of analyzing the problems and debugging

4.3 Case 3 – whether to accept retweets while crawling the tweets

Unit testing

```
30
31  #트윗 크롤링을 날짜 기준으로 저장합니다.
32  def get_tweets(keyword, num_limit):
33      i=0
34      for tweet in tweepy.Cursor(api.search, q=keyword, since='2018-11-01', until='2018-11-14',lang="ko").items(num_limit):
35          if (not tweet.retweeted) and ('RT @' not in tweet.text):
36              StatusObject = tweet._json
37              dict1 = {
38                      'id': StatusObject['id_str'],
```

```
30
31  #트윗 크롤링을 날짜 기준으로 저장합니다.
32  def get_tweets(keyword, num_limit):
33      i=0
34      for tweet in tweepy.Cursor(api.search, q=keyword, since='2018-11-01', until='2018-11-14',lang="ko").items(num_limit):
35          if (not tweet.retweeted) and ('RT @' not in tweet.text):
36              StatusObject = tweet._json
37              dict1 = {
38                      'id': StatusObject['id_str'],
```

While processing the result of analysis,

we found that the retweets are also have enough value for representing the preference of keywords

Therefore, we deleted the line

which exclude the retweets from crawled data


Thank you