SINERGI Vol. 23, No. 1, February 2019: 1-10
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi.2019.1.001

DESIGN OF PEOPLE PROFILING AND MODELING REPUTATION COMPUTATION BASED ON SENTIMENT ANALYSIS

Ahmad Mafazi Damanhuri, Zhang Huaping
Computer Science, School of Computer Science, Beijing Institute of Technology
Side Rd of N. 3rd Ring Rd W, Haidian Qu, Beijing Shi, ZhongGuo
Email: a.mafazi24@gmail.com kevinzhang@bit.edu.cn

Abstract -- *The number of popular people is still growing because of the easiness to access information technology. Every time people upload things and let people watch it and give it a like or comment. People who can impress other people will grow their popularity and fame. Some famous people make influences, help poor people with powers, and others are causing troubles. Community these days drives people perspective by share their thoughts on social media. They spread information and makes others want to see things they are talked about. Troublesome popular people defended by their fan base and attacked by other communities. By these cases, the research tried to gather information on social media and used it for calculation and profiling. The method that proposed to rely on this information is based on sentiment analysis to look up someone's record and listing them into top 10 best got from DBpedia. This system shows the list of people and contains all important record about that person which can be used for decision support for a policy or rewarding people. The results have successfully visualized the output in the list of people with any further details following by clicking their names.*

Keywords: *DBpedia; Decision Support; Sentiment Analysis; Text mining*

Received: May 25, 2018 Revised: November 5, 2018 Accepted: November 8, 2018

INTRODUCTION

A number of popular people are always growing in every day due the social media is growing too and make people more comfortable to know each other and show exciting things that they've got, and they can do. Through sentiment analyst, it can discover someone is the person that favorite enough or so much popular, because of good things or because of bad things either. By DBpedia to find articles, tweets, etc. as the source we going to connect the result with the trained model, to find out who is the most famous people and because of what are those people can be so much distinguished. Some people may know our favorite can be the one who has many good records or can be the one who has many bad records. Thera are categorizing process which people who have good careers and list them from the most to the least (Tavakolifard et al., 2013; Yu et al., 2012; Can, 2011).

In 2016, the microblogging service averaged at 317 million monthly active users. Millions of them are these famous people, and the rest are people who talk about them. For example, as of July 2016, @realDonaldTrump had 10,267,655 followers and still growing adding an average of 30,574 new followers per day. It's so outperforming because recently his tweets have been retweeted a total of 12 million times. Donald Trump includes a hashtag in almost every other tweet for example #Trump2016 which is used 279 times and #MakeAmericaGreatAgain which is used 186 times.

The paper proposed people profiling using a sentiment analysis system. The sentiment analysis system is an ideal choice for modeling the reputation of a public figure from people thoughts in social media (Hussein, 2018; Ranjan et al., 2018; Ozturk & Avyaz, 2018). The system also can differentiate the sentences into positive and negative sentences, calculate those sentences to create a score of likes and dislikes toward someone, and makes a list of people who have liked the most and having the dislikes the most. Its performance is good enough; hence can process thousands of data of tweets in a short time without much of manual work. The results hopefully have successfully visualized the output in the list of people with any further details following by clicking their names.

MATERIAL AND METHOD

Sentiment analysis was being used by companies to calculate people thoughts about their products or services. They decided to use sentiment analysis because it calculates the percentage of positive, negative and neutral opinion from people. It helped the company owner to make a wise decision and able to improve their products or services to satisfy people needs. They also paid attention to the aspect-based sentiment analysis, as it focuses on aspects being targeted by the reviewer. The reviewer gives their sentiments about a specific character of their products or services. It will narrow the review to a particular area and improve.

User Profiling

The use of user profiling previously done by Gulla et al. (2014). In their work, they have estimated and rank the evaluations of news articles to a user. The proposed method gives an advance recommendation technology to fulfill user preferences of news reading. Further, their research shows the categories that user enjoyed to read. It also considered the relevant news to the user profile.

A running context for a user is built from all user acts of that user that have happened after the last time a full user profile was generated. It describes the overall topics of what the user has been clicking on or reading lately without reflecting what she might have been interested in at earlier occasions. The running context gives us the user's current news focus. The mobile news app records every gesture from the user and maintains an updated running context at all time. Even the user decides to reset his context, the running context is compared with his old user profile on the server side. They presume to construct the user profile of a particular user following by these steps (1) extract interest from user acts, (2) build running context from all user acts, (3) combine running context and long-term interests into a new user profile (Lian et al., 2018).

However, experiments with the news recommender system show that some issues need to be careful of, their research implemented the category interests and content interests equally important. Many users would prefer a stronger focus on either the category as Sport or on its particular topics as its player or just sure team. The balance between stable long-term user interests and short-term news context should be delicate, and even if long-term benefits are preferred, the user risks less there is no relevant news available. And should there is a balance between profile-relevant news stories and also report that are not directly within a user's profile to trigger new interests and widen their perspective. The system is highly configurable, with some parameters that seriously affect the news stories recommended to the user no visible best configuration of the system. They assume wisely expanding the recommender system with semantic features for modeling news event and entities. A new log-in feature also intends to use social media sites like Twitter to deepen the understanding of the user's preferences.

Online Reputation Management

Reputation dimensions contribute to a better understanding of the topic of a tweet or group of tweets, while author profiling provides essential information for priority ranking of tweets. In 2014, RepLab focused on two aspects of reputation analysis, which is Reputation Dimensions Classification and Author Profiling (Amigo et al., 2014). RepLab has always been focused on Twitter content, as Twitter is the essential media for early detection of potential reputation issues. Reputation dimensions classification purpose is to assign tweets to one of the seven standard reputation dimensions of the RepTrak Framework developed (Cossu et al., 2013; Amigo et al., 2013a; Amigo et al., 2013b).

These dimensions reflect the practical and cognitive perceptions of the company by any stakeholder groups (Performance, Products & services, Leadership, Citizenship, Governance, Workplace, and Innovation). Author profiling is composed of two subtask, Author categorization that classify twitter profiles by type (i.e. Company, Professional, Celebrity, Employee, Stockholder, Investor, Journalist, Sportsman, Public Institution, and Non-Governmental Organization) and Author Ranking to find out which authors had more reputational influence and which profiles are less or have no influence at all. Aspects that determine the impact can be the number of followers, number of comments on a domain or type of author.

They let participants involved in this research, 11 groups of 49 groups have submitted result in the time. Eight groups participated in the Reputation Dimension task, and five groups introduced their effect to Author Profiling. They included a baseline that employs SVM using words as features besides the participant systems. By classifying every tweet as majority class baseline would get an accuracy of 56% for Reputation Dimension Classification task the top systems used a variety of methods such as a basic Naïve Bayes approach. Tweets that labeled as "Undefined" harmed their performance. Replacing the "Undecidable" labels by "Product and Services" just giving a similar result. Most of the systems tend to assign the majority class "Product and Services" to a greater extent than the gold standard. One of participant applying distance to class vectors get the most significant *accuracy* and the most tweets being processed. Author categorization, on the other hand, proved to be challenging in this initial approximation.

SFL (Systemic Functional Linguistics)

Computational linguistics is still linguistics, and this is something that they think especially these days people may lose sight of because a lot the computational linguistics world is very highly computational. There are a problem that deals with language theory. To solve the problem, you're counting words in a text then you are saying that the only useful feature of the language that you need is there are words in a text, and some of them occur more frequently. That is not just a statistical model that helps you solve a problem that's a reflection on how you're thinking about language when you're trying to solve that problem. Every time you do something computationally you are making some assumptions about how language works, and that is something that they are found is valuable.

SFL is probabilistic if there is most of the way you are producing is not by discrete choices but by skewing endure in particular directions this its network. System networks from the basis of the description of grammars within SFL, they have semantically organized a network of choices doing possible distributions. They will end up having realizations in particular grammatical constructions or particular words but that decision is only made at the most delicate level within their system network, up until then you are making choices between possible distributions or meaning. By selecting across all the system in grammar at once, you will end up determining the final text. So, if they had two clauses and we'd like to join them together in some way, then the way we merge these two clauses expresses a meaning between those two clauses, so in SFL this is broken down in terms of conjunction system. It could choose to use the second clause to elaborate upon the first clause so they will be performing elaboration. It can be used to clarify the second clause to specify the first clause.

MediaWiki API

MediaWiki action API provides developers code-level access of the entire Wikipedia reference. MediaWiki API can be used to high-level access to the data contained in MediaWiki databases. It is going to use to get all the names of famous people and their ontology. The API uses RESTful calls and support a wide variety of format including XML, JSON, PHP, YAML, etc. (Qureshi et al., 2014).

Twitter

Twitter is a treasure trove of sentiment; people around the world output thousands of reactions and opinions on every topic under the sun every second and every day. Twitter is becoming a psychological database that's continually being updated, and we can use it to analyze millions of text snippets in seconds with the power of machine learning (Cossu et al., 2014).

System Planning

Fig. 1 shows a diagram that describes the task and flow of system planning.

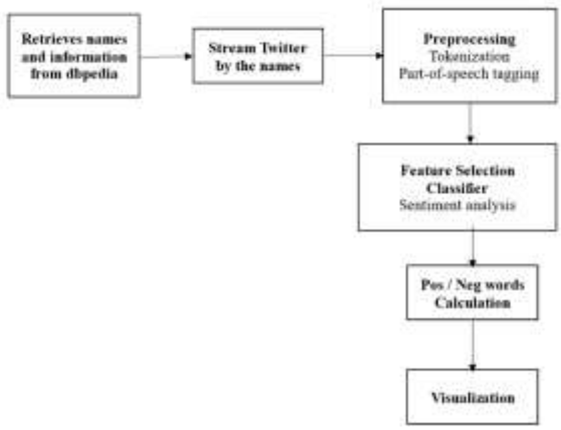

Figure 1. Task and Flow of System Planning

The paper is arranging over the task above, and the first task is about retrieving names. We were planning to retrieves all of the names which come to the internet, and we know DBpedia provides many data about most important people. So, we were going to coding with SPARQL to get DBpedia data. The data that we want to retrieves are the link to the article and the name. We download it to our directory in JSON format.

These files are going to be opened by the python program, and the name is read for being the query in Stream twitter step. In the stream twitter step, the tweet that is going to show up is the tweet that includes the question which is the name of a specific person in it. And the tweets are tweets from someone who sent it to or talked about him (the person) or from the person himself. All tweets will be saved in a JSON file with format 'person_name.json'.

By calling the files with format above in specific directory we able to read all the tweet that saved in JSON files. Those tweets are going to be tokenized and classified by Textblob. Textblob is one of a python module that able to process NLP and help with tokenization and sentiment analysis (Cambria & White, 2014). Textblob is going to divide the tweet one by one, word by word and classify them positive or negative tweets. The tweet that has been classified is saved and kept in a different directory.

Positive and negative tweets are saved with format "ID+person_name+PosTweets.json" or "ID+person_name+NegTweets.json" so we were able to call the tweets of specific person only if needed.

The tweet that has been classified can be scored for sentiment purposes. By using Textblob feature ".sentiment.polarity" where tweet will be scored -1 to 1. Less than 0 for negative and more than 0 for positive. The train data that we are going to use is provided by textblob and the default library of textblob is pattern library. The purpose of this step is to measure the likes and the dislikes by calculating the number of positive and negative tweets.

The visualization step is going to read the data from sentiment step. It is going to arrange the array of people from the one who is having the most likes or from the one who is having the most dislikes. This page will be a list of 5 people and showing their names, description, some likes and dislikes and the position of that person in the list. It will be two pages to show the detail even more. The second page will be teaching some positive tweets and the tweets and also negative tweets.

DBpedia

By querying on DBpedia frontend (https://dbpedia.org/sparql) using SPARQL language, aims to retrieves all names from all of the countries. Names stored to JSON file to be read by the application that we are going to use which is C# and Python. Fig. 2 shows a flowchart of retrieving data.

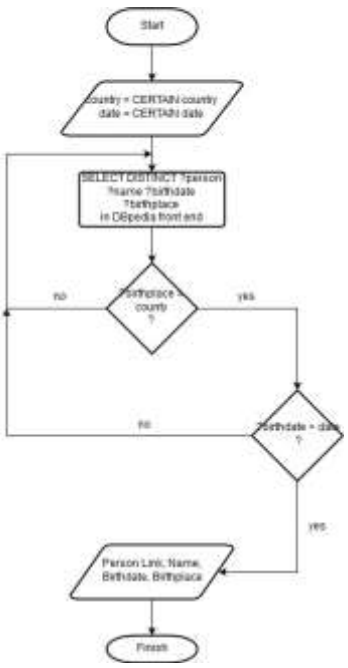

Figure 2. Flowchart of retrieving data from DBpedia

Since this DBpedia limits the output to 9999 data at once, so we only able to retrieve all names by a country that we put in the filter, we again filter the output by alphabet and birth date related to who possibly are alive these nowadays this meant to get the output less than 9999 data.

Twitter Streaming API

Once the names stored in JSON files and saved in the specified directory. We can call the JSON file and load all the names inside, and use them to the next task. The study uses Twitter API and applies keyword filtering to crawl most relevant tweets, which then is stored in a database for easier retrieval and manipulation. To use the API first thing, we need to do is register to get token keys.

After registering and requesting permission to Twitter for building an application, some keys and token were received. These keys are necessary for Twitter to grant consent while streaming the tweets. This study made the use of **Tweepy**, a python module that connects to Twitter Streaming API with modification to suit the study. Fig. 3 shows a flowchart of parsing twitter tweets.

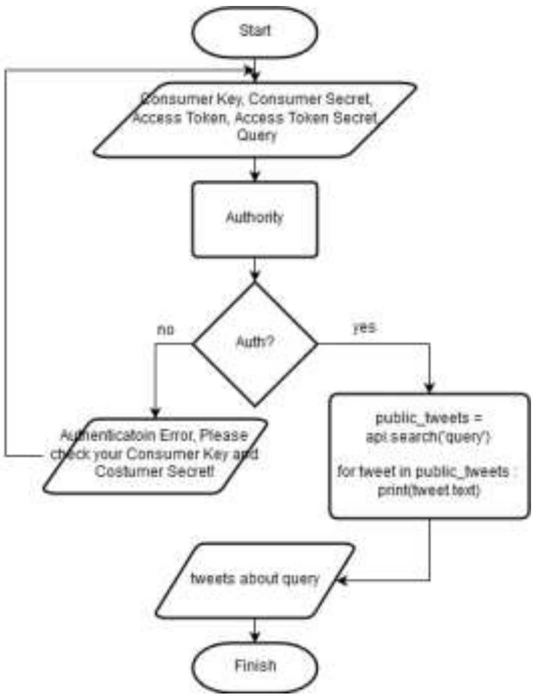

Figure 3. Flowchart of parsing twitter tweets

Once we've been through the authorization, we can search tweets that include our query inside their text body and parse the text only, the poster, the date or another attribute. The question itself is one of the names from DBpedia. So, the tweets are going to tweet from their self or someone to them. But tweepy limits the output only tweets from the day of the running system to three days behind. Again, all the tweets are going to be saved to JSON file according to the name of the query.

Preprocessing

The next step is preprocessing of text we used textblob to manage tokenization, part-of-speech tagging, noun phrase extraction, sentiment analysis, and language translation and detection. Other twitter centric features are also removed like URL and punctuations.

a. Tokenization

>>> from textblob import TextBlob

>>>

>>> wiki = TextBlob("I enjoying myself live in beijing and studying at BIT")

>>> wiki.words

WordList(['I', 'enjoying', 'myself', 'live', 'in', 'beijing', 'and', 'studying', 'at', 'BIT'])

b. Part-of-speech Tagging

Each sentence object also has an attribute that works as a unit called phrases. Phrases can be divided into a smaller phrase and forming a structure. Textblob makes extracting noun phrases super easy:

>>> from textblob import TextBlob

>>>

>>> wiki = TextBlob("I enjoying myself live in beijing and studying at BIT")

>>> wiki.tags

[('I', u'PRP'), ('enjoying', u'VBG'), ('myself', u'PRP'), ('live', u'JJ'), ('in', u'IN'), ('beijing', u'NN'), ('and', u'CC'), ('studying', u'VBG'), ('at', u'IN'), ('BIT', u'NNP')]

Tags are assigning to a single word according to what is its role in the sentence. Traditional grammar classifies words based on eight part of speech: verb, noun, pronoun, adjective, adverb, preposition, conjunction, and interjection.

Sentiment Analysis

Sentiment analysis will be used this time right after the current tweet is tokenized by textblob. Textblob is also going to perform its sentiment analysis feature ".sentiment.polarity". Textblob used NaiveBayes Library. Textblob has values 0 to 1 for positive words and 0 to -1 for negative words. Textblob could look up the sentiment value for each word from a sentiment lexicon that has it all pre-recorded to classify the total sentiment value of our tweet. If a sentiment exists in a dictionary, the sentiment will be scored, and it is counted toward a person, and the score is calculated. And this step will be going on and on until all sentiment is all checked and there are no tweets scored for each person.

After getting the number of people and the number of tweets of each person, the number of likes dislikes, a score of positive or negative tweets will be set to 0 for the first time. And it will be added in every time it occurs inside the loop together with sentiment process. A flowchart of sentiment analysis is shown in Fig. 4.

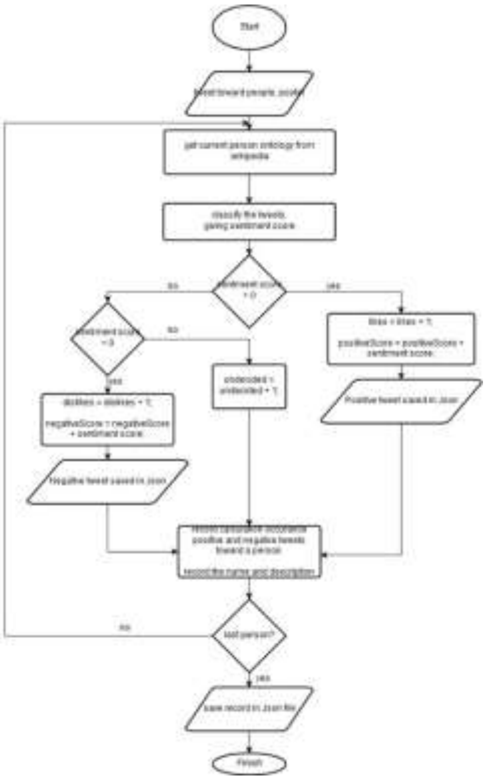

Figure 4. Flowchart of a Sentiment analysis process

Visualization

All calculation result recorded inside an array, hence every element of the collection contains the JSON file name and the likes and dislikes calculation the variety can be arranged from the most likes or the most dislikes. C# will arrange the array and provide the list by clicking the button of "Likes the most" or "Dislikes the most". C# window can handle the visualization part, and the design will be done on Photoshop. Fig. 5 and Fig. 6 show diagrams that describe the configuration for visualization model.

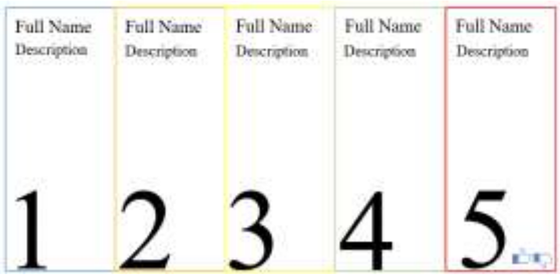

Figure 5. Design for visualization model (first page)

At this time, we will have two pages to visualize this model, and the first page will show five people from the most likes and most dislikes. This page contains their full name, description, and position on the list and there are two buttons on the right corner to show like or dislike the most.

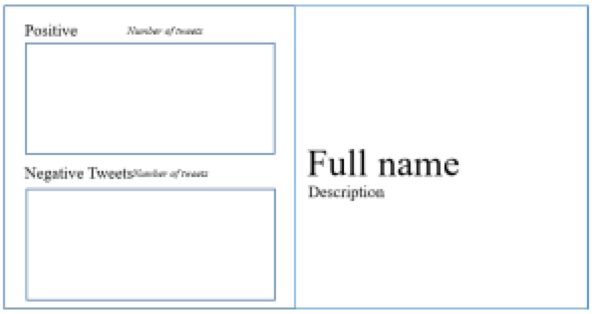

Figure 6. Design for visualization model (second page)

The second page can be accessed by clicking the full name label of a person to show us the detail of that person like his / her number of positive tweets and negative tweets, text of positive tweets also negative tweets.

RESULTS AND DISCUSSION

Data Set

The data set that is used in this project is retrieved from two different sources. The first one retrieved from DBpedia by coding on https://dbpedia.org/sparql using SPARQL language, and retrieved 1603 names from a country called Indonesia.

The names that we got from DBpedia is the query for getting the next data set which is tweets from twitter. In this step, we are coding on python, and we are using a python library called tweeps. By read the name from JSON file and parse the tweets from 3 days before the system is running (March $27^{th,}$ 2017) we have got 81,825 tweets.

SPARQL

In purpose to retrieving people names who internet ever is known, we look into DBpedia to looking for the database used by Wikipedia. We looked after all people in DBpedia, but it limits its output to 9999 names only. Next, we filtered the output by birthplace, which is we need to name a country to define the birthplace. We used Indonesia as a filter for these people's birthplace. And the result showing 1603 names even the name of independent fighter included in it. We filtered the birthdate right after that and showing the expected result.

The code showed numbers of politicians, musicians, singers, sports players (footballer, tennis player, golf player, etc.), actors and actresses and the other names whoever recorded on the internet with their achievement. Fig. 7 shows names retrieved from DPpedia.

Figure 7. Names retrieved from DBpedia

Tweepy

We need to register to use twitter API to get the consumer key and access token. Consumer key, consumer secret is used to confirm the authority of the Twitter user. The following code is my authority keys:

Authentic with twitter which means login via code, to do that we will create the variable code off for authentication and use the OAuth handler method of tweeps. This method takes two arguments, the consumer key, and the consumer secret. This method was written inside of the tweepy library with a bunch of code that is hidden to us. We can name the method, and all of its functionality is in our hands. The arguments are what the method uses to perform its internal calculation. We are halfway through the authentication. The other half is to call the set access token method on the auth variable which takes two arguments, the access token, and the access token secret. That is, we created out authentication variable. Next, we are going to search for tweets for our use case we want to collect tweets that a specific keyword in this a person name. To do that create a public tweets variable that is going to store a list of tweets to fill it, we will call the search method of the API variable. The search method takes a sing argument, the query which is the name. This method is going to retrieve a bunch of that contain the word this “person name”.

The writer is having trouble with the workspace, the pc that the writer use to do this experiment cannot parse the tweets because of the proxy. Even VPN is used to perform this step the system kept showing "time out". Hence, the writer borrowed another pc from his mate. This incident makes we can retrieve all tweets from each person in a JSON file.

Because of that, we took ten random names from the file. We used these names as a query to find the tweets that include their names inside the text body. Tweepy limit the output only will be the tweets by the date of the system is running until the tweets from 3 (three) days ago. And it will be the tweets "To" this person or "From" this person that will be showing up.

The tweets that retrieved contains another attribute not only the tweet text itself. It includes the data of the poster (the one who posted it), the item that attached such image, video, and link, and contain data retweeted from another post.

Experiments

We ran some test and minimized the model on my system. We called a first hundred tweets from one of person and used textblob to extract it. We used textblob noun phrase extraction by access through the noun phrases property, and the system can understand it. Fig. 8 show some part of the results.

Figure 8. List of noun phrase from first 100 tweets

Fig. 8 is lists of noun phrase from 100 sentences. The system extracted those words which are known as noun or word from another language. These are a total of 513 words from 100 sentences, which is quite good. It extracts the word that normally denoted by NN, NNS or NNP all of which indicate Noun. For example, the sentence “@iko_uwais You are an amazing martial artist Iko. The Raid is one of the best action films I have seen. You're very talented #theraid”. The parse is shown in Fig. 9.

[(u'@', u'JJ'), (u'iko_uwais', u'NN'), (u'You', u'PRP'), (u'are', u'VBP'), (u'an', u'DT'), (u'amazing', u'JJ'), (u'martial', u'JJ'), (u'artist', u'NN'), (u'Iko', u'NNP'), (u'The', u'DT'), (u'Raid', u'NNP'), (u'is', u'VBZ'), (u'one', u'CD'), (u'of', u'IN'), (u'the', u'DT'), (u'best', u'JJS'), (u'action', u'NN'), (u'films', u'NNS'), (u'I', u'PRP'), (u'have', u'VBP'), (u'seen', u'VBN'), (u'You', u'PRP'), (u"'re", u'VBP'), (u'very', u'RB'), (u'talented', u'JJ'), (u'theraid', u'NN')]

From the sentence, it can be seen that the noun is Action films.

Figure 9. List of adjective words from first 100 tweets

This is the list of sentiments extracted for those 100 sentences. These are extracted by showing only the word tagged by POS JJ, JJR, or JJS. For each tweet, one or more sentiments are extracted if found. These sentiments usually are adjective denoted. We found 233 adjectives from 100 sentences in this case tweets.

Precision and Recall

At precision and recall step we used 2339 tweets to be checked manually, annotated the sentiments in sentences. There are 2189 sentiments are selected. The system is run on the data and showed the result. According to which the precision and recall are calculated. From 2339 sentences, we decided 2031 tweets denoted as Tp (True positive), and it is followed by 158 tweets denoted as Fp (False negative). And 37 tweets were missing. They were expected as positive but missed, indicated as Fn (False image). If we put together these values in formals of precision and recall, we get the following result.

a. Precision:

$$P = \frac{Tp}{Tp+Fp} \quad (1)$$

$$P = \frac{2031}{2031+158}$$

$$P = = \frac{2031}{2189}$$

$$P = 0.9278209$$

b. Recall:

$$R = \frac{Tp}{Tp+Fn} \quad (2)$$

$$R = \frac{2031}{2031+37}$$

$$R = \frac{2031}{2069}$$

$$R = 0.9816336$$

c. F1 Score:

$$F1 = 2\frac{P*R}{P+R} \quad (3)$$

$$F1 = 2\frac{92*98}{92+98}$$

$$F1 = 94.90526$$

Testing

The system proposed is tested by running different test cases. Test cases will explain the performance of the system when it performs a task. It shows the actual result and the expected result. And also, pre-conditions and post-conditions of the test case will be given. The test case will show the priority low, medium or high. These test cases are showing the step of how the system is running to get a happy result.

Retrieve Tweets.

Test case ID: Input_2
Test Title: Retrieved tweets and details from Twitter.
Priority: High
Pre-condition: User has opened the system.
Post-condition: The tweets are retrieved, selected only the related tweets.

Table 1. Test Case Retrieves tweets

#	Steps	Expected Result	Actual Result
1	read auth key	Authorization	Authorized
2	Read given name	Read query	Show message reading
3	Search tweets	Search related tweets	Got tweets “from” and “toward” current name
4	Retrieve tweets	Tweets and the other attribute arranged in an array	Divide tweets per array
5	Save into json file	Create new json file	New json file named as person name

Table 1 list the test case that explains how the tweets retrieved per person and the actual result meets the expectation. It also explained the pre-condition which from a user has opened the system and postcondition for the test case able to retrieves only for the related tweets that saved. This test case is passed.

Arrange Tweet’s attributes.

Test case ID: Input_3
Test Title: Create a simple list of arrays.
Priority: Medium
Pre-condition: Tweets files are loaded.
Post-condition: The tweets array is simplified, a list of tweets is created for each person.

Table 2. Test Case Arrange tweet’s attributes

#	Steps	Expected Result	Actual Result
1	read json content	Read json string	json array detected
2	Format string	Replace null space by “,”	Replace null space by “,”
3	Deserialize object	String read as json list of arrays	String read as json list of arrays
4	get attributes that needed	Needed attributes called	Need attributes called
5	Clear symbol and ascii character	Removed symbol and ascii characters	Removed symbol and ascii characters
6	Repeat step 1 to 5	Repeat till all tweets selected	Repeat all tweets
7	Create simplified list	Create new list	Create new simplified list
8	Save new json file	Create new json file	Created new json file

The test case that listed in Table 2 explains the steps to simplify the tweet's attributes that we have got from the previous task. The pre-condition for this test case starts from loading the tweets files and end when the tweets list is simplified and saved to another JSON file. The test case showing we removed the symbol and ascii characters inside the tweets body text, and it shows the expected and the actual result for all the step too. It matched the postcondition and created the new JSON files also. Thus, the test case is passed.

Tokenize and sentiment analyze.

Test case ID: Input_4
Test Title: Tokenize and sentiment analyze the tweets.
Priority: High
Pre-condition: New tweets files are loaded.
Post-condition: All tweets got classified, and scored. Likes and dislike calculated

Table 3 lists the test case that explains how the tweets got sentiment analyzed and scored. The pre-condition is that new tweets files are loaded. The system loads them by calling the file per file, and tweet per tweet. The system checks the sentiment of tweets in a dictionary if sentiment exists it will be scored. And it will decide the tweet is positive or negative and likes and dislike calculated as postcondition of the test case. Thus, the test case is passed.

Position the list.

Test case ID: Input_5
Test Title: Position on the list.
Priority: High
Pre-condition: People likes dislikes json file loaded.
Post-condition: Five people most liked or dislike arranged.

Table 3. – Test Case tokenize and sentiment analyze

#	Steps	Expected Result	Actual Result
1	read people files	Read all person json files	Read all person json files
2	Call person file	Calls 1 by 1 Person file	Calls file per person
3	Retrieves person details from Wiki	Retrieves first paragraph from wiki	Show retrieving message
4	Call tweets	Load tweets 1 by 1	Load tweets 1 by 1
5	Tokenize tweet	Tokenize per sentence	Tokenize per sentence
6	Giving score	Giving score for each tweet	Giving score for each tweet
7	Clustering positive & negative tweets	Divides the positive and the negative tweets to different places	Showing saving positive and negatives tweets
8	Repeat step 1 to 7	Repeat steps until all tweets got scored	Repeat until all tweets got scored
9	Repeat step 1 to 8	Repeat until all files classified	Repeat until all files classified
10	Save each person likes, dislikes calculation	Save each person likes and dislikes calculation in json file	Create new json files

Table 4. Test Case position the list

#	Steps	Expected Result	Actual Result
1	read file	Read people record	Read people record
2	Call the list	Call the list of people	Call the list of people
3	Move the person who has more likes	Move the person who has more likes to the first of list	Moves the person who has more likes to the first of list
4	Repeat step 3	Repeat step 3 until the most likes are the first one in the list	Repeat step 3, the most liked person in first of list
5	Move the person who has more dislikes	Move the person who has more dislikes to the first of list	The list arranged well from the most dislikes to the less dislikes
6	Repeat step 5	Repeat step 5 until the most dislikes are the first one	Repeat step 5, the most dislikes person in the first

The above test case that listed in Table 4 explains step by step how to the list arranged. The pre-condition for this test case is people likes dislikes files are loaded. The system is looking for the person with more likes or dislikes and moves the position to the first of the list. That step will be repeated until all the person in the list moved and arranged. The test case meets the post-condition which is most likes and dislikes people arranged, thus test case is passed.

Visualizing detail and tweets.

Test case ID: Input_6
Test Title: Visualizing detail and tweets.
Priority: High
Pre-condition: List has visualized, a person has clicked.
Post-condition: Showing description, likes, dislikes, positive and negative tweets.

Table 5. – Test Case visualizing detail and tweets

#	Steps	Expected Result	Actual Result
1	Got the name	Clicked name received	Got the name
2	Read the list	Open the list	Read the list
3	Call a certain person	Call person from the list who has same name as name	Called same person as the name that received before
4	Shows desc, likes, dislikes.	Load the correct desc, likes, and dislikes.	Loaded the correct desc, likes, and dislikes.
5	Read positive and negative tweets	Read current person positive and negative tweets files	Read current person positive and negative files
6	Load positive and negative tweets	Loads positive and negatives tweets	Loaded positive and negative tweets

Table 5 lists the test case explains how the related tweets showed after the pre-condition. The pre-condition is that the list has visualized and a person's name has clicked. Then the name is sent to the second page, the second page opened the list and matched the name. The system shows the description, likes, dislikes, and read positive and negative tweets saved directory. That's how the system gets related tweets for the selected person. The test case matched all the expected result and met the post-condition which is shows description, likes, dislikes, positive and negative tweets. Thus, the test case is passed.

CONCLUSION

The people profiling and modeling reputation computation based on sentiment analysis is a system for monitoring reputation. It looks up people opinion and conversation about someone on Twitter and gives a score of it. It provides the ratings for each people based on tweets sentiments and arranges the list of these people from the most liked or the most disliked to be easy to understand. Let see the result and it also provides proof about how many likes and dislikes that particular person has according to people's tweets on twitter.

The people profiling and modeling reputation computation based on sentiment analysis system is running well and performing good by tested with some test case. The precision and the recall showing the good result either. But in the future, this system needs to be updated. The new library will be required as the original word will also keep growing. For now, system people profiling and modeling reputation computation based on sentiment analysis only run by crawling the tweets that saved to the local directory. Twitter cannot be accessed without a VPN in China, and it became a problem in building this system. Hence, in the future people, profiling and modeling reputation computation based on sentiment analysis will retrieve the tweets as real time, so able to monitoring and updating the list while the program keeps getting the new datasets which are the tweets.

REFERENCES

Amigo E., Albornoz J., Chugur I., Coruju A., Gonzalo J., Meij E., Rijke M., Spina D. (2014). Overview of Replab 2014: Author Profiling and Reputation Dimensions for Online Reputation Management. In *Workshop Proceeding of CLED 2014 Working Notes*, *1180*, Sheffield, UK. (pp. 1438-1457).

Amigo, E., Carrillo De Albornoz, J., Chugur, I., Coruju, A., Gonzalo, J., Martin, T., Meij, E., De Rijke, M., Spina, D. (2013). Overview of Replab 2013: Evaluating Online Reputation Monitoring System. In *Proceeding of the Fourth International Conference of the CLEF Initiative*. LNCS, Springer. (pp. 333-352).

Cambria, E. & White, B. (2014). Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article]. *IEEE Computational Intelligence Magazine, 9*(2), 48-57. http://doi.org/10.1109/MCI.2014.2307227

Cann A. (2011). *Social Media: A Guide for Researchers*. Research Information Network. UK.

Cossu, J., Bigot, B., Bonnefoy, L., Morchid, M., Bost, X., Senay, G., Dufour, R., Bouview, V., Torres-Moreno, J., El-Beze, M. (2013). LIA@Replab 2013. In *CLEF 2013 Evaluation Labs and Workshop Online Working Notes*. (pp.1-13)

Cossu, JV, Bigot, B. Bonnefoy, L. & Senay, G. (2014). Towards the Improvement of Topic Priority Assignment Using Various Topic Detection Methods For E-Reputation Monitoring on Twitter. In *International Conference on Applications of Natural Language to Data Bases/Information Systems*. (pp. 154-159). http://doi.org/10.1007/978-3-319-07983-7_20

Gulla. J.A., Fidjestol. A.D., Su. X. & Castejon. H. (2014). Implicit User Profiling in News Recommender System. In *International Conference on Web System and Technologies 2013*. Barcelona, Spain.

Hussein, D.M.E.M. (2018). A survey on sentiment analysis challenges, *Journal of King Saud University - Engineering Sciences*, *30*(4), 330-338. http://doi.org/10.1016/j.jksues.2016.04.002

Lian, S., Zhang, Z., Ren, Z. & Kanoulas, E. (2018). Dynamic Embeddings for User Profiling in Twitter. In Proceeding of the 24th ACM SIGKDD International Conference on Discovery & Data Mining. London, UK. (pp. 1764-1773). http://doi.org/10.1145/3219819.3220043

Ozturk, N. & Ayvaz, S. (2018). Sentiment analysis on Twitter: A text mining approach to the Syrian refugee crisis, *Telematics and Informatics*, *35*(1), 136-147. https://doi.org/10.1016/j.tele.2017.10.006

Qureshi M.A., O'Riordan C., Pasi G. (2014) Exploiting Wikipedia for Entity Name Disambiguation in Tweets. In *Métais E., Roche M., Teisseire M. (eds) Natural Language Processing and Information Systems. NLDB 2014. Lecture Notes in Computer Science*, *8455*. Springer, Cham. http://doi.org/10.1007/978-3-319-07983-7_25

Ranjan, S., Sood, S. and Verma, V. (2018). Twitter Sentiment Analysis of Real-Time Customer Experience Feedback for Predicting Growth of Indian Telecom Companies. In *2018 4th International Conference on Computing Sciences (ICCS)*, Jalandhar, India. (pp. 166-174). http://doi.org/10.1109/ICCS.2018.00035

Tavakolifard, M., Gulla, J. A., Almeroth, K. C., Ingvaldsen, J. E., Nygreen, G., & Berg, E. (2013). Tailored News in The Palm of Tour Hand: A Multi-Perspective Transparent Approach to News Recommendation. In *Proceeding of the 22nd International Conference on World Wide Web*. Rio de Janeiro, Brazil. (pp. 305-308).

Yu, K., Zhang B., Zhu H., Cao H., Tian J. (2012) Towards Personalized Context-Aware Recommendation by Mining Context Logs through Topic Models. In: *Tan PN., Chawla S., Ho C.K., Bailey J. (Eds.) Advances in Knowledge Discovery and Data Mining. PAKDD 2012. Lecture Notes in Computer Science*, *7301*. Springer, Berlin, Heidelberg. http://doi.org/10.1007/978-3-642-30217-6_36