The current issue and full text archive of this journal is available at
www.emeraldinsight.com/1744-0084.htm

Twitter data collecting tool with rule-based filtering and analysis module

Changhyun Byun, Hyeoncheol Lee, Yanggon Kim and Kwangmi Ko Kim
Computers and Information Sciences, Towson University, Towson, Maryland, USA

Abstract

Purpose – It is difficult to build our own social data set because data in social media is generally too vast and noisy. The aim of this study is to specify design and implementation details of the Twitter data collecting tool with a rule-based filtering module. Additionally, the paper aims to see how people communicate with each other through social networks in a case study with rule-based analysis.

Design/methodology/approach – The authors developed a java-based data gathering tool with a rule-based filtering module for collecting data from Twitter. This paper introduces the design specifications and explain the implementation details of the Twitter Data Collecting Tool with detailed Unified Modeling Language (UML) diagrams. The Model View Controller (MVC) framework is applied in this system to support various types of user interfaces.

Findings – The Twitter Data Collecting Tool is able to gather a huge amount of data from Twitter and filter the data with modest rules for complex logic. This case study shows that a historical event creates buzz on Twitter and people's interests on the event are reflected in their Twitter activity.

Research limitations/implications – Applying data-mining techniques to the social network data has so much potential. A possible improvement to the Twitter Data Collecting Tool would be an adaptation of a built-in data-mining module.

Originality/value – This paper focuses on designing a system handling massive amounts of Twitter Data. This is the first approach to embed a rule engine for filtering and analyzing social data. This paper will be valuable to those who may want to build their own Twitter dataset, apply customized filtering options to get rid of unnecessary, noisy data, and analyze social data to discover new knowledge.

Keywords Twitter, Crawling, Data-mining, Social analysis, Super Bowl 2012, Rule engine, Social networks, Social media

Paper type Research paper

1. Introduction

In recent years, with the increasing popularity of diverse online social network sites, such as Facebook, Twitter, Blogger, LinkedIn, and MySpace, a massive amount of data has become available. Technology and the internet allow users of social media to access and share data frequently and rapidly. Particularly, Twitter has become one of the fastest growing social media sites. Since its launch on March 21, 2006, Twitter has reached a user population above 500 million total users and more than 200 million active users in 2012. Additionally, the number of messages that Twitter users exchanged per day has increased from 300,000 in 2008 to 340 million in 2012.

People post their messages on social network sites for a variety of purposes, such as sharing information, conversations, updating real-time statuses, reporting the news, and

International Journal of Web Information Systems
Vol. 9 No. 3, 2013
pp. 184-203

1744-0084
DOI 10.1108/IJWIS-04-2013-0011

expressing opinions. In addition, various business companies have made substantial efforts to accommodate such swift trends and have paid attention to the competitive advantages of using social media in marketing. Analyzing sets of data in social media can lead to some understanding of individual and human behavior, detection of hot topics, identification of influential people, or discovery of a group or community. However, it is difficult to discover useful information from social data without automated information processing because of three main characteristics of social media data sets: the data is large, noisy, and dynamic. In order to overcome these challenges of social media, data-mining techniques can be used by data seekers to discover a diversity of perspectives that would otherwise not be possible. Data-mining techniques are widely used to handle large sets of data and to discover new knowledge and useful information in a data set that is not readily obtainable and not always easily detectable. Hence, applying data-mining techniques to online social media benefits many groups, such as market researchers, psychologists, sociologists, businesses, and politicians, fascinating insights into human behavior, marketing, business or political views (Cortizo *et al.*, 2011; Sottara *et al.*, 2011; Al-Khalifa, 2012).

To apply data-mining techniques to social data, the target data set must be prepared from social networks before the analyzing process. For these reasons, Twitter enables researchers and data analyzers to access a variety of data in Twitter by providing application programming interface (API). Numerals of researchers collected Twitter data through Twitter APIs and applied data-mining techniques into their own data set to detect issues, such as earthquakes (Okazaki and Matsuo, 2010) and influenza by using Twitter (Aramaki *et al.*, 2011) or recommending tags to users (Correa and Sureka, 2011). However, there is a restriction on data collection from Twitter: the method call of Twitter API is limited by 350 calls per hour for one authorized developer account (Twitter, 2013). Furthermore, it is impossible to collect enough data to apply data analysis techniques and filter out unnecessary data, such as spam messages without an automated data collector and filter. In order to overcome these data access problems, we aim to design and implement our own Twitter data-collecting tool, which includes data filtering and analysis capabilities. This allows us, as well as other researchers and data seekers, to build their own Twitter dataset.

In this research, we will introduce the design specifications and explain the implementation details of the Twitter data collecting tool we developed in Section 3. In Section 4, we provide an analysis of Twitter data gathered by the Twitter data collecting tool in a case study about the super bowl. The case study aims to address the question of how people use Twitter and to assess the power of Twitter in creating consumer interest in brands and commercials. The main objective of this study is to find the relationship between Twitter and Super Bowl advertisements by analyzing data on Twitter. The result of this data analysis can be used for various purposes as well as the following research questions: how were the conversations about Super Bowl advertising developed? How positive/negative are tweets about Super Bowl advertising?

The remainder of this paper is constructed as follows: in Section 2, the related work done so far is summarized. Section 3 introduces the design specifications and explains implementation details of our data collecting and filtering system. Section 4 presents results of the data collection and analysis of Super Bowl 2012. The last part, Section 5, concludes the work by summarizing this paper and suggesting future research directions.

2. Related works

In this section, we introduce the key concepts, terminology, and methodologies that are used in this research.

2.1 Data crawling tools in Twitter

2.1.1 TwitterEcho-opensource Twitter crawler. Bošnjak *et al.* presented an open source crawler, TwitterEcho, which is used to retrieve data from Twitter (Twitter, 2013; Bošnjak *et al.*, 2012). It allows data seekers to collect data from a focused community of interest. TwitterEcho adapts a centralized distributed architecture and includes three main components: clients, servers, and modules. Figure 1 shows the architecture and main components of the TwitterEcho Crawler.

A client consists of two modules. The first module collects tweets, user profiles, and simple statistics. The second module collects social network relations. The number of clients can be increased to retrieve more data from Twitter. The server manages the crawling process and allocation of user lists to each client. It also maintains the database in which the downloaded data is saved. Modules consist of user expansion, user selection, and user inspection. The user expansion module analyzes downloaded tweets, extracts the user's mentions, and adds the user's followers to the list of tentative users. The user selection module identifies users' accounts to be monitored by analyzing

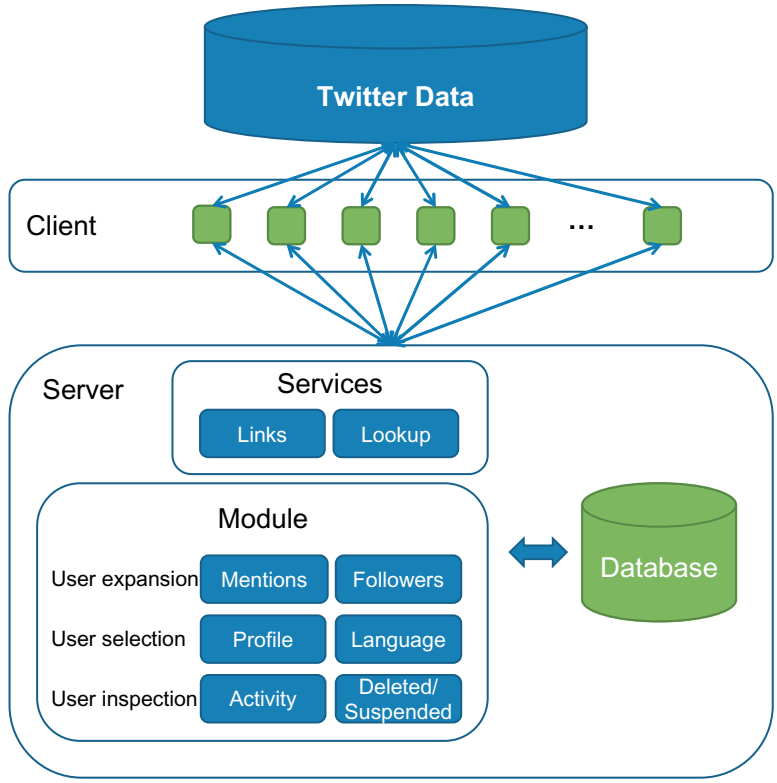

Figure 1.
Architecture of
Twitter Echo

profiles and identifying languages. The user inspection module also monitors events, such as deletion, suspension, and the activity of users' accounts.

Despite the fact that TwitterEcho is able to gather a huge amount of data from Twitter, there are still some problems with the program. First of all, it would be more efficient to retrieve data for a focused community of interest, starting with multiple seeds. However, the program does not support this function. Second, TwitterEcho starts with a seed node and keeps inspecting its followers and their followers' data. Since TwitterEcho does not restrict the level of followers, it is likely to have an enormous amount of noisy data. Finally, retrieving data is restricted to Portuguese only. Even though we can adapt additional modules to TwitterEcho for a specific area, it needs time finding or implementing these additional modules.

2.1.2 Twitter user data using resource of cloud. Another approach to collecting Twitter data is the use of the computation power of cloud computing (Noordhuis *et al.*, 2010). Noordihuis *et al.* gathered Twitter data and applied the PageRank algorithm to rank Twitter users using the computation power of cloud computing.

There were five steps in this cloud system. First, a queue and table are set up to maintain all user IDs that need to be crawled. Then, users' and followers' IDs are saved in the SimpleDB, which is a service for storing structured data in the cloud. Furthermore, different users' information is gathered for different instances simultaneously by using their own web service. In the fourth step, the PageRank Algorithm is applied to rank users. Finally, a web interface enables public users to access their data. As a result, 50 million users and 1.8 billion followers' information were crawled and Twitter users were analyzed using the PageRank algorithm.

Even though they showed cloud services are feasible to gather huge amounts of social data, there are also some problems with their approach. First, they did not save gathered data into local storage because the additional usage of storage would be costly. Moreover, whitelist accounts were used to gather Twitter data through Twitter API. However, due to current Twitter policy, getting a new whitelist account is not feasible. Furthermore, it is not suitable for some specific research areas, such as data-mining. Because they crawled all user data from Twitter, there is a lot of noisy data that many researchers are not interested in. For that reason, it would consume a lot of resources to analyze and filter this data. Additionally, they did not gather tweets data that can be used for analyzing users' tendencies and opinions.

2.1.3 Crawling Twitter data using whitelist accounts. Kwak *et al.* gathered Twitter data to study the topological characteristics of Twitter and its power for information sharing (Kwak *et al.*, 2010). By using Twitter APIs, they gathered all users' profiles, trending topics, and tweets that mentioned the trending topics. As a result, they successfully crawled the entire Twitter site, including 41.7 million user profiles, 1.47 billion social relations, 4,262 trending topics, and 106 million tweets. However, they used whitelist accounts that are useful to gather huge amounts of data from Twitter but are no longer available. Also, Twitter is not suitable for some research areas as mentioned in 2.2.

2.2 Rule-based engine

Related works described in the previous section show that there is a huge amount of noisy data that data analyzers are not interested in. Furthermore, analyzing sentiment in social networks has been issued in data-mining of social networks. To solve these

problems, we embedded a rule-based engine in the Twitter Data Collecting Tool to filter the data and analyze the sentiment of the messages.

A rule is a syntactic construct that is used to check if certain consequences satisfy the premises (Sottara *et al.*, 2010). A rule engine, also known as an inference engine, is a computer program which analyzes patterns by comparing facts and rules, and executes actions declared in the rules. The rule engine can be embedded in the Twitter Data Collecting Tool to filter data. There are many kinds of rule engines available. Byun *et al.* studied java-based rule engines (Byun *et al.*, 2011). Among the free and java-based rule engines, Drools (JBoss, 2013) and Jess (Friedman, 2013) have been studied in the research. The result is described in the following table. Table I gives comparison for the selected inference engines measured on different performance metrics.

We have chosen the Drools engine to filter and analyze tweets in the Twitter Data Collecting Tool. This is because it is a java-based open-source, which offers different semantics for encoding rules based on the programming language used for expressing conditions and actions, and has a syntax that favors readability.

2.3 Methodologies of analyzing sentiment of messages in social networks

Analyzing sentiment of messages in social networks has been studied in various ways. Dey and Haque proposed a localized linguistic approach to extract opinion expressions from noisy text that are generated from online chat, emails, blogs, customer feedback, and reviews (Dey and Haque, 2008). Based on the pre-processed noisy texts, they analyzed opinion expressions using a classifier algorithm and candidate opinion words from Wordnet. Hu and Liu suggested a technique to identify opinion sentences in each review about product features and to decide whether each opinion sentence is positive and negative (Hu and Liu, 2004). They determine its semantic orientation using a set of adjective words, which are identified by a natural language processing. O'Connor *et al.* presented an approach to polarity classification by counting the number of positive and negative expressions in a tweet and selecting the category with more terms (O'Connor *et al.*, 2010) Label propagation using graph-based methods can also be used to find opinion from social network sites (Zhu and Ghahramani, 2002; Baluja *et al.*, 2008; Talukdar and Crammer, 2009). Choi and Cardie used domain specific lexicon and relations among words and opinion expressions to classify polarity of messages in a social network site (Choi and Cardie, 2009). However, the accuracy for polarity classification still needs to be improved as the accuracy in other research shows polarity classification of 80 percent.

Table I.
A comparison of rule engines

Category	Rule engine: Drools	Rule engine: Jess
Algorithm	RETE algorithm	RETE algorithm
OWL-DL entailment	No	Yes
Java support	Yes	Yes
Rule support	DRL (own rule format)	SWRL
Version	5.0	7.1
Licensing	Free/open source	Academic use only

3. The Twitter data collecting tool

Faced with limited options, we designed and developed our own Twitter Data Collecting Tool. In this section, we present requirements and architecture of this system.

3.1 System requirements

Problems that we discussed in the previous section encouraged us to define the following requirements for the Twitter Data Collecting Tool:

- *Continuously and automatically collects data from Twitter.* Once seed IDs and authorized accounts are configured, the tool must be able to collect tweets and relations related without any human interaction until the stop event has occurred or the target amount of data is collected.
- *Runs with multiple seed nodes.* In the gathering process, users may want to start the data collecting process from certain Twitter users. To fulfill this need, the system should be able to accept multiple seed nodes.
- *Handles a multitude of authorized keys.* The tool must be able to handle more than one authorized key. This increases the total number of Twitter API calls.
- *Stores collected data in a database.* The system has to save all collected data, such as users' follower/following relations and tweets, into a database.
- *Supports intuitive user interface.* The tool must support user interface to interact with users. A straightforward and intuitive interface must be provided to start and manipulate the program.

3.2 Architecture

Figure 2 shows you the constituent controllers of the Data Collecting Tool. The given data collector consists of the Account Handler, the Data Gathering Controller, the Database Handler, Data Gatherer Thread Pool, as well as, the Data Filtering Handler.

3.2.1 Account handler. The Account Handler is in charge of managing the Twitter developer's account. As long as Twitter limits the number of method calls for one developer's account to 350 calls an hour, a program feature to handle multiple developers' accounts is necessary to build a large database. The main role of the handler is to provide the Data Gathering Controller with current available resources consistently by managing developers' account objects in a wait queue and a ready queue. When a developer's account hits its limit of method calls, the module moves the account object to the wait queue and keeps it there until the limit is recharged.

3.2.2 Data gathering controller. The Data Gathering Controller acts as a backbone of the tool. It receives available developer's account objects from the Developer Account Controller module, and then sends a request to the Data Gatherer threads with specific requirements, such as gathering seed-related users' information or gathering certain users' tweets. When the threads return with results to the Data Gathering Controller, it passes the data to the Database Handler to save the information in a database.

3.2.3 Database handler. The Database Handler is required in the data-collecting tool to store data gathered in the database and to retrieve stored data back, as well. The Database Handler manipulates multiple database connection objects using a connection pool technique. The handler is designed to endure massive update transactions streamed from the Data Gathering Controller.

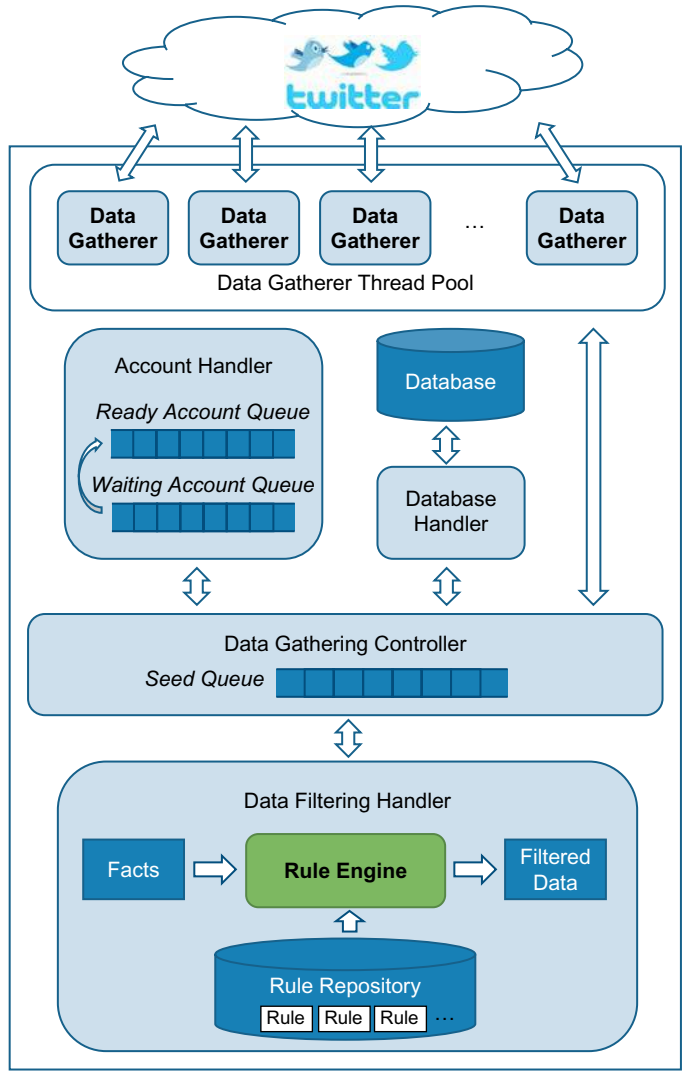

Figure 2.
Architecture of the twitter data collecting tool

3.2.4 Data gatherer thread pool. Each thread has its own data gathering rules to collect users' account information and users' tweets or follower relationships. The gathering rules also contain data filtering options to sift noisy data from the collected data. To run each thread, at least one developer account should be received from the Data Gathering Controller.

3.2.5 Data filtering handler. The Data Filtering Handler is in charge of both filtering and analyzing data. We imbedded an open source rule engine, the Drools rule engine, in the Twitter Data Collecting Tool for these purposes. Once a user enters a keyword in the tool through the user interface, the Data Filtering Handler generates rules to filter data and save the rules in a rule repository. Then, the rule engine generates new

knowledge through pattern matching process and then outputs the filtered data. For the data analysis process, the Data Filtering Handler reads predefined rules from the rule repository. Then, user profiles and tweets are analyzed by the rule engine. Particularly, the rule repository contains predefined rules to analyze tweets and user profiles. For instance, one of data analysis algorithms to discover a user location from a user profile is defined as one of predefined rules in the repository.

3.2.6 Database architecture. We designed and implemented a database to store and handle plenty of data easily and effectively. To reflect following/follower relationships between seed nodes and follower nodes, three tables, tbl_seed table, tbl_relationship table and tbl_followers table, are defined. A combination of the seed ID and user ID can identify each of the following relations. The user table, tbl_user table, is designed for storing the user's detailed information, such as screen name, first name, last name, location, etc. To save data about tweets, a table, called tbl_tweets table, is defined as well. The tbl_tweets table contains information about tweet ID, user ID, actual message, and date posted. A relationship between user and tweet is formed to track the message creator and its seed.

3.3 Design of the Twitter data collecting tool

To introduce and explain the implementation details and the design specifications of the Twitter Data Collecting Tool, the Unified Modeling Language (UML) diagram is used. Two of the major UML diagrams, class diagram and sequence diagram, are shown in Figures 3 and 4. The system provides two different types of user interfaces, Graphical User Interface (GUI) and Command Line Interface (CLI). These two user interfaces are shown in Figures 6 and 7 as well.

3.3.1 Object specification. The Model View Controller (MVC) framework is applied to this system to support various types of user interfaces, such as a standalone interface and a web-based interface. Followed by the MVC framework, objects of the system are divided into three packages: model, view and controller, as shown in Figure 3.

Only interface-related objects are defined in the View package. Three different types of user interfaces, the CommandUserInterface, SwingGraphicalUserInterface and WebUserInterface, realize the UserInterface interface so that the system is able to

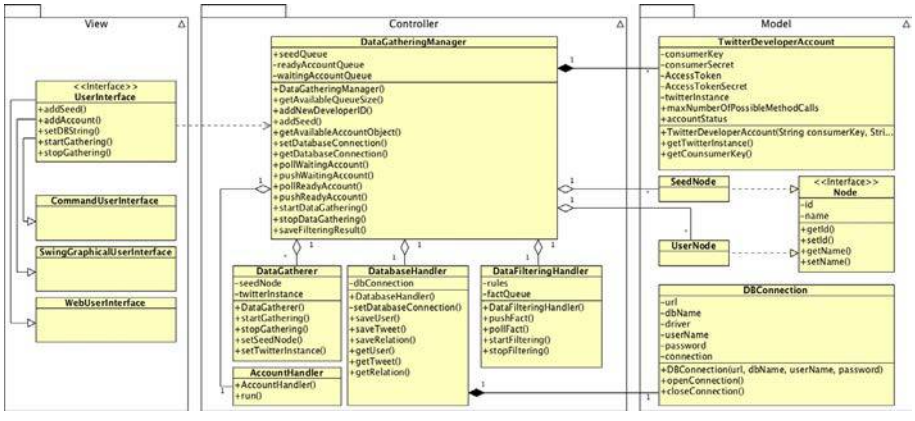

Figure 3.
Class diagram of the twitter data collecting tool

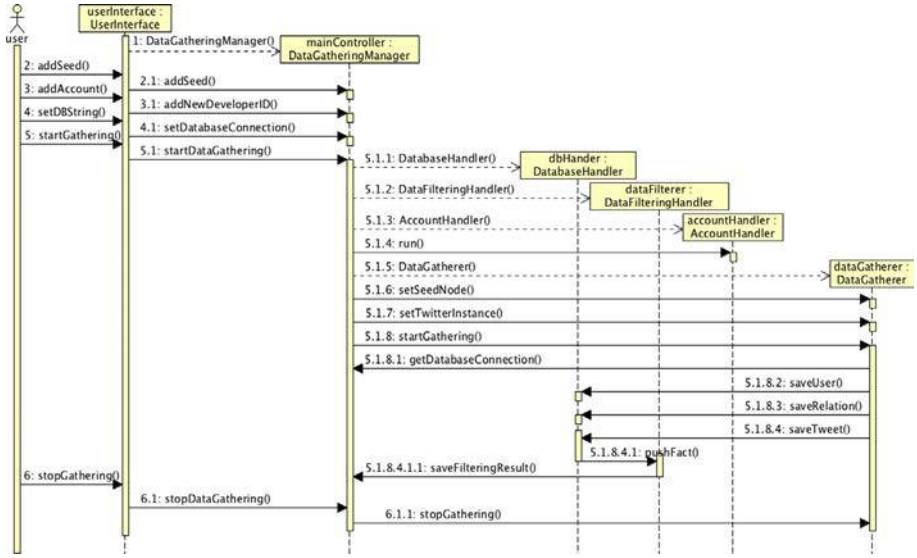

Figure 4.
Sequence diagram of data collection from twitter using the twitter data collecting tool

provide various interfaces to a user, as well as a new type of interface that can be defined and realized easily.

The controller package consists of a DataGatheringManager class, a DataGatherer class, an AccountHandler class, a DatabaseHandler class, and a DataFilteringHandler class. The DataGatheringManager acts as the backbone of the system and has duties to transform messages from a user interface to a handler and to control all handlers in the controller package. The DataGatheringManager class receives information about Twitter accounts, seed IDs, and filtering rules and keywords from one of user interfaces using public operations, such as addSeed() and addNewDeveloperID(). The DataGatherer class is supposed to connect to the Twitter database and collect data through Twitter API. The DatabaseHandler class takes responsibilities to store data into a local database and to retrieve stored data from the local database. The AccountHandler class is in charge of checking if a Twitter account exceeds its limits. If the account has hit its maximum allowed method calls, the AccountHandler class pushes the account into the waiting queue. If the account has not hit the limits the AccountHandler class pushes the account into the ready queue so that the DataGatherer class can always get an available account instance from ready queue. The DataFilteringHandler class is keeping an eye on the fact queue. If there is any case filtered by the rule engine, the DataFilteringHandler class sends the DataGatheringManager class the matched data.

The model package contains four basic classes and one interface. The four basic classes, the TwitterDeveloperAccount class, the SeedNode class, the UserNode class, and the DBConnection class, are defined as message types. The SeedNode and UserNode are implementations of the Node interface.

3.3.2 Sequence of data collecting process. In this section, we explain the process of data gathering and filtering in the Twitter Data Collecting Tool. A sequence diagram depicts the entire process in Figure 4. First, the process starts with a user's input from a

user interface. A user enters seed node IDs, Twitter developer account information, and filtering keywords in the user interface. Once the user enters the input, it invokes the DataGatheringManager class. Then, the DataGatheringManager class executes the DatabaseHandler class, the DataFilteringHandler class, the AccountHandler class to set up a database connection and options for gathering and filtering. Based on the connection and gathering options, the DataGatherer class gathers users, relation and tweets data from Twitter and save them in a designated database. Finally, the DataFilteringHandler class filters the tweets using the keywords. The gathering process ends when the user calls stopGathering() method from user interface.

3.3.3 Sequence of data analysis process. In this section, we explain the process of data analysis in the Twitter Data Collecting Tool. A sequence diagram depicts the entire process in Figure 5. The data analysis process starts when a user triggers the analyzing process through a user interface. Once the process is started, the DataGatheringManager class executes the DatabaseHandler class and the DataFilteringHandler to analyze data stored in a database through a rule-based filtering engine. When the DataFilteringHandler is executed for the data analysis purpose, it loads predefined rules from a rule repository. For instance, the rule repository contains predefined rules for analyzing user profiles or tweets. After the DataFilteringHandler class loads whole rules, it collects tweets and user profiles from database to analyze them through rules. Finally, the DataFilteringHandler class sends the DataGatheringManager class results of data analysis. The data analysis process ends when all tweets and user profiles are analyzed.

3.3.4 User interfaces. We implemented GUI of the Twitter Data Collecting Tool as shown in Figure 6. It is designed for users to be able to enter filtering and gathering options easily. On the left side of the basic option tab, users can add or remove seed IDs by clicking the Add or Remove button. On the right side, users can add or remove Twitter developers' account information. In this menu, users can add multiple Twitter developers' account information to gather the data faster. Moreover, the Twitter Data Collecting tool also provides a keyword-filtering function. In the Filtering Option tab, users can add multiple filtering keywords. Based on the keywords that users entered, the Twitter Data Collecting tool generates rules to filter the gathered data using the keywords. At the bottom of the interface, three buttons are given to manipulate the tool, which are Start Analysis, Start Gathering and Stop Gathering buttons.

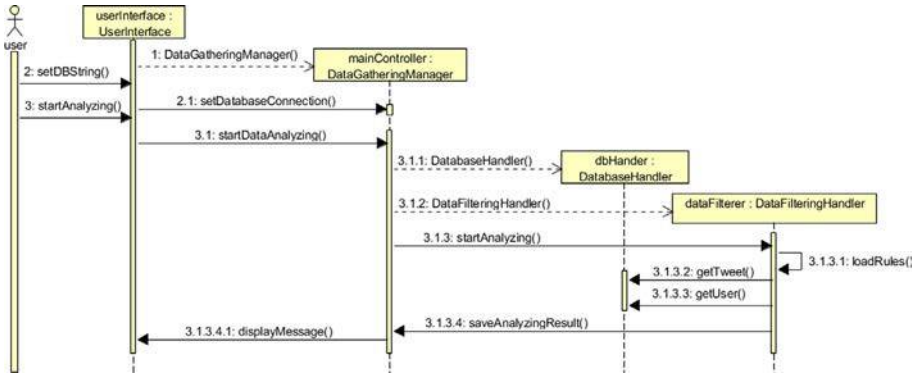

Figure 5.
Sequence diagram of data analysis process in the twitter data collecting tool

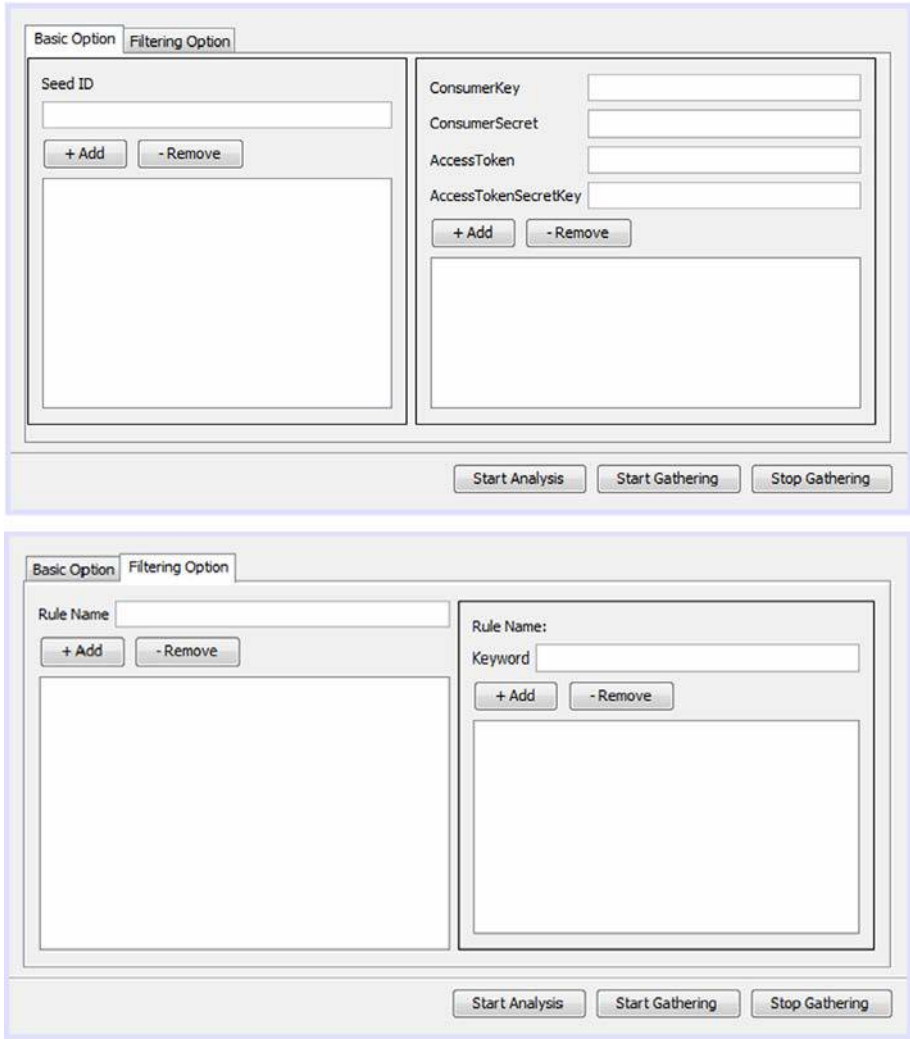

Figure 6.
GUI of the twitter data collecting tool

4. Case study

In this section, we gather Twitter data using the Twitter Data Collecting Tool we developed and analyze the data using a data-mining algorithm and tools.

4.1 Data gathering plan

We planned to analyze the effects of Super Bowl 2012 commercials on the preference of car manufacturers that advertised their commercials during the event. To gather the data about car companies and Super Bowl commercials, we decided to gather data from January 30 to February 15, which includes one week before and two weeks after the Super Bowl game. This is necessary to track the trends of messages about Super Bowl advertising since marketers released their commercials to social media sites

(e.g. YouTube) prior to the actual broadcast of the game, hoping to create more buzz and interest from consumers. Also, we found the 11 car-related companies.

4.2 Data gathering execution

Twitter data will be collected and filtered through 8 steps: selecting a seed, finding a Twitter account for each selected seed, gathering all followers' IDs for each seed node, picking random followers for each seed node, gathering tweets from picked followers, gathering profile information of each picked follower, saving all retrieved data into a local database, and filtering tweets with an imbedded rule-based engine. The detailed data handling process is as follows:

(1) *Select seeds.* As we mentioned in the introduction, 11 car-related companies were selected – Volkswagen, Toyota, Kia, Hyundai, Honda, Chrysler, Cadillac, Acura, Lexus, Chevrolet, and Audi.

(2) *Find Twitter IDs for all selected seeds.* To start gathering data from selected seed nodes, Twitter IDs for each seed node were identified. Twitter API provided a method to lookup a user's ID using a user's screen name or first or last name.

(3) *Retrieve all followers' IDs for seed nodes.* Using the Twitter data-collecting tool, find and retrieved all followers' IDs for seed nodes.

(4) *Pick 1,500 followers randomly for each seed.* We picked followers randomly to generalize the case.

(5) *Gather tweets of randomly picked followers, posted between January 1, 2012 and April 29, 2012.* The Super Bowl was held on February 5, 2012. However, gathering data from before the game created a baseline to compare before and after findings. To see maintainability of Super Bowl commercial effects, data after the big game might be required.

(6) *Gather personal information of picked followers.* To determine a user's preference, the user's personal information was collected. If the user allowed the developer to gather their personal information, we can gather the user's name, location, number of followers, number of followings, and even date account created.

(7) *Save all gathered data into a database.* We saved all data gathered into database we designed in Section 3.2.5.

(8) *Filter tweets using an embedded rule engine.* We filtered tweets using Super Bowl Commercials related keywords and Drools, which are embedded in the Twitter Data Collecting Tool. Also, we filtered the tweets again using positive and negative keywords to rank the 11 car-related companies.

4.3 Results of data collecting from Twitter

Table II shows the number of data for each category and the total number of data we gathered using the Twitter data-collecting tool we developed. As we selected 11 car-related companies, 11 rows of seed nodes are gathered and stored in the database. About 1.1 million following relationships between seeds and user nodes are gathered, and all users' 1 million user IDs are collected without data duplication. From 1,500 selected users for each seed nodes, 16,500 selected users in total, about 5.1 million tweets were retrieved from Twitter. Including 16,500 selected users, 88,121 of users had their information stored in the database.

4.4 Data filtering and analysis

Instead of using a traditional content analysis, this study used data-mining techniques and tools that allowed us to analyze massive amounts of messages about cars to derive useful knowledge.

4.4.1 Relevance between Twitter and traditional media. In the first phase, we analyzed the relevance between tweets about the car-related companies and Super Bowl commercials. First, we filtered tweets that are related to Super Bowl car commercials using keywords and rules. The keywords are based on the name of companies that aired commercials during the Super Bowl and actors' and car names from the commercials. This is because users mostly mentioned the companies, cars, actors, or characters from the commercials in their tweets. For this reason, we made a rule that if a tweet includes at least one of the keywords, it is a tweet about the company. Table III shows the keywords for each company.

Based on the number of tweets filtered by keywords and the embedded rule engine, we visualized the tweets using WEKA (Hall *et al.*, 2009), as shown in Figure 7. However, there are a lot of noisy data that are unrelated to the Super Bowl Commercials in the data. For that reason, we filtered the data again using the rule engine and Super Bowl Commercial related keywords, such as Super Bowl, Commercial, and ads. The filtered data is visualized again as shown in Figure 8.

The number of tweets that are related to car companies and their commercials during the Super Bowl is higher than other times, which means the Super Bowl commercials created buzz on Twitter and many Twitter users were interested in them.

4.4.2 User sentiments in tweets. In the second phase, we inferred the meaning of the tweets about Super Bowl commercial using the rule engine. For this experiment,

Table II.
Collected Twitter data results

Data type	Rows
The number of seed nodes	11
The number of following relationships between seed nodes and followers	1,134,798
The number of follower IDs for all seed nodes	968,641
The number of tweets of selected users	5,157,887
The number of personal information of selected users	88,121
Total	7,349,758

Table III.
Company name and keywords to filter tweets about the commercials

Company name	Keyword
Chevrolet	*Chevrolet*, *Chevy *
Chrysler	*Chrysler*, *Clientwood*
Hyundai	*Hyundai*, *Veloster*, *Genesis C*
Audi	Audi *, * Audi *
Acura	*Acura*, *NSX*
Toyota	*Toyota*, *Camry*
Lexus	*Lexus*
Cadillac	*Cadillac*
Kia	*Kia*, *Optima*
Honda	*Honda*, *CR-V*, *Matthew Broderick*
Volkswagen	*Volkswagen*, *new beetle*

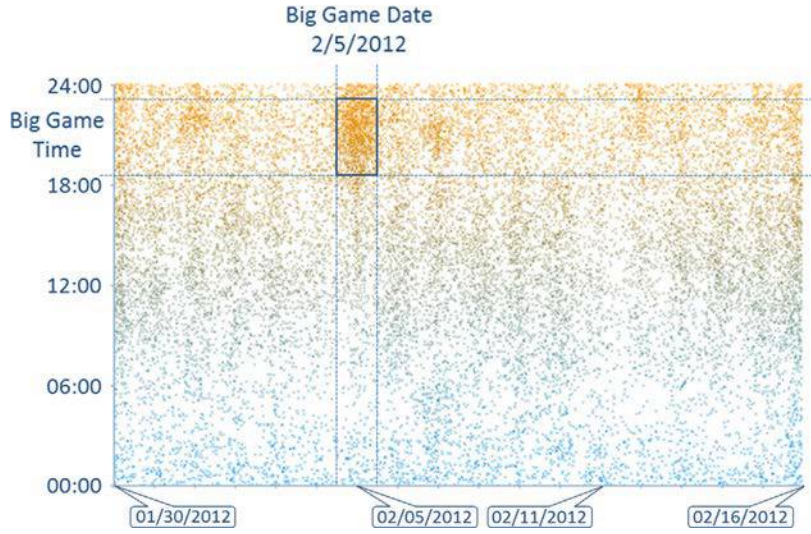

Figure 7.
The increased number of tweets during the super bowl before filtering

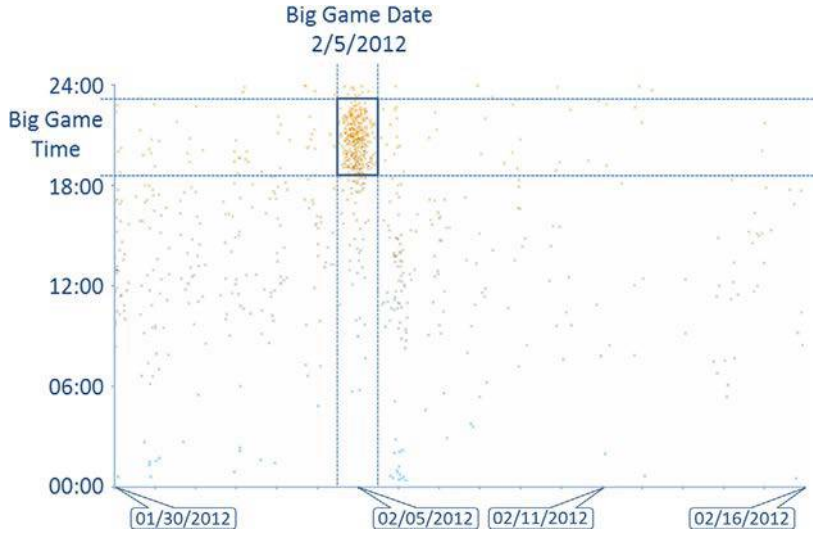

Figure 8.
The increased number of tweets during the super bowl after filtering

we retrieved tweets about Super Bowl 2012 commercials as we did in the first phase. Then, we derived users' sentiments by analyzing the tweets. User's sentiments are categorized into four different groups, which are positive, negative, positive-negative-mixed, and neutral sentiments. If a user's tweet contains positive words only or contains both positive and negative words, but has more positive words than negative words, then the tweet is categorized into the positive group. The concept of categorizing negative tweets is the same as categorizing positive tweets, but it focuses on the number of negative keywords in each tweet instead. Occasionally, users post tweets with both

positive and negative words equally. In this case, if a tweet includes both positive and negative words and the amount of each sentiment is equal, the tweet is characterized as a positive-negative-mixed tweet. Otherwise, if there is no word reflecting a user's positive or negative opinion in a tweet, it is classified as a neutral tweet.

The categorizing user's sentiments process is implemented in the rule-based module as a rule. A rule for Categorizing User's Sentiments shows a structure of the rule and partial positive and negative words we used:

```
rule "Positive"
   when
      t:Tweet (tweet matches ".*[a,A]mazing.*") or
         ...
      t:Tweet (tweet matches "[t,T]op.*") or
      t:Tweet (tweet matches "hot")
   then
      t.setPositiveValue(t.getPositiveValue() + 1);
end
rule "Negative"
   when
      t:Tweet (tweet matches ".*[a,A]mbiguous.*") or
         ...
      t:Tweet (tweet matches ".*[p,P]oor.*") or
      t:Tweet (tweet matches ".*[t,T]arrible.*") or
   then
      t.setNegativeValue(t.getNegativeValue() + 1);
end
```

Among 6,457 words, 2,304 words annotated as positive and 4,153 as negative, used in the OpinionFinder subjectivity lexicon, due to our specific dataset, we selected 72 positive words, such as "Amazing," "Awesome,", etc. and 52 negative words are selected for the negative group, such as "Hate," "Horrible,", etc. The rule, categorizing user's sentiments, consists of two sub-rules: one is counting positive words in a tweet and another one is counting negative words in the tweet. Once a tweet is forwarded to this counting words rule, both the number of positive words and the number of negative words are contended and stored in the tweet object.

To ensure the reliability of the algorithm, we investigated the accuracy of it by comparing the data set results categorized by a human and the other data set results categorized by our algorithm. 1,500 tweets are randomly selected from the whole dataset and are coded by two coders who are not involved in the classification algorithm process. After these 1,500 tweets are coded into the four categories, the result of the classification by our algorithm is compared with that of human analysis to see the accuracy of our classification algorithm. Table IV shows the accuracies of our rule-based algorithm compared to categorization result by human. The results indicate that the accuracy of the rule-based approach has an acceptable rate (more than 80 percent). This means that our sentiment categorization algorithm is reliable for analyzing user's sentiments in their tweet especially tweets about Super Bowl advertisement.

Figure 9 shows percentages of users' sentiments of each car-related company. In all cases, half of the tweets about each car-related company do not contain user's sentiments about their commercial. As well as, tweets containing positive sentiments about car commercials are posted on Twitter more than tweets containing negative sentiments.

4.4.3 User location analysis. In the third phase, we introduce a user's location filtering rule to identify the state where a user lives from the user's profile where he can enter characters arbitrarily to hide their real information. Basically, the rule examine if the location field of a user contains full name and abbreviations of all states. For example, if the location field of a user contains "CA" or "California", the rule assumes that the user live in California. In addition, the rule excludes the user whose location field contains other regions that are similar to abbreviation of state. For example, if the location field of a user contains "CANADA" where "CA" is included, the rule assumes that the user does not live in California. A Rule for Identifying User Location shows an example of rule to identify user's location:

```
rule "findCA"
  when
    u:User (location matches ".*CA.*") or
      ...
```

Category	Positive	Negative	Neutral	Mixed	Total
Number of tweets categorized by rule-based algorithm	452	75	941	32	1,500
Number of tweets categorized by human	470	70	941	19	1,500
Number of tweets matched with results of rule	357	25	819	6	1,207/1,500
Number of tweets unmatched with results of rule	113	45	122	13	293/1,500
Accuracy (matched) (%)	76	36	87	32	80.47

Table IV. Accuracies of the rule-based sentiment categorization algorithm compared to categorization results of human analysis

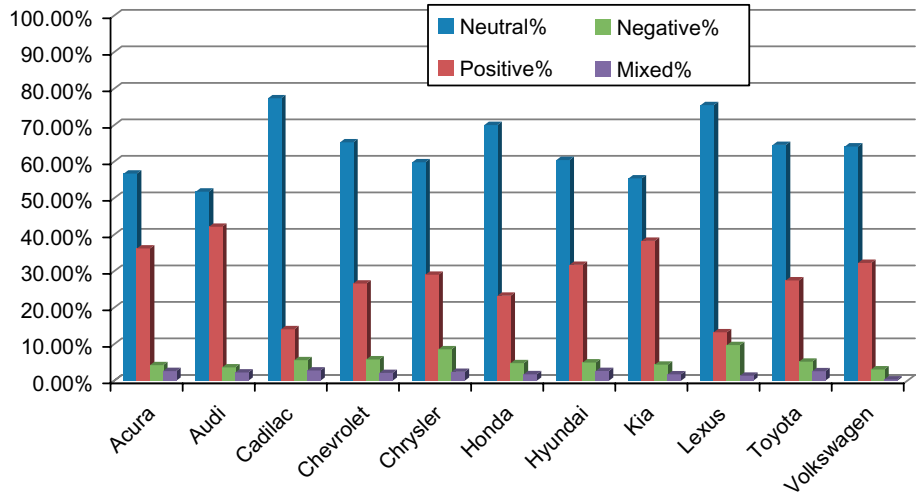

Figure 9. Percentages of twitter users' sentiments by car-related company

```
        u:User (location matches ".*[c,C]alifornia.*") and
        u:User (location not matches ".*[c,C]anada.*")
    then
        u.setAnalyzedLocation ("CA, U.S.A");
end
rule "findNY"
    when
        u:User (location matches ".*,NY.*") or
            ...
        u:User (location matches ".*, NY.*") or
        u:User (location matches ".*[n,N]ew York.*")
    then
            u.setAnalyzedLocation ("NY, U.S.A");
end
```

Table V shows a list of users and their location filtered by the user location filtering rule.

California		New York	
User id	Location	User id	Location
******034	Los Angeles, CA	******620	Rochester, NY
******770	Venice, CA	******026	New York City
******656	N.California	******659	New York
******269	California dmv girl	******228	New York
******510	California	******288	NEW YORK
******002	San Francisco, CA	******689	New York
******950	!CALIFORNIA!:D	******262	Buffalo Wing, New York
******578	SanBernardino.,California	******922	NEW YORK CITY
******285	Candy Mountain, California	******359	NewYork, NY
******561	Orange County, California	******250	New York
******364	San Diego, CA	******480	New York City
******967	Los Angeles, CA, USA	******457	Miami Beach, New York, LA
*****052	San Francisco, CA	******426	Clinton, NY
******911	Los Angeles, CA	******341	New York, NY
*****962	Dana Point, California!	******050	New York, NY
******442	San Diego, California	*****252	New York City!
******925	Clovis, CA	******831	Rochester, NY
****841	These days? Santa Barbara, CA	******703	New York
******536	Olivehurst, CA	******821	New York,New York
******078	Riverside, CA	******039	Canandaigua, NY
******438	Gnarstow, CA, USA	******119	Yonkers, NY
******617	San Pedro, California	******000	New York City
******638	Orange County, CA	******586	Central Square, NY
******230	Los Angeles, CA	******450	Shortsville, NY
******412	San Diego, CA	******288	New York City
******293	California	******278	New York
******566	Moreno Valley, CA	******570	New York
******031	Los Angeles, California	******524	Massapequa, NY
******834	San Diego, CA	******777	Coram, NY 11727
******390	San Pedro, CA	******118	New York, NY

Table V.
List of users and their location filtered by the user location filtering rule

5. Conclusions

The Twitter Data Collecting Tool allows researchers to gather users' information, follow relationships and tweets from Twitter. It is characterized by the following features. First, it is able to collect the data continuously and automatically. Second, it is able to start the collection process with multiple selected nodes. Third, it is able to handle a multitude of authorized developers' accounts. Fourth, it is able to save the collected data into a database. Fifth, it minimizes waste of hourly available methods calls. Finally, it is able to filter and analyze data using Drools that is embedded in the Twitter Data Collecting Tool.

By using the Twitter Data Collecting Tool, we gathered Twitter data about Super Bowl 2012 commercials, especially those related to cars. As a result, 11 rows of seed nodes, 1,134,798 rows of following relationships, 968,641 rows of follower IDs, 5,157,887 rows of tweets and 88,121 rows of user information have been gathered. After that, we filtered the tweets using Drools and retrieved the tweets about Super Bowl Commercials, 2012. Furthermore, we inferred the meaning of tweets using the rule engine and ranked the 11 car companies that advertised their commercials at Super Bowl 2012. In addition, data-mining techniques and rule-based data analysis are applied to the gathered data. As a result, new meaningful knowledge was discovered based on the results that are made by the Twitter Data Collecting Tool. With these results, we could prove that the Twitter Data Collecting Tool is able to gather a huge amount of data from Twitter and filter the data so it can be used in research areas. This paper will be valuable to those who may want to build their own Twitter dataset, apply customized filtering options to get rid of unnecessary, noisy data, and analyze social data to discover new knowledge.

Future work on the Twitter Data Collecting Tool can extend computing and storage capacity to gather more data from Twitter faster, because it consumes a lot of computing and storage resources. Using scalable clouding resources will be an easy way of extending these resources. Another improvement to the Twitter Data Collecting Tool is to adapt a built-in data-mining module. Applying data-mining techniques to the social network data has so much potential. For example, applying natural language processing techniques or text-mining to the Twitter data can be used to analyze or detect social opinions (Yassine and Hajj, 2010; Dziczkowski *et al.*, 2009). Therefore, a tool that supports collecting and mining the social data is needed for researchers to be able to use this untapped resource.

References

Al-Khalifa, H. (2012), "A first step to words understanding Saudi political activities on Twitter", *International Journal of Web Information Systems*, Vol. 8 No. 4, pp. 390-440.

Aramaki, E., Maskawa, S. and Morita, M. (2011), "Twitter catches the flu: detecting influenza epidemics using Twitter", *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing in Edinburgh, Scotland, UK*, Association for Computational Linguistics, Stroudsburg, PA, pp. 1568-1576.

Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnic, J., Kumar, S., Ravichandran, D. and Aly, M. (2008), "Video suggestion and discovery for Youtube: taking random walks through the view graph", *Proceeding of the 17th International Conference on World Wide Web, Beijing, China*, ACM, New York, NY, pp. 895-904.

Bošnjak, M., Oliveira, E., Martins, J., Mendes, E. and Sarmento, L. (2012), "TwitterEcho – a distributed focused crawler to support open research with Twitter data", *Proceedings of the 21st International Conference Companion on World Wide Web in Lyon, France*, ACM, New York, NY, pp. 1233-1240.

Byun, C., Park, K., Yun, J. and Kim, Y. (2011), "Design and implementation of the context-aware collaboration framework with the XCREAM (XLogic collaborative RFID/USN-enabled adaptive middleware)", paper presented at The Third International Conference on Smart IT Applications.

Choi, Y. and Cardie, C. (2009), "Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification", *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore*, Association for Computational Linguistics, Stroudsburg, PA, pp. 590-598.

Correa, D. and Sureka, A. (2011), "Mining tweets for tag recommendation on social media", *Proceedings of the 3rd International Workshop on Search and Mining User-Generated Contents in Glasgow, Scotland, UK*, ACM, New York, NY, pp. 69-76.

Cortizo, J.C., Carrero, F.M., Gomez, J.M., Monsalve, B. and Puertas, P. (2011), "Introduction to mining social media", *International Journal of Electronic Commerce*, Vol. 15 No. 3, pp. 5-8.

Dey, L. and Haque, S.M. (2008), "Opinion mining from noisy text data", *Proceedings of the Second Workshop on Analytics for Noisy Unstructured Text Data in Singapore*, ACM, New York, NY, pp. 80-90.

Dziczkowski, G., Bougueroua, L. and Wegrzyn-Wolska, K. (2009), "Social network – an tutonoumous system designed for radio recommendation", *International Conference on Computational Aspects of Social Networks in Fontainebleau, France, Cason*, pp. 57-64.

Friedman, E. (2013), "Jess, the rule engine for the JavaTM platform", available at: www.jessrules.com/jess/index.shtml (accessed 2 February 2013).

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I.H. (2009), "The WEKA data mining software: an update", *SIGKDD*, Vol. 11 No. 1, pp. 10-18.

Hu, M. and Liu, B. (2004), "Mining and summarizing customer reviews", *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining in Seattle, WA, USA*, ACM, New York, NY, pp. 168-177.

JBoss (2013), "Drools. Java rule engine", available at: www.jboss.org/drools/ (accessed 1 February 2013).

Kwak, H., Lee, C., Park, H. and Moon, S. (2010), "What is Twitter, a social network or a news media?", *Proceedings of the 19th International Conference on World Wide Web in Raleigh, NC, USA*, ACM, New York, NY, pp. 591-600.

Noordhuis, P., Heijkoop, M. and Lazovik, A. (2010), "Mining Twitter in the cloud", *Proceeding of IEEE 3rd International Conference on Cloud Computing in Miami, FL, USA*, IEEE Press, Washington, DC, pp. 107-114.

O'Connor, M., Balasubramanyan, B., Routledege, B.M. and Smith, N.M. (2010), "From tweets to polls: linking text sentiment to public opinion times series", *Proceedings of the International AAAI Conference on Weblogs and Social Media in Washington, DC, USA*.

Okazaki, T.M. and Matsuo, Y. (2010), "Earthquake shakes Twitter users: real-time event detection by social sensors", *Proc. of Conf. on World Wide Web* (*WWW*).

Sottara, D., Mello, P. and Proctor, M. (2010), "A configurable Rete-OO engine for reasoning with different types of imperfect information", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22 No. 11, pp. 1535-1548.

Sottara, D., Mello, P. and Proctor, M. (2011), "A configurable Rete-OO engine for reasoning with different types of imperfect information", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22 No. 11.

Super Bowl Commercials (2012), "10 best super bowl commercial", available at: www.superbowl-commercials.org/14261.html (accessed 10 February 2013).

Talukdar, P.P. and Crammer, C. (2009), "New regularized algorithms for transductive learning", *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, Bled, Slovenia*, Springer, Berlin, pp. 442-457.

Twitter (2013), "Twitter rate limiting in v.1.1", available at: https://dev.twitter.com/docs/rate-limiting/1.1 (accessed 25 February 2013).

Yassine, M. and Hajj, H. (2010), "A framework for emotion mining from text in online social networks", *IEEE 10th International Conference on Data Mining Workshops in Sydney, NSW, Austrailia*, IEEE Press, Washington, DC, pp. 1136-1142.

Zhu, M. and Ghahramani, Z. (2002), "Learning from labeled and unlabeled data with label propagation", Technical Report CMU-CALD-02-107, Carnegie Mellon University, Pittsburgh, PA.

Further reading

Bastian, M. (2009), "Gephi: an open source software for exploring and manipulating networks", paper presented at Third International AAAI Conference on Weblogs and Social Media in San Jose, California.

Domingos, P. and Richardson, M. (2001), "Mining the network value of customers", *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining in San Francisco, CA, USA*, ACM, New York, NY, pp. 57-66.

Forgy, C.L. (1982), "Rete: a fast algorithm for the many pattern/many object pattern match problem", *Department of Computer Science*, Carnegie-Mellon University, Pittsburgh, PA.

Fruchterman, T. and Reingold, E. (1991), "Graph drawing by force-directed placement", *Software-Practice and Experience*, Vol. 21 No. 11, pp. 1129-1164.

King, I., Li, J. and Chan, K.T. (2009), "A brief survey of computational approaches in social computing", *Proceedings of the 2009 International Joint Conference on Neural Networks in Piscataway, NJ, USA*, IEEE Press, Washington, DC, pp. 2699-2706.

Sakaki, T., Okazaki, M. and Matsuo, Y. (2010), "Earthquake shakes twitter users: real-time event detection by social sensors", *Proceedings of the 19th International Conference on World Wide Web in Raleigh, NC, USA*, ACM, New York, NY, pp. 851-860.

Corresponding author

Changhyun Byun can be contacted at: cbyun1@students.towson.edu

This article has been cited by:

1. Youngsub Han, Hyeoncheol Lee, Yanggon Kim. A real-time knowledge extracting system from social big data using distributed architecture 74-79. [Crossref]