

**MODELLING OF DEPENDENCIES & CASCADING EFFECTS
FOR EMERGENCY MANAGEMENT IN CRISIS SITUATIONS**

Project report submitted in partial fulfillment of the
requirement for the degree of Bachelor of Technology

**Computer Science and Engineering/Information
Technology**

By

Shivank Thakur (121207)

Under the Supervision of

Ms. Ruchi Verma

to



Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

TABLE OF CONTENTS

S. No.	Topic	Page No.
	CERTIFICATE	I
	ACKNOWLEDGEMENT	ii
	ABSTRACT	iii
	LIST OF FIGURES	iv
	CHAPTER 1	
	INTRODUCTION	1-15
1.1	INTRODUCTION	1
1.2	RESEARCH DOMAINS	2
1.3	SENTIMENT ANALYSIS	3
1.4	TAXONOMY OF SENTIMENT ANALYSIS	4
1.5	SUBJECTIVITY/OBJECTIVITY IDENTIFICATION	5
1.6	FEATURE/ASPECT-BASED SENTIMENT ANALYSIS	5
1.7	METHODS AND FEATURES	5
1.8	SENTIMENT ANALYSIS AND WEB 2.0	7
1.9	SENTIMENT VOCABULARIES AND ANNOTATED WORD LISTS	8
1.10	ONLINE SENTIMENT ANALYZERS	8
1.11	CREATING A SENTIMENT ANALYSIS MODEL	9
1.12	PROBLEM STATEMENT	12
1.13	OBJECTIVES	13
1.4	METHODOLOGY	14

CHAPTER 2	
LITERATURE SURVEY	16-37
2.1 REVIEW OF LITERATURE	16
2.2 SOCIAL AND OPINION MINING	21
2.3 FACEBOOK MINING	23
2.4 TWITTER API	27
2.5 GOOGLE MAPS API	28
2.6 FACEBOOK API	29
2.7 INSTAGRAM MINING	30
CHAPTER 3	
SYSTEM DEVELOPMENT	38-40
3.1 PROPOSED MODEL	38
3.2 HARDWARE REQUIREMENTS	39
3.3 SOFTWARE REQUIREMENTS	39
CHAPTER 4	
IMPLEMENTATION SETUP	41-49
4.1 FACEBOOK APPLICATION MODEL	41
4.2 FACEBOOK JAR FILES	42
4.3 INTEGRATION WITH TWITTER	46
CHAPTER 5	
CONCLUSION	50
REFERENCES / BIBLIOGRAPHY	51

CERTIFICATE

I hereby declare that the work presented in this report entitled “ **MODELLING OF DEPENDENCIES AND CASCADING EFFECTS FOR EMERGENCY MANAGEMENT IN CRISIS SITUATIONS**” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2015 to December 2015 under the supervision of **Ms. Ruchi Verma** (Assistant Professor, Computer Science & Engineering).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Shivank Thakur

121207

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Ms. Ruchi Verma

Assistant Professor

Computer Science & Engineering

Dated:

ACKNOWLEDGEMENT

This project work would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost I offer my sincerest gratitude to my supervisor Ms. Ruchi Verma who has supported me throughout my project work with his patience and knowledge whilst allowing me the room to work in my own way. I attribute the level of my Masters degree to her encouragement and effort and without him this thesis, too, would not have been completed or written. One simply could not wish for a better or friendlier supervisor.

The work was performed at Department of Computer Science and Engineering at the University, and I would like to thank all the people there for their hospitality and support.

Last, but not the least, my family and the one above all of us, the omnipresent God.

Date:

Shivank Thakur

ABSTRACT

This work is focused on the mining of real time data and messages using Facebook and Twitter APIs with Advance Java based implementation. The performance evaluation of Facebook and Twitter API is done for comparative analysis and conclusions. Opinion mining is a type of natural language processing for tracking the mood of the public about a particular product. Opinion mining, which is also called sentiment analysis, involves building a system to collect and categorize opinions about a product. Opinion mining, which is also called sentiment analysis, involves building a system to collect and categorize opinions about a product. Automated opinion mining often uses machine learning, a type of artificial intelligence (AI), to mine text for sentiment. Opinion mining can be useful in several ways. It can help marketers evaluate the success of an ad campaign or new product launch, determine which versions of a product or service are popular and identify which demographics like or dislike particular product features. For example, a review on a website might be broadly positive about a digital camera, but be specifically negative about how heavy it is. Being able to identify this kind of information in a systematic way gives the vendor a much clearer picture of public opinion than surveys or focus groups do, because the data is created by the customer. There are several challenges in opinion mining. In this work the opinion mining and comparative analysis is done using Facebook and Twitter APIs and performed the pragmatic review.

Keywords - Crisis Management using Social Media, Facebook Mining, Driving Innovation, Twitter API and Tweets Fetching, Java Based Twitter Integration

LIST OF FIGURES

Title	Page No.
Figure 1.1 – Methodology and Flow of Work	14
Figure 2.1 – Graph API for Social Mining	26
Figure 2.2 – Graph API Explorer	27
Figure 2.3 – Instagram Mining	31
Figure 2.3 – Clients Management in Instagram Mining	32
Figure 2.4 – Client Registration in Instagram Mining	33
Figure 2.5 – Managing Clients and Keys in Instagram	34
Figure 2.1 – Machine Learning and Data Modeling Process	38
Figure 2.2 – Sentiment Analysis	39
Figure 4.1 – Comparative Analysis between FACEBOOK and TWITTER Analytics	49

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Social media mining is the process of representing, analyzing, and extracting actionable patterns from social media data. Social media mining introduces basic concepts and principal algorithms suitable for investigating massive social media data; it discusses theories and methodologies from different disciplines such as computer science, data mining, machine learning, social network analysis, network science, sociology, ethnography, statistics, optimization, and mathematics. It encompasses the tools to formally represent, measure, model, and mine meaningful patterns from large-scale social media data.

As defined by Kaplan and Haenlein, social media is the "group of internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of user-generated content." There are many categories of social media including, but not limited to, social networking (Facebook or LinkedIn), microblogging (Twitter), photo sharing (Flickr, Photobucket, or Picasa), news aggregation (Google reader, StumbleUpon, or Feedburner), video sharing (YouTube, MetaCafe), livecasting (Ustream or Twitch.tv), virtual worlds (Kaneva), social gaming (World of Warcraft), social search (Google, Bing, or Ask.com), and instant messaging (Google Talk, Skype, or Yahoo! messenger).

The first social media site was introduced by GeoCities in 1994, which allowed users to create their own homepages. The first social networking site, SixDegree.com, was introduced in 1997. Since then, many other social media sites have been introduced, each providing service to millions of people. These individuals form a virtual world in which individuals (social atoms), entities (content, sites, etc.) and interactions (between individuals, between entities, between individuals and entities) coexist. Social norms and human behavior govern this virtual world. By understanding these social norms and models of human behavior and combining them with the observations and measurements of this virtual world, one can systematically analyze and mine social media. Social media mining is the process of representing, analyzing, and extracting

meaningful patterns from data in social media, resulting from social interactions. It is an interdisciplinary field encompassing techniques from computer science, data mining, machine learning, social network analysis, network science, sociology, ethnography, statistics, optimization, and mathematics. Social media mining faces grand challenges such as the big data paradox, obtaining sufficient samples, the noise removal fallacy, and evaluation dilemma.

Social media mining represents the virtual world of social media in a computable way, measures it, and designs models that can help us understand its interactions. In addition, social media mining provides necessary tools to mine this world for interesting patterns, analyze information diffusion, study influence and homophily, provide effective recommendations, and analyze novel social behavior in social media.

1.2 RESEARCH DOMAINS

- Community structure (Community Detection/Evolution/Evaluation) – Identifying communities on social networks, how they evolve, and evaluating identified communities, often without ground truth.
- Network measures – Measuring centrality, transitivity, reciprocity, balance, status, and similarity in social media.
- Network models – Simulate networks with specific characteristics. Examples include random graphs (E-R models), Preferential attachment models, and small-world models.
- Information cascade – Analyzing how information propagates in social media sites. Examples include herd behavior, information cascades, diffusion of innovations, and epidemic models.
- Influence and homophily – Measuring network assortativity and measuring and modeling influence and homophily.
- Recommendation in social media – recommending friends or items on social media sites.
- Social search – Searching for information on the social web.
- Sentiment analysis in social media – Identifying collectively subjective information, e.g. positive and negative, from social media data.

- Social spammer detection – Detecting social spammers who send out unwanted spam content appearing on social networks and any website with user-generated content to targeted users, often corroborating to boost their social influence, legitimacy, credibility.
- Feature selection with social media data - Transforming feature selection to harness the power of social media.
- Trust in social media - Studying and understanding of trust in social media.
- Distrust and negative links - Exploring negative links in social media.
- Role of social media in crises - Social media is continuing to play an important role during crises, particularly Twitter. Studies show that it is possible to detect earthquakes and rumors using tweets published during crisis. Developing tools to help first responders to analyze tweets towards better crisis response and developing techniques to provide them faster access to relevant tweets is an active area of research.
- Location-based social network mining - Mining Human Mobility for Personalized POI Recommendation on Location-based Social Networks.
- Provenance of information in social media - Provenance informs a user about the sources of a given piece of information. Social media can help in identifying the provenance of information due its unique features: user-generated content, user profiles, user interactions, and spatial or temporal information.
- Vulnerability management - A user's vulnerability on a social networking sites can be managed in three sequential steps: (1) identifying new ways in which a user can be vulnerable, (2) quantifying or measuring a user's vulnerability, and (3) reducing or mitigating them.

1.3 SENTIMENT ANALYSIS

Sentiment analysis (also known as opinion mining) refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. Sentiment analysis is widely applied to reviews and social media for a variety of applications, ranging from marketing to customer service.

Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document. The attitude may be

his or her judgment or evaluation (see appraisal theory), affective state (that is to say, the emotional state of the author when writing), or the intended emotional communication (that is to say, the emotional effect the author wishes to have on the reader).

1.4 TAXONOMY OF SENTIMENT ANALYSIS

A basic task in sentiment analysis is classifying the *polarity* of a given text at the document, sentence, or feature/aspect level — whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, "beyond polarity" sentiment classification looks, for instance, at emotional states such as "angry," "sad," and "happy."

Early work in that area includes Turney and Pang who applied different methods for detecting the polarity of product reviews and movie reviews respectively. This work is at the document level. One can also classify a document's polarity on a multi-way scale, which was attempted by Pang and Snyder among others: Pang and Lee expanded the basic task of classifying a movie review as either positive or negative to predicting star ratings on either a 3 or a 4 star scale, while Snyder performed an in-depth analysis of restaurant reviews, predicting ratings for various aspects of the given restaurant, such as the food and atmosphere (on a five-star scale). Even though in most statistical classification methods, the neutral class is ignored under the assumption that neutral texts lie near the boundary of the binary classifier, several researchers suggest that, as in every polarity problem, three categories must be identified. Moreover it can be proven that specific classifiers such as the Max Entropy and the SVMs can benefit from the introduction of neutral class and improve the overall accuracy of the classification.

A different method for determining sentiment is the use of a scaling system whereby words commonly associated with having a negative, neutral or positive sentiment with them are given an associated number on a -10 to +10 scale (most negative up to most positive). This makes it possible to adjust the sentiment of a given term relative to its environment (usually on the level of the sentence). When a piece of unstructured text is analyzed using natural language processing, each concept in the specified environment is given a score based on the way sentiment words relate to the concept and its associated score. This allows movement to a more sophisticated understanding of sentiment, because it is now possible to adjust the sentiment value

of a concept relative to modifications that may surround it. Words, for example, that intensify, relax or negate the sentiment expressed by the concept can affect its score. Alternatively, texts can be given a positive and negative sentiment strength score if the goal is to determine the sentiment in a text rather than the overall polarity and strength of the text.

1.5 SUBJECTIVITY/OBJECTIVITY IDENTIFICATION

This task is commonly defined as classifying a given text (usually a sentence) into one of two classes: objective or subjective. This problem can sometimes be more difficult than polarity classification. The subjectivity of words and phrases may depend on their context and an objective document may contain subjective sentences (e.g., a news article quoting people's opinions). Moreover, as mentioned by Su, results are largely dependent on the definition of subjectivity used when annotating texts. However, Pangshowed that removing objective sentences from a document before classifying its polarity helped improve performance.

1.6 FEATURE/ASPECT-BASED SENTIMENT ANALYSIS

It refers to determining the opinions or sentiments expressed on different features or aspects of entities, e.g., of a cell phone, a digital camera, or a bank. A feature or aspect is an attribute or component of an entity, e.g., the screen of a cell phone, the service for a restaurant, or the picture quality of a camera. The advantage of feature-based sentiment analysis is the possibility to capture nuances about objects of interest. Different features can generate different sentiment responses, for example a hotel can have a convenient location, but mediocre food. This problem involves several sub-problems, e.g., identifying relevant entities, extracting their features/aspects, and determining whether an opinion expressed on each feature/aspect is positive, negative or neutral. The automatic identification of features can be performed with syntactic methods or with topic modeling. More detailed discussions about this level of sentiment analysis can be found in Liu's work.

1.7 METHODS AND FEATURES

Existing approaches to sentiment analysis can be grouped into three main categories: knowledge-based techniques, statistical methods, and hybrid approaches. Knowledge-based techniques classify text by affect categories based on the presence of unambiguous affect words such as

happy, sad, afraid, and bored. Some knowledge bases not only list obvious affect words, but also assign arbitrary words a probable “affinity” to particular emotions. Statistical methods leverage on elements from machine learning such as latent semantic analysis, support vector machines, “bag of words” and *Semantic Orientation — Pointwise Mutual Information* (See Peter Turney's work in this area). More sophisticated methods try to detect the holder of a sentiment (i.e., the person who maintains that affective state) and the target (i.e., the entity about which the affect is felt). To mine the opinion in context and get the feature which has been opinionated, the grammatical relationships of words are used. Grammatical dependency relations are obtained by deep parsing of the text. Hybrid approaches leverage on both machine learning and elements from knowledge representation such as ontologies and semantic networks in order to detect semantics that are expressed in a subtle manner, e.g., through the analysis of concepts that do not explicitly convey relevant information, but which are implicitly linked to other concepts that do so.

Open source software tools deploy machine learning, statistics, and natural language processing techniques to automate sentiment analysis on large collections of texts, including web pages, online news, internet discussion groups, online reviews, web blogs, and social media. Knowledge-based systems, instead, make use of publicly available resources, e.g., WordNet-Affect, SentiWordNet, SenticNet, and Emoji Sentiment Ranking to extract the semantic and affective information associated with natural language concepts. Sentiment Analysis can also be performed on visual content, i.e., images and videos. One of the first approach in this direction is SentiBank utilizing an adjective noun pair representation of visual content.

A human analysis component is required in sentiment analysis, as automated systems are not able to analyze historical tendencies of the individual commenter, or the platform and are often classified incorrectly in their expressed sentiment. Automation impacts approximately 23% of comments that are correctly classified by humans.

Sometimes, the structure of sentiments and topics is fairly complex. Also, the problem of sentiment analysis is non-monotonic in respect to sentence extension and stop-word substitution

(compare *THEY would not let my dog stay in this hotel* vs *I would not let my dog stay in this hotel*). To address this issue a number of rule-based and reasoning-based approaches have been applied to sentiment analysis, including Defeasible Logic Programming. Also, there is a number of tree traversal rules applied to syntactic parse tree to extract the topicality of sentiment in open domain setting.

The accuracy of a sentiment analysis system is, in principle, how well it agrees with human judgments. This is usually measured by precision and recall. However, according to research human raters typically agree 79% of the time (see Inter-rater reliability).

Thus, a 70% accurate program is doing nearly as well as humans, even though such accuracy may not sound impressive. If a program were "right" 100% of the time, humans would still disagree with it about 20% of the time, since they disagree that much about *any* answer . More sophisticated measures can be applied, but evaluation of sentiment analysis systems remains a complex matter. For sentiment analysis tasks returning a scale rather than a binary judgement, correlation is a better measure than precision because it takes into account how close the predicted value is to the target value.

1.8 SENTIMENT ANALYSIS AND WEB 2.0

The rise of social media such as blogs and social networks has fueled interest in sentiment analysis. With the proliferation of reviews, ratings, recommendations and other forms of online expression, online opinion has turned into a kind of virtual currency for businesses looking to market their products, identify new opportunities and manage their reputations. As businesses look to automate the process of filtering out the noise, understanding the conversations, identifying the relevant content and actioning it appropriately, many are now looking to the field of sentiment analysis. Further complicating the matter, is the rise of anonymous social media platforms such as 4chan and Reddit. If web 2.0 was all about democratizing publishing, then the next stage of the web may well be based on democratizing data mining of all the content that is getting published.

One step towards this aim is accomplished in research. Several research teams in universities around the world currently focus on understanding the dynamics of sentiment in e-communities through sentiment analysis. The CyberEmotions project, for instance, recently identified the role of negative emotions in driving social networks discussions.

The problem is that most sentiment analysis algorithms use simple terms to express sentiment about a product or service. However, cultural factors, linguistic nuances and differing contexts make it extremely difficult to turn a string of written text into a simple pro or con sentiment. The fact that humans often disagree on the sentiment of text illustrates how big a task it is for computers to get this right. The shorter the string of text, the harder it becomes.

Even though short text strings might be a problem, sentiment analysis within microblogging has shown that Twitter can be seen as a valid online indicator of political sentiment. Tweets' political sentiment demonstrates close correspondence to parties' and politicians' political positions, indicating that the content of Twitter messages plausibly reflects the offline political landscape.

Resources for sentiment analysis

1.9 SENTIMENT VOCABULARIES AND ANNOTATED WORD LISTS

- Affective Norms for English Words (ANEW)
- AFINN
- SenticNet
- SentiWordNet
- WordNet-Affect
- Emoji Sentiment Ranking

1.10 ONLINE SENTIMENT ANALYZERS

- 30dB (free)
- AlchemyAPI (commercial)
- BitextAPI (commercial)
- HPE Haven OnDemand (commercial, with freemium)
- Semantria (commercial)

- Sentiment140 (commercial, for Twitter)
- Stanford NLP (academic)
- Sentic API (commercial)
- Twinword (commercial, free / unlimited)
- Werfamous (free)
- WordStat (commercial)
- Buzzlogix (free and commercial versions)
- Twitter dataset in 4 languages (12,500 tweets)

1.11 CREATING A SENTIMENT ANALYSIS MODEL

A sentiment analysis model is used to analyze a text string and classify it with one of the labels that you provide; for example, you could analyze a tweet to determine whether it is positive or negative, or analyze an email to determine whether it is happy, frustrated, or sad. You should read the Hello Prediction page before reading this introduction.

Step 1: Collect Data

You must train your sentiment model against examples of the type of data that you are going to see when you use your model. For example, if you are trying to determine the sentiment of tweets, you will need to obtain examples of tweet entries. For example:

- "Feeling kind of low...."
- "OMG! Just had a fabulous day!"
- "Eating eggplant. Why bother?"

You can either provide your own data or buy it. For a good model, you will need at least several hundred examples, if not thousands. All of the samples should be in the same language (though it doesn't matter what language, as long as its consistent and space-delimited).

Step 2: Label Your Data

Once you have collected your training samples, you will need to pre-classify each sample with a label. A label is a string that you think best describes that example, for example: "happy", "sad", "on the fence". So, to assign labels to the previous examples:

- "sad", "Feeling kind of low...."
 - "excited", "OMG! Just had a fabulous day!"
 - "bored", "Eating eggplant. Why bother?"
- You can have up to 1,000 labels for a model, but you should only use as many labels as are useful to you, and you must have at least a few dozen examples assigned each type of label that you assign.
 - Labels are just strings, so they can have spaces. However, you should put double quotes around any labels that have spaces, and you should escape any nested quotation marks using a \ mark. Example: "that\'s fine"
 - Labels are case-sensitive. So "Happy" and "happy" will be seen as two separate labels by the training system. Best practice is to use lowercase for all labels, to avoid mix-ups.
 - Each line can only have one label assigned, but you can apply multiple labels to one example by repeating an example and applying different labels to each one. For example:
 - "excited", "OMG! Just had a fabulous day!"
 - "annoying", "OMG! Just had a fabulous day!"

If you send a tweet to this model, you might get a classification something like this:
 "excited":0.6, "annoying":0.2.

Step 3: Prepare Your Data

The Google Prediction API takes training data formatted as a comma-separated values (CSV) file with one row per example. The format of this file is basically this:

```
label1, feature1, feature2, feature3,....
label2, feature1, feature2, feature3....
...
```

In the previous example, each example had a single feature: a text string that is a tweet. So the file would look something like this:

```
"sad", "Feeling kind of low...."
"excited", "OMG! Just had a fabulous day!"
"bored", "Eating eggplant. Why bother?"
```

However, if you have more data that you think would help Google Prediction find some underlying patterns, it would be useful to include that information as well. For example, if you think that message length is meaningful (longer messages indicate happier tweets) or time of day (daytime tweets are happier than nighttime tweets), you could create additional features for that data. The following example shows the label, tweet text, message word count, and numeric version of the time of day for each tweet:

```
"sad", "Feeling kind of low....", 4, 18.30  
"excited", "OMG! Just had a fabulous day!", 6, 9.10  
"bored", "Eating eggplant. Why bother?", 4, 12.00
```

Google Prediction API doesn't take datetime values, so you must find an equivalent; here, times are specified as numbers. Also note that there should be a good correlation between all features and the label that you assign. See [Designing a Good Model](#) for more tips on creating good training data.

There are offline tools you can use (such as Weka and R) that can help build sample models and identify what features impact your model the most. There are unfortunately no guarantees though when building a model for your problem, the best way to achieve optimal results is to build a few models with different feature sets and use the one that works best. As a general rule though including all the relevant data possible will produce the best results as the algorithms are good at ignoring features that don't influence outcomes.

Step 4: Upload Data to Google Cloud Storage

Once you have your data available in CSV format it is time to upload it to Google Cloud Storage. There are many ways to do this.

- With a web interface in the Google Cloud Platform Console
- With a command line tool: GSUtil
- By using the Google Cloud Storage API

Step 5: Train a Model with the Google Prediction API

Train your model using either the Google API Explorer or a client library. The method for training is `prediction.trainedmodels.insert()`, passing in the path to your training data in Google Cloud Storage. Note: do not prefix the data file path with “gs://”. That syntax is exclusively for the GSUtil utility.

Step 6: Make Predictions with the Google Prediction API in your application.

Now that you have successfully built a model, it is time to actually make predictions! The output from a prediction call for a classification problem like sentiment analysis will include several important fields:

1. `outputLabel` - The label that the API determined was most likely to apply to this sample
2. `outputMulti` - A detailed breakdown per label of how likely that label is to apply to this sample (All scores in `outputMulti` will sum to 1)

Step 7: Update Your Model with New Data

You can continue to improve your model by adding additional examples. There are two ways to add additional examples to your model:

- Add the new data to the original data file and retrain the model with another `insert()` call.
- Update your existing model using the `update` method; this adds new examples on the fly to an existing model. The downside of this option is that some classifiers cannot be updated, so it is possible that an updated classifier will have worse accuracy than the classifier trained on all the data in batch.

1.12 PROBLEM STATEMENT

- To fetch the live real time data from social media for detailed and deep investigation with sentiment analysis
- To use the advance Java Wrappers for social media mining
- To analyze the results of social media and web scraping to evaluate the overall score
- Evaluation of the results from Facebook and Twitter

1.13 OBJECTIVES

1. To search the keywords / strings on social media
 - Facebook
 - Twitter
2. To perform a comparative analysis of various APIs used by assorted social media applications
3. To compare the performance between Facebook and Twitter APIs
4. The tweets will be searched based on any string with a database connectivity using JDBC to MySQL Database Engine.
5. Creation of Java Based GUI for fetching the real time datasets in JSON and CSV formats

METHODOLOGY

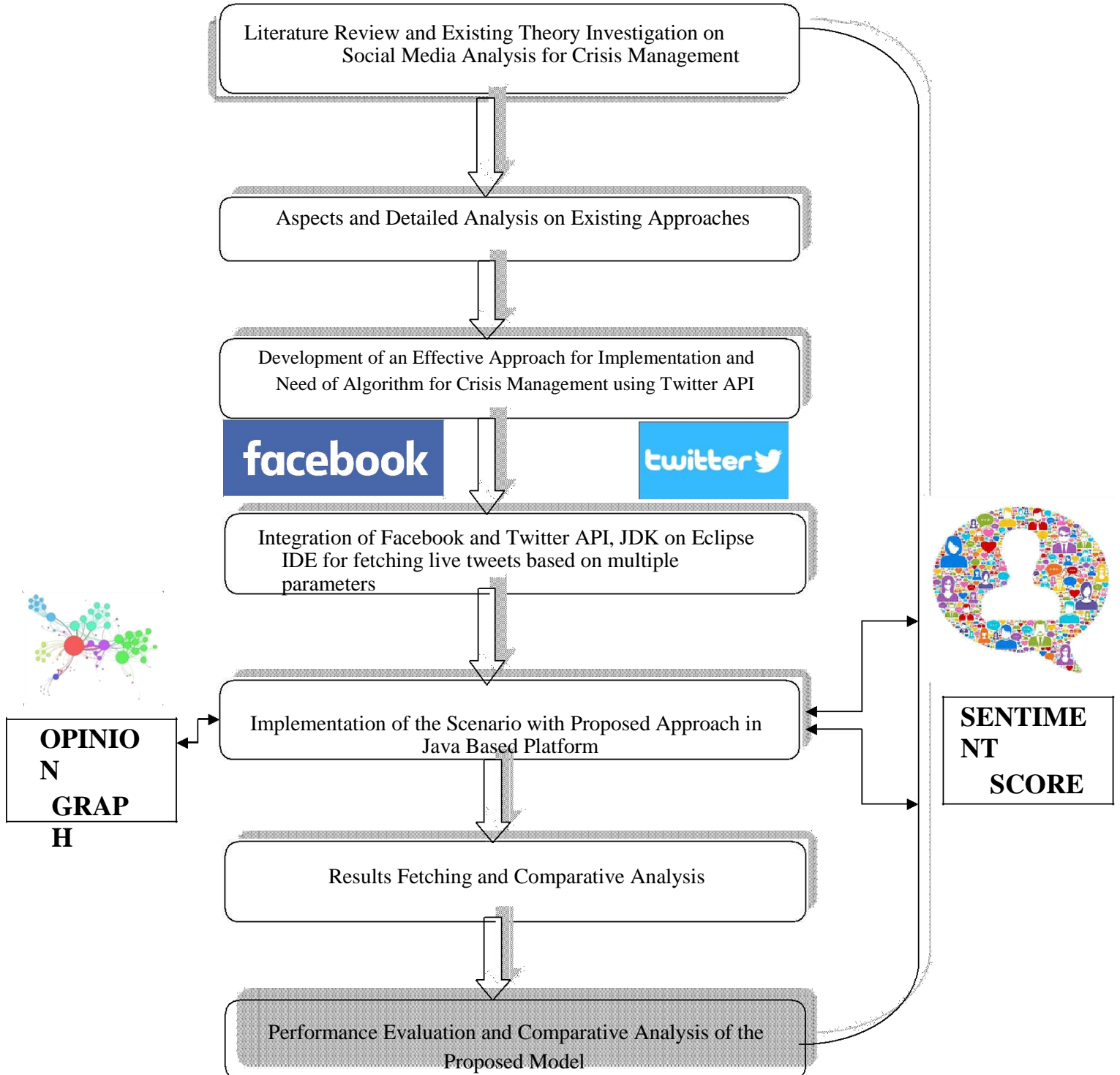


Figure 1.1 – Methodology and Flow of Work

ORGANIZATION

Chapter 1 highlights and underlines the assorted aspects of social media mining. In this chapter, the introduction to various approaches and technologies for sentiment score is covered. The key focus on Social Media Mining and Prediction is done so that the proposed work can be defended.

Chapter 2 is having the detailed literature review from the research paper, books, journals and conferences. In this chapter, the extracts from assorted research papers on Social Media Mining and Prediction are taken and depicted.

Chapter 3 covers the system development which is the key aspect of this work. In this chapter, the proposed model, algorithm and related parameters are emphasized.

The simulation of implementation results with the relative performance analysis is shown in **Chapter 4**. In this chapter, the simulation results and screenshots are revealed to depict and defend the work

Chapter 5 ends with the detailed conclusion and scope of the future work which guides the upcoming students and research scholars to enhance the current work with higher efficiency and effectiveness.

CHAPTER 2

LITERATURE SURVEY

For completion, justification and solving the problem definition, a number of research papers, magazines, journals and online links are investigated in details. A number of research scholars and scientists has written a number of research papers and found excellent results. This section underlines all those research papers and their extracts

2.1 REVIEW OF LITERATURE

Kumar et al. implements three phases of Web usage mining namely preprocessing, pattern discovery, and pattern analysis. Apriori algorithm is used to generate an association rule that associates the usage pattern of the clients for a particular website. The output of the system was in terms of memory usage and speed of producing association rules. A clustering algorithm to find out data clusters for both numerical and nominal data is proposed by Sharma et al. by calculating the average and log values of data set. This algorithm improves the techniques of Web Usage Mining by first discover the log files of individual users at one place.

Martinez-Romo et al. have analyzed different information retrieval methods for both, the selection of terms used to construct the queries submitted to the search engine, and the ranking of the candidate pages that it provides, in order to help the user to find the best replacement for a broken link. To test the sources, they have also defined an evaluation methodology which does not require the user judgments, what increases the objectivity of the results.

A new reactive session reconstruction method is given by Dohare et al. . This algorithm is better than previously developed both time and navigation oriented heuristics as it does not allow page sequences with any unrelated consecutive requests to be in the same session. They have also implemented agent simulator for generating real user sessions. Das et al. analyzed the web server user access logs of Firat University to help system administrator and Web designer to improve their system by determining occurred system errors, corrupted and broken links by using web using mining.

Fayyad et al. have focused on web log file format, its type and location. Log files usually contain noisy and ambiguous data. Preprocessing involves removal of unnecessary data from log file. Data preprocessing is an important step to filter and organize appropriate information before using to web mining algorithm. They have also proposed two algorithms for field extraction and data cleaning. Preprocessing web log file is used in data mining techniques, also used in intrusion detection system as input to detect intrusion. Apriori - the first scalable algorithm designed for association-rule mining algorithm. Apriori is an improvement over the AIS and SETM algorithms stated, Das et al. . The algorithm is based on the large itemset property which states: Any subset of a large itemset is large and if an itemset is not large and then none of its supersets are large.

The Apriori algorithm searches for large itemsets during its initial database pass and uses its result as the seed for discovering other large datasets during subsequent passes Brachman et al. specifies that the rules having a support level above the minimum are called large or frequent itemsets and those below are called small itemsets. Agrawal et al. derived Association Rules from data called the “market-basket problem”. Given a set of items and a large collection of transactions which are sets (baskets) of items from a database, the task is to find relationships between the containments of various items within those baskets. The task in Association Rule mining involves finding all rules that satisfy user defined constraints on minimum support and confidence with respect to a given dataset. Most commonly used Association Rule discovery algorithm that utilizes the frequent itemset strategy is exemplified by the Apriori algorithm.

Data mining (DM) a step from Knowledge Discovery in Database (KDD) process, Andrassyova et al. defines it as a “nontrivial process of identifying valid, novel, potentially useful and ultimately understandable pattern in data”. The term pattern here refers some abstract representation of a subset data of the data, that is, an expression in some language describing a data subset or a data subset or a model applicable to that subset.

Kleinberg categorized web mining into three areas of interest based on which part of the Web to mine; Web Content mining, Web Structure mining, and Web Usage Mining. In Web mining, data collected at the server-side, client-side, proxy servers or a consolidated Web/business

database. Hedberg [11] provided data sources that can be used to construct several data abstractions, namely users, page-views, click-streams and server sessions.

Tang et al. have used re-ranking method and generalized Association Rules to extract access patterns of the Web sites pattern usage.

Bollen (2010) - Behavioral economics tells us that emotions can profoundly affect individual behavior and decision-making. Does this also apply to societies at large, i.e. can societies experience mood states that affect their collective decision making? By extension is the public mood correlated or even predictive of economic indicators? Here we investigate whether measurements of collective mood states derived from large-scale Twitter feeds are correlated to the value of the Dow Jones Industrial Average (DJIA) over time. This work analyze the text content of daily Twitter feeds by two mood tracking tools, namely OpinionFinder that measures positive vs. negative mood and Google-Profile of Mood States (GPOMS) that measures mood in terms of 6 dimensions (Calm, Alert, Sure, Vital, Kind, and Happy). This paper cross-validate the resulting mood time series by comparing their ability to detect the public's response to the presidential election and Thanksgiving day in 2008. A Granger causality analysis and a Self-Organizing Fuzzy Neural Network are then used to investigate the hypothesis that public mood states, as measured by the OpinionFinder and GPOMS mood time series, are predictive of changes in DJIA closing values. The results in this paper indicate that the accuracy of DJIA predictions can be significantly improved by the inclusion of specific public mood dimensions but not others. The authors find an accuracy of 87.6% in predicting the daily up and down changes in the closing values of the DJIA and a reduction of the Mean Average Percentage Error by more than 6%.

Bollen (2009) - Microblogging is a form of online communication by which users broadcast brief text updates, also known as tweets, to the public or a selected circle of contacts. A variegated mosaic of microblogging uses has emerged since the launch of Twitter in 2006: daily chatter, conversation, information sharing, and news commentary, among others. Regardless of their content and intended use, tweets often convey pertinent information about their author's mood status. As such, tweets can be regarded as temporally-authentic microscopic instantiations of public mood state. In this article, we perform a sentiment analysis of all public tweets

broadcasted by Twitter users between August 1 and December 20, 2008. For every day in the timeline, we extract six dimensions of mood (tension, depression, anger, vigor, fatigue, confusion) using an extended version of the POMS, a well-established psychometric instrument. The authors in this paper compare the results to the values recorded by stock market and crude oil price indices and major events in media and popular culture, such as the U.S. Presidential Election of November 4, 2008 and Thanksgiving Day. This work finds that events in the social, political, cultural and economic sphere do have a significant, immediate on the various dimensions of public mood. The authors speculate that large scale analyses of mood can provide a solid platform to model collective emotive trends in terms of their predictive value with regards to existing social as well as economic indicators.

Asur (2010)– Sentiment Analysis is important part for social networking and content sharing. And yet, the content that is generated from these websites remains largely untapped. In this paper, the authors demonstrate how social media content can be used to predict real-world outcomes. In particular, this work uses the chatter from Twitter.com to forecast box-office revenues for movies. This paper shows that a simple model built from the rate at which tweets are created about particular topics can outperform market-based predictors. This work further demonstrates how sentiments extracted from Twitter can be further utilized to improve the forecasting power of social media.

Tan (2011)– The authors show that information about social relationships can be used to improve user-level sentiment analysis. The main motivation behind the approach is that users that are somehow “connected” may be more likely to hold similar opinions; therefore, relationship information can complement what we can extract about a user’s viewpoints from their utterances. Employing Twitter as a source for our experimental data, and working within a semi-supervised framework, we propose models that are induced either from the Twitter follower/followee network or from the network in Twitter formed by users referring to each other using “@” mentions. The proposed transductive learning results reveal that incorporating social-network information can indeed lead to statistically significant sentiment classification improvements over the performance of an approach based on Support Vector Machines having access only to textual features.

Saif (2012) - Sentiment analysis over Twitter offer organisations a fast and effective way to monitor the publics' feelings towards their brand, business, directors, etc. A wide range of features and methods for training sentiment classifiers for Twitter datasets have been researched in recent years with varying results. In this paper, we introduce a novel approach of adding semantics as additional features into the training set for sentiment analysis. For each extracted entity (e.g. iPhone) from tweets, we add its semantic concept (e.g. "Apple product") as an additional feature, and measure the correlation of the representative concept with negative/positive sentiment. The authors apply this approach to predict sentiment for three different Twitter datasets. The results show an average increase of F harmonic accuracy score for identifying both negative and positive sentiment of around 6.5% and 4.8% over the baselines of unigrams and part-of-speech features respectively. We also compare against an approach based on sentiment-bearing topic analysis, and find that semantic features produce better Recall and F score when classifying negative sentiment, and better Precision with lower Recall and F score in positive sentiment classification.

Davidov (2010) - Automated identification of diverse sentiment types can be beneficial for many NLP systems such as review summarization and public media analysis. In some of these systems there is an option of assigning a sentiment value to a single sentence or a very short text. In this paper the authors proposes a supervised sentiment classification framework which is based on data from Twitter, a popular microblogging service. By utilizing 50 Twitter tags and 15 smileys as sentiment labels, this framework avoids the need for labor intensive manual annotation, allowing identification and classification of diverse sentiment types of short texts. The authors evaluate the contribution of different feature types for sentiment classification and show that the proposed framework successfully identifies sentiment types of untagged sentences. The quality of the sentiment identification was also confirmed by human judges. This paper also explores dependencies and overlap between different sentiment types represented by smileys and Twitter hashtags.

Saif (2012) - Twitter has brought much attention recently as a hot research topic in the domain of sentiment analysis. Training sentiment classifiers from tweets data often faces the data sparsity

problem partly due to the large variety of short and irregular forms introduced to tweets because of the 140-character limit. In this work the authors proposes using two different sets of features to alleviate the data sparseness problem. One is the semantic feature set where this work extracts semantically hidden concepts from tweets and then incorporate them into classifier training through interpolation. Another is the sentiment-topic feature set where we extract latent topics and the associated topic sentiment from tweets, then augment the original feature space with these sentiment-topics. Experimental results on the Stanford Twitter Sentiment Dataset show that both feature sets outperform the baseline model using unigrams only. Moreover, using semantic features rivals the previously reported best result. Using sentiment topic features achieves 86.3% sentiment classification accuracy, which outperforms existing approaches.

Bifet (2009) - Micro-blogs are a challenging new source of information for data mining techniques. Twitter is a micro-blogging service built to discover what is happening at any moment in time, anywhere in the world. Twitter messages are short, and generated constantly, and well suited for knowledge discovery using data stream mining. The authors briefly discuss the challenges that Twitter data streams pose, focusing on classification problems, and then consider these streams for opinion mining and sentiment analysis. To deal with streaming unbalanced classes, this work propose a sliding window Kappa statistic for evaluation in time-changing data streams. Using this statistic this work performs a study on Twitter data using learning algorithms for data streams.

2.2 SOCIAL AND OPINION MINING

Social networking, blogging and online forums have turned the web into a vast repository of comments on many topics, generating a potential source of information for social science research (Thelwall, Wouters, & Fry, 2008). The availability of large scale electronic social data from the web and elsewhere is already transforming social research (Savage & Burrows, 2007). The social web is also being commercially exploited for goals such as automatically extracting customer opinions about products or brands. An application could build a large database of web sources (Bansal & Koudas, 2007; Gruhl, Chavet, Gibson, Meyer, & Pattanayak, 2004), use information retrieval techniques to identify potentially relevant texts, then extract information about target products or brands, such as which aspects are disliked (Gamon, Aue, Corston-

Oliver, & Ringger, 2005; Jansen, Zhang, Sobel, & Chowdury, 2009). From a social sciences perspective, similar methods could potentially give insights into public opinion about a wide range of topics and are unobtrusive, avoiding human subjects research issues (Bassett & O'Riordan, 2002; Enyon, Schroeder, & Fry, 2009; Hookway, 2008; White, 2002).

The sheer size of the social web has also made possible a new type of informal literature-based discovery (for literature based discovery, see: Bruza & Weeber, 2008; Swanson, Smalheiser, & Bookstein, 2001): the ability to *automatically* detect events of interest, perhaps within pre-defined broad topics, by scanning large quantities of web data. For instance, one project used time series analyses of (mainly) blogs to identify emerging public fears about science (Thelwall & Prabowo, 2007) and businesses can use similar techniques to quickly discover customer concerns. Emerging important events are typically signalled by sharp increases in the frequency of relevant terms. These bursts of interest are important to study because of their role in detecting new events as well as for the importance of the events discovered. One key unknown is the role of sentiment in the emergence of important events because of the increasing recognition of the importance of emotion in awareness, recall and judgement of information (Fox, 2008, p. 242-244, 165-167, 183; Kinsinger & Schacter, 2008) as well as motivation associated with information behaviour (Case, 2002, p. 71-72; Nahl, 2006, 2007a).

The research field of sentiment analysis, also known as opinion mining, has developed many algorithms to identify whether an online text is subjective or objective, and whether any opinion expressed is positive or negative (Pang & Lee, 2008). Such methods have been applied on a large scale to study sentiment-related issues. One widely-publicised study focused on the average level of sentiment expressed in blogs (as well as lyrics and US presidential speeches) in order to identify overall trends in levels of happiness as well as age and geographic differences in the expression of happiness (Dodds & Danforth, 2010). A similar approach used Facebook status updates to judge changes in mood over the year and to assess "the overall emotional health of the nation" (Kramer, 2010) and another project assessed six dimensions of emotion in Twitter, showing that these typically reflect significant offline events (Bollen, Pepe, & Mao, 2009). Nevertheless, despite some research into the role of sentiment in online communication, there are no investigations into the role that sentiment plays in important online events. To partially fill

this gap, this study assesses whether Twitter-based surges of interest in an event are associated with increases in expressed strength of feeling. Within the media, a well-established notion is that emotion is important in engaging attention, as expressed for violence by the common saying, "if it bleeds, it leads" (Kerbel, 2000; Seaton, 2005), and through evidence that audiences emotionally engage with the news (Perse, 1990). It seems logical, therefore, to hypothesise that events triggering large reactions in Twitter would be associated with increases in the strength of expressed sentiment, but there is no evidence yet for this hypothesis.

According to Alexa, and based upon its panel of toolbar users, Twitter had become the world's ninth most popular web site by October 2010 (alexa.com/topsites, accessed October 8, 2010), despite only beginning in July 2006. The rapid growth of the site may be partly due to celebrities tweeting regular updates about their daily lives (Johnson, 2009). Also according to Alexa, amongst Internet users people aged 25-44 were slightly over-represented in Twitter and those aged 55+ were much less likely to use it than average; women were also slightly over-represented (<http://www.alexa.com/siteinfo/twitter.com>, accessed October 8, 2010). Thus, despite the mobile phone connection Twitter is not a teen site, at least in the US: "Teens ages 12-17 do not use Twitter in large numbers, though high school-aged girls show the greatest enthusiasm" (Lenhart, Purcell, Smith, & Zickuhr, 2010).

2.3 FACEBOOK MINING

The Graph API is the primary way to get data in and out of Facebook's platform. It's a low-level HTTP-based API that you can use to query data, post new stories, manage ads, upload photos and a variety of other tasks that an app might need to do.

The Graph API is named after the idea of a 'social graph' - a representation of the information on Facebook composed of:

- **nodes** - basically "things" such as a User, a Photo, a Page, a Comment
- **edges** - the connections between those "things", such as a Page's Photos, or a Photo's Comments
- **fields** - info about those "things", such as a person's birthday, or the name of a Page

The Graph API is HTTP based, so it works with any language that has an HTTP library, such as cURL, urllib. We'll explain a bit more about what you can do with this in the section below, but it means you can also use the Graph API directly in your browser, for example a Graph API request is equivalent to:

```
GET graph.facebook.com
```

```
  /facebook/picture?
```

```
    redirect=false
```

Most Graph API requests require the use of access tokens which your app can generate by implementing Facebook Login.

This overview will show you how the Graph API can read and publish data to the social graph. We cover this fully in our Using Graph API guide, but in general you can read APIs by making HTTP GET requests to nodes or edges on those nodes.

Almost all requests are passed to the API at graph.facebook.com - the single exception is video uploads which use graph-video.facebook.com.

Each node has a unique **ID** which is used to access it via the Graph API. We specifically do not document any node/object ID structure or format because it is extremely likely to change over time and apps should not make assumptions based on current structure.

Here's how you'd use the ID to make a request for a node:

```
GET graph.facebook.com
```

```
  /{node-id}
```

or edge:

```
GET graph.facebook.com
```

`/ {node-id} / {edge-name}`

You can generally publish to APIs by making HTTP POST requests with parameters to the node:

POST graph.facebook.com


`/ {node-id}`

or edge:

POST graph.facebook.com

`/ {node-id} / {edge-name}`

The Graph API has multiple versions available to access at any one time. Each version contains a set of **core** fields and edge operations. We guarantee that those core APIs will be available and un-modified in that version for at least 2 years from release. The platform changelog can tell you which versions are currently available.

Certain operations such as publishing within an edge, or certain fields within a node, may be core without the entire edge or node being core. We annotate these core APIs using this  symbol within our Graph API reference docs.

Everything outside of these core APIs are called **extended** APIs. These APIs are still accessed through versioned paths, but they can potentially be modified or removed at any time, subject to 90-day migrations that would be announced on our platform roadmap. Alternatively they may simply be included in the next available API version.

You can read more about the intent of versioning in our guide, but here we'll explain how you make a call to a specific version of the Graph API.

Get Publishing Permissions

Next we'll try publishing something to Facebook using the Graph API. You would do this in your app only if you're building your own custom composer and not using one of the Share

Dialogs on web, iOS, or Android . The Facebook Share Dialogs don't require you to implement Facebook login or build your own interface for letting people share.

To explore publishing with the Graph API, click on that "Get Access Token" button again, and this time, choose publish_actions permission:

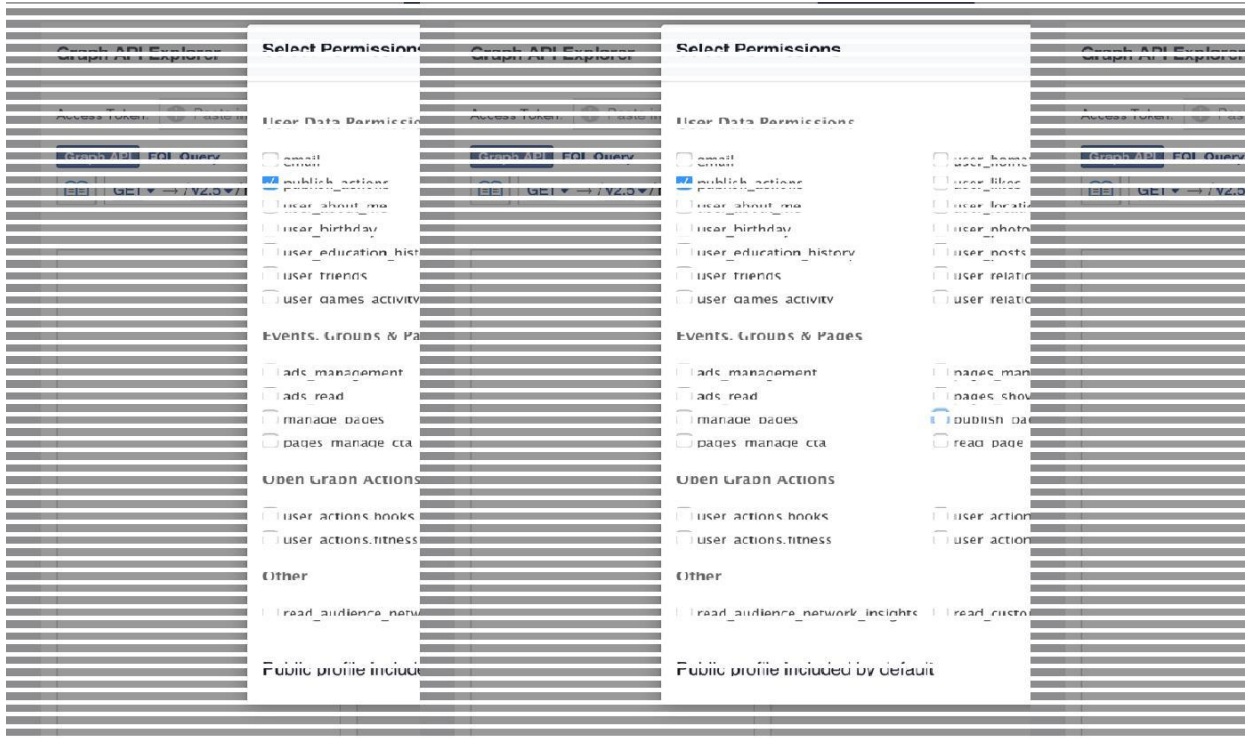


Figure 2.1 – Graph API for Social Mining

Now click on the blue "Get Access Token" button, and you'll see the Login Dialog again. Here you'll be asked for permission for the Graph API Explorer to post on your behalf. If you want you can change the audience here to 'Only Me' so that only you will be able to see the post, but you should accept this dialog and move onto the next step.

Publish a Post

If you've asked for publish_actions permission, now click on the button that says "GET" on it and choose "POST" from the dropdown selector that appears. Enter me/feed in the path field, and then click "Add a Field".

In the new fields that appear, put message as the "name", and Hello, World as the "value". It should look something like this:

Now click that blue "Submit" button, and after a few seconds the response panel should show something like:

```
{  
  "id": "{new-post-id}"  
}
```

This means that you've just published your first post via the Graph API! You can go to your profile, and you should be able to see the post there:

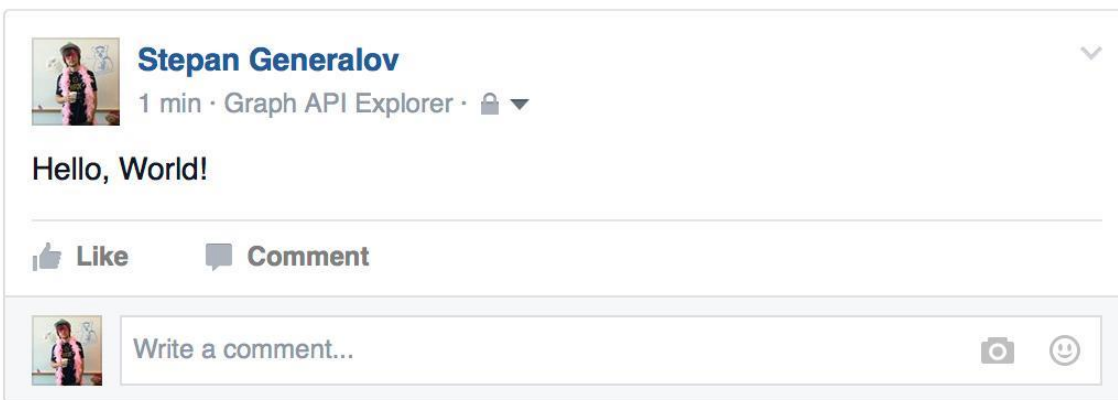


Figure 2.2 – Graph API Explorer

2.4 Twitter API

Twitter's API is very well documented and has a lot of useful functionality. It's especially useful for journalists who want to search Twitter for a term and either show or parse the results. Let's take a look at how we can easily do that.

Here is some simple example code that searches Twitter for the term "earthquake" and then creates a bulleted list for the tweets that are found. You can copy and paste the code and replace the word "earthquake" with whatever term you want to search for.

```
<html>  
<head>  
<script
```

```
type ="text/javascript"
```

```

src="https://ajax.googleapis.com/ajax/libs/jquery/1.6.2/jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function() {
    $.getJSON('http://search.twitter.com/search.json?q=earthquake&lang=en&callback=?',
function(data) {
    var data = data.results;
    var html = "<ul>";
    for(var i=0; i<data.length; i++) {
    html += "<li><a href='http://twitter.com/' + data[i].from_user + ">@"
+ data[i].from_user + "</a>: " + data[i].text + "</li>";
    }
    html += "</ul>"

    $('<div class="content">').html(html); });
});
</script>
</head>
<body>
<h2>Twitter</h2>
<div class="content">
</div>
</body>
</html>

```

If your development team uses open-source tools, there are some great libraries for parsing the Twitter API with much more advanced functionality than the snippet above. For PHP, use [TwitterOAuth](#). For Python, use [Tweepy](#). For Ruby, use [Grackle](#).

2.5 Google Maps API

Google Maps has an extensive API with many different features. You can use it to build maps, geolocate tweets (or other pieces of data with latitude and longitude variables), search for local

schools, or even measure elevation or distance. The API has a lot of documentation with several examples. Depending on what you're trying to do, you can find code snippets to help you achieve your goal.

Here are a few good examples from the documentation:

- Follow the Google Maps "Hello World" tutorial to see how easy it is to get a map up and running.
- Use the Google Places API to track information such as the density of schools in different neighborhoods.
- Build a Twitter location application using the Google Geocoding API.

There are also numerous wrappers for the Google Maps API in PHP, Python and Ruby, so consider reaching out to your development team for advice on how to integrate Google Maps with the apps they're already building.

2.6 Facebook API

The Facebook Graph API is JSON-enabled and has search functionality similar to Twitter. You can search for who is posting publicly about topics, search Facebook Places, and see photos and videos posted publicly or by your page's followers.

Facebook has a fairly extensive section of code examples that you can use and tweak to your liking. Some highlights are:

- The ability to create a feed of your friend's likes.
- The ability to show photos of your friends who like a particular story or article.
- An API Explorer, which enables you to see relationships between graph data and gather information for your beat or a particular story.

Other noteworthy APIs

The folks at Participatory Politics and Sunlight Labs have built a series of amazing APIs with access to government data -- a notoriously tricky task. They have a new project called Open

Government that aims to help users track numerous government data sets pertaining to politicians, bills, campaign donations and voting -- all the way down to the local level.

Although they don't have support for every state yet, Open Government is under active development, so more states will be added over time. The code they're using is all open-source and built in Ruby.

Developers at the Chicago Tribune and The New York Times regularly blog and share their own APIs. Check out what projects they're working on, and build and release your own.

Experimenting with APIs is just one of many ways to build your skills as a digital journalist. The more you know about the open-source culture, the more you can effectively share data, collaborate with others in the newsroom and, perhaps most importantly, tell innovative stories.

Using Facebook's Graph API Explorer to retrieve Page Insights data

In the world of APIs, an "API Explorer" is an interface that helps you craft a request URL. This URL is a like a command line that tells Facebook to do something on your behalf. For instance, performing a GET request to pull data from your Facebook profile.

2.7 INSTAGRAM MINING

Step 1:

Each API needs registration before you can use its features. Please note, for showing images under a specific hashtag, we don't need any authentication but for connecting to Instagram API we do need client ID and client secret keys which we will get once we will register an app on Instagram API platform.

So our first step will be to register an app on Instagram API page <https://instagram.com/developer/> and click on "Register Your Application" button.

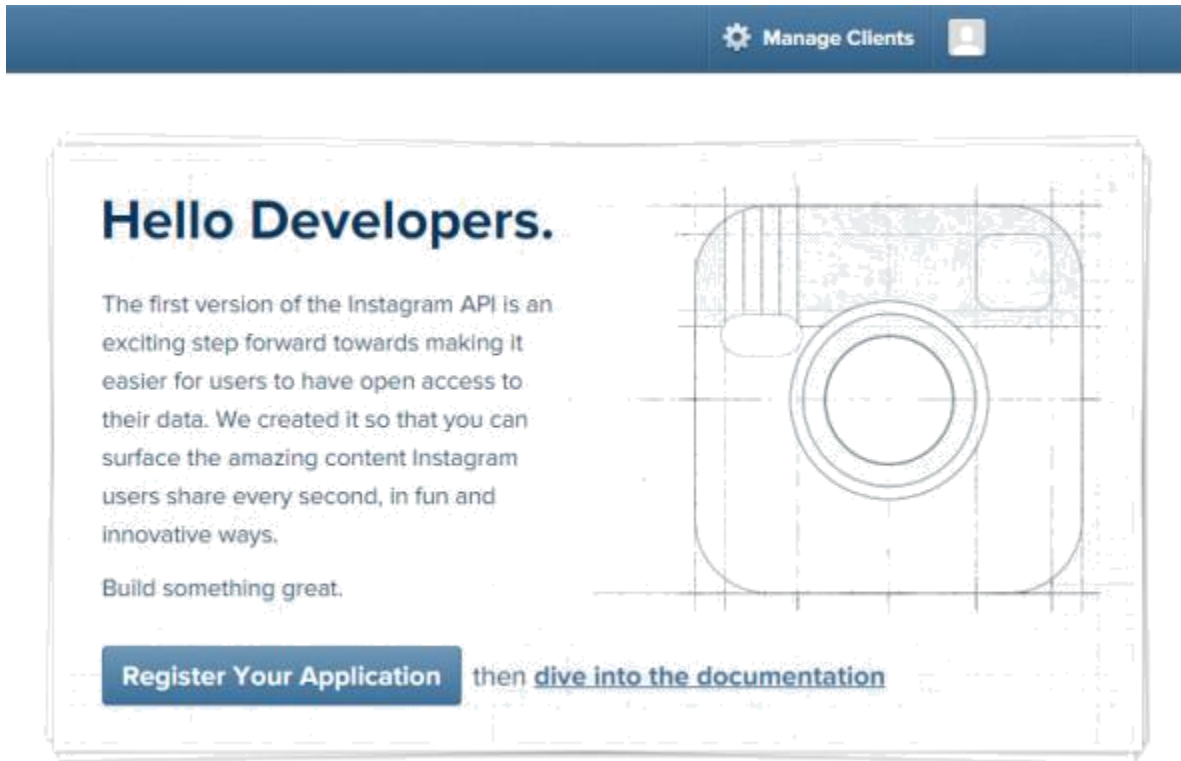


Figure 2.3 – Instagram Mining

Step 2:

Next page is Manage Clients page, where we need to add a client to get the required authentication keys, i-e Client ID and Client Secret. Please click on “Register New Client” button to see the client registration form.

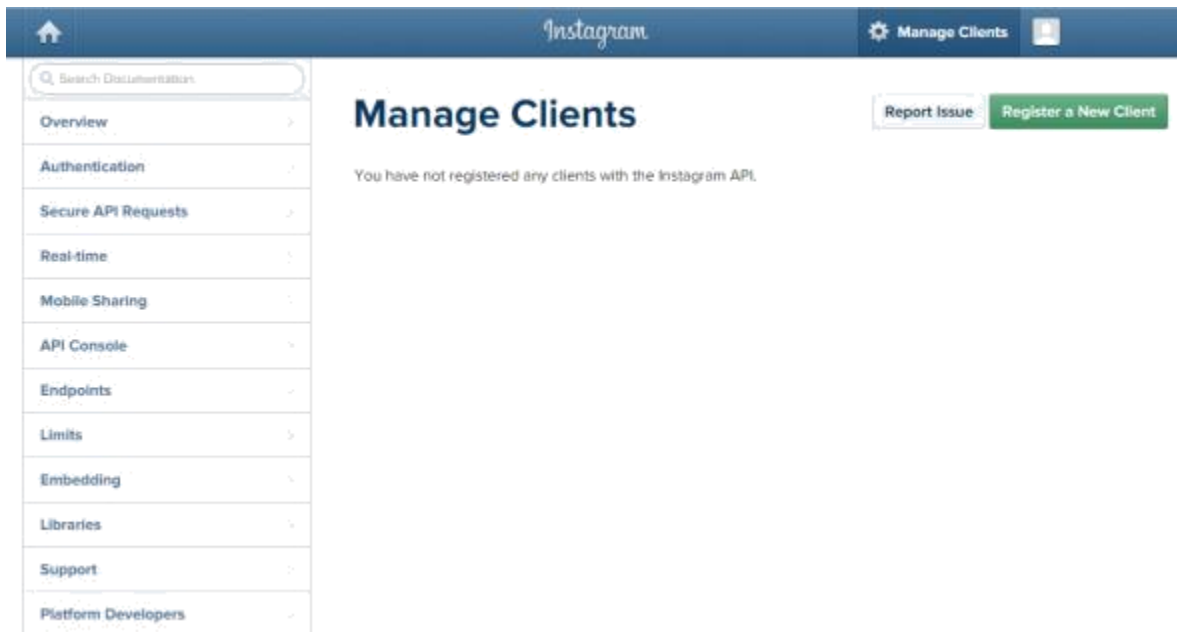


Figure 2.3 – Clients Management in Instagram Mining

Step 3:

In this form , we will provide app name which should represent your website or company name for easily understandable for the end users. I used “99Points Hashtag” here as an example. Provide a short but clear description to let user know what is this app for. Add website and redirect URL. Both URLs can be same as we don’t need it here for showing the images only. Enter the information and click on “Register” button.

The screenshot shows the 'Register new Client ID' page in the Instagram Developer Portal. The page has a dark blue header with the Instagram logo on the left, a 'Manage Clients' button with a gear icon in the center, and a user profile icon on the right. Below the header, the main heading is 'Register new Client ID'. There are two tabs: 'Basic' (selected) and 'Security'. The form contains the following fields:

- Application Name:** A text input field containing 'Hashtag 99Points'.
- Description:** A text area containing 'Just an app to show images'.
- Website URL:** A text input field containing 'http://www.99points.info/'.
- Redirect URI(s):** A text input field containing 'http://www.99points.info/ x'.

Below the 'Application Name' field, there is a note: 'Do not use *Instagram*, *IG*, *insta* or *gram* in your app name. Make sure to adhere to the API Terms of Use and Brand Guidelines.'

Below the 'Redirect URI(s)' field, there is a note: 'The `redirect_uri` specifies where we redirect users after they have chosen whether or not to authenticate your application.'

At the bottom of the form, there are two buttons: a green 'Register' button and a grey 'Cancel' button.

Figure 2.4 – Client Registration in Instagram Mining

Step 4:

The next screen shows the success message with Client ID and Client Secret keys which we will use in our last step.

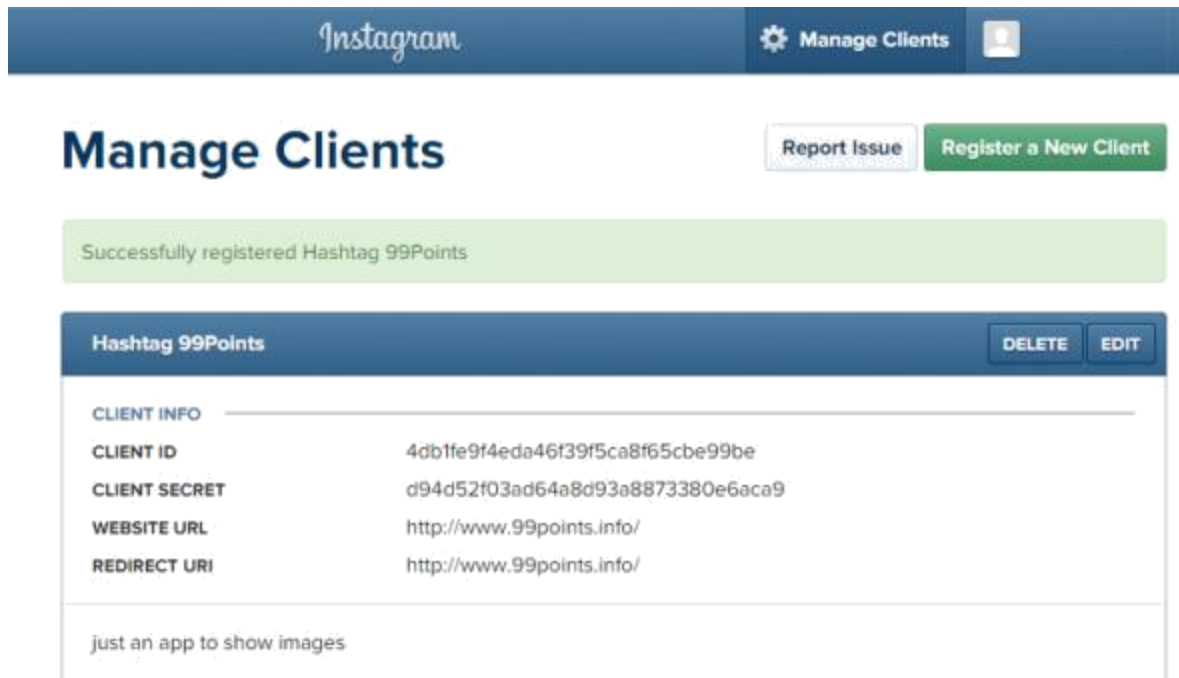


Figure 2.5 – Managing Clients and Keys in Instagram

Step 5:

Finally, we have created the app on the Instagram, now its LANG-Action-Time. The below API url is a direct way to bring all the images having a specific hashtag. It takes hashtag and client ID which we already have above and then it brings the results in JSON format.

```
1 "https://api.instagram.com/v1/tags/???/media/recent?client_id=???"
```

Just put your tag and client ID in the URL and you will see all images but wait we need to read all images and for this we will parse this url from a Curl function and then we will decode the JSON result.

That's where we need a curl function to execute the api URL. You can use this function any place where you need to execute or access an external URL within from our LANG website.

```
1 function processURL($url)
2 {
```

```

3  $ch = curl_init();
4  curl_setopt_array($ch, array(
5  CURLOPT_URL => $url,
6  CURLOPT_RETURNTRANSFER => true,
7  CURLOPT_SSL_VERIFYPEER => false,
8  CURLOPT_SSL_VERIFYHOST => 2
9  ));
10
11 $result = curl_exec($ch);
12 curl_close($ch);
13 return $result;
14 }

```

Next thing is to build our API url. We have now added tag and client ID and will execute the curl function.

```

1 $tag = '99points';
2 $client_id = "4db1fe9f4eda46f39f5ca8f65cbe99be";
3 $url = 'https://api.instagram.com/v1/tags/'.$tag.'/media/recent?client_id='.$client_id;

```

Just pass the above line to our curl function which will return us JSON encoded output and we can easily decode that using LANG “json_decode” function as shown below. You can also print the result array to see what type of data it contains.

```

1 $all_result = processURL($url);
2 $decoded_results = json_decode($all_result, true);
3 //echo '<pre>';
4 //print_r($decoded_results);
5 //exit;

    foreach($decoded_results['data'] as $item){
1  $image_link = $item['images']['thumbnail']['url'];
2  echo '';

```

3 }

4

For better understanding, we can print the \$decoded_results array which will have all the data it brought.

```
[meta] => Array
(
    [code] => 200
)
[data] => Array
(
    [0] => Array
        (
            [attribution] =>
            [videos] => Array
                (
                    [low_bandwidth] => Array
                        (
                            [url] => https://scontent.cdninstagram.com/hphotos-xfa1/t50.2886-16/11175455_442022525965975_275459143_s.mp4
                            [width] => 480
                            [height] => 480
                        )
                    [standard_resolution] => Array
                        (
                            [url] => https://scontent.cdninstagram.com/hphotos-xaf1/t50.2886-16/11190400_891075790935678_1364235018_n.mp4
                            [width] => 640
                            [height] => 640
                        )
                    [low_resolution] => Array
                        (
                            [url] => https://scontent.cdninstagram.com/hphotos-xfa1/t50.2886-16/11175455_442022525965975_275459143_s.mp4
                            [width] => 480
                            [height] => 480
                        )
                )
            [tags] => Array
                (
                    [0] => rusty
                    [1] => php
                    [2] => sleepy
                    [3] => goodnight
                    [4] => tired
                )
        )
)
```

```
1 function processURL($url)
2 {
3     $ch = curl_init();
4     curl_setopt_array($ch, array(
5         CURLOPT_URL => $url,
6         CURLOPT_RETURNTRANSFER => true,
7         CURLOPT_SSL_VERIFYPEER => false,
8         CURLOPT_SSL_VERIFYHOST => 2
9     ));
10
11     $result = curl_exec($ch);
12     curl_close($ch);
```

```
13     return $result;
14 }
15
16 $tag = '99points';
17 $client_id = "4db1fe9f4eda46f39f5ca8f65cbe99be";
18 $url = 'https://api.instagram.com/v1/tags/'.$tag.'/media/recent?client_id='.$client_id;
19
20 $all_result = processURL($url);
21 $decoded_results = json_decode($all_result, true);
22
23 // echo '<pre>';
24 // print_r($decoded_results);
25 // exit;
26
27 //Now parse through the $results array to display your results...
28 foreach($decoded_results['data'] as $item){
29     $image_link = $item['images']['thumbnail']['url'];
30     echo '';
31 }
```

In our upcoming tutorials, we will talk more about Instagram API and will show you how to build your own small Instagram application using its API.

3.1 PROPOSED MODEL

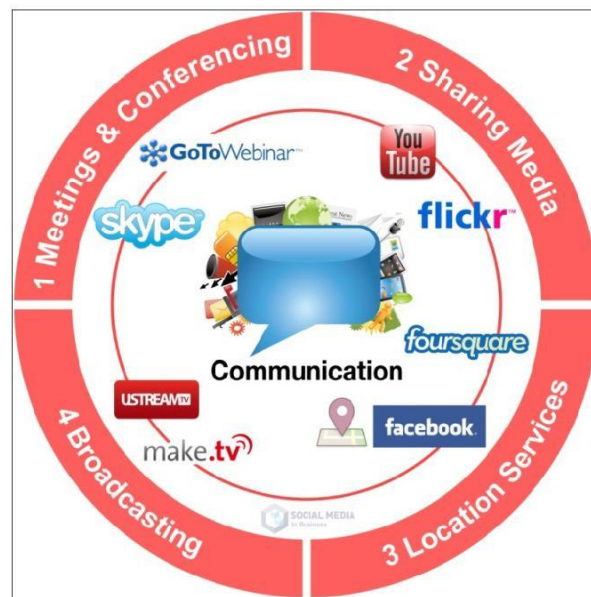
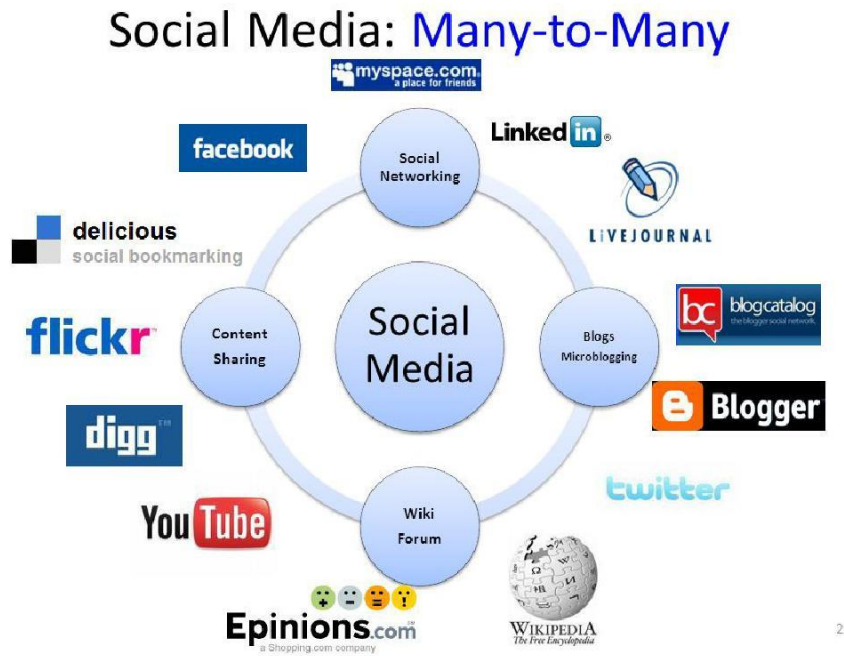


Figure 2.1 – Machine Learning and Data Modeling Process

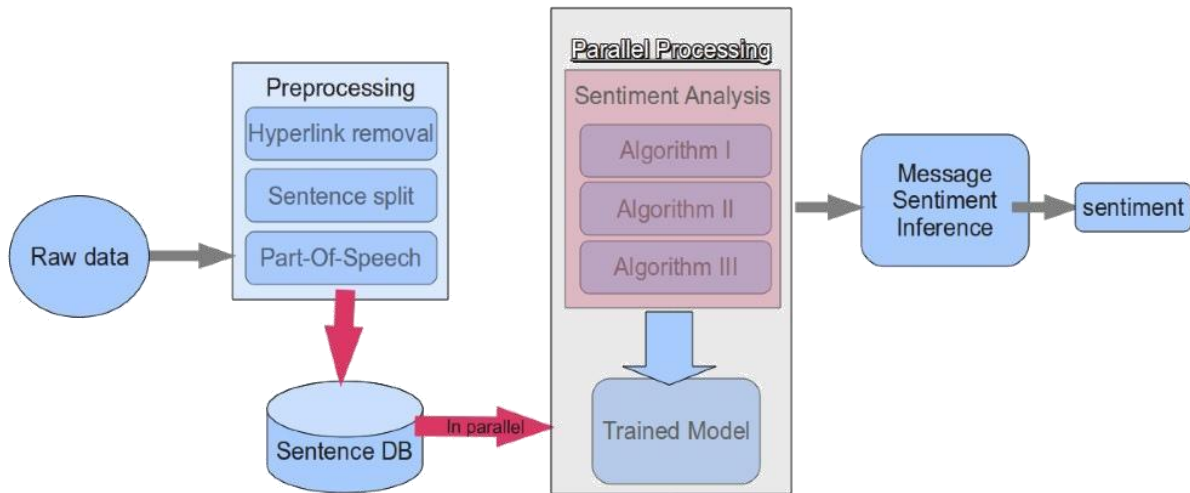


Figure 2.2 – Sentiment Analysis

3.2 HARDWARE REQUIREMENTS

The minimum requirements needed to perform operations are

- Intel Pentium Processor at 2 GHz or Higher
- RAM 256MB or more
- Hard disk capacity 10GB or more

3.3 SOFTWARE REQUIREMENTS

The software required to perform the implementation are

- Windows or Linux Operating System (Ubuntu, Fedora)
- Advance Java
- Twitter APIs
- Instagram API
- Facebook API
- AJAX
- MySQL Database Engine
- Notepad++

- Twitter4J
- Eclipse IDE
- JSON (JavaScript Object Notation)
- WEKA – Data Mining and Machine Learning Tool

CHAPTER 4

IMPLEMENTATION SETUP

Many social networks and apps have their own interface that programmers can work with. These interfaces are called APIs (short for Application Programming Interface).

4.1 FACEBOOK APPLICATION MODEL

Before going into the actual implementation, let's understand the Facebook application model. Facebook opens its platform to developers using REST web services. As a developer, you are free to use the APIs of your choice to integrate Facebook features in your application. You are also free to use the technology of your choice. Facebook uses a proxy server model as a main integration point. The Facebook proxy server follows the following steps:

The web application will reside in your web/application server and you need to register the base URL in a Facebook account.

When the application is visited in Facebook, it will call the registered URL on the application server.

The application will call necessary Facebook APIs to get the relevant information. Your application uses its own database data and Facebook data and render it After this Facebook returns your application's output to the user

Before we integrate Facebook with your Java application we need some third party libraries. These third party libraries will help you with integration and communication with Facebook (actually accessing Facebook application installed in their server). Different independent groups of Java and open source developers have made efficient Facebook libraries for integration purpose.

You can check <http://code.google.com/p/Facebook-java-api> to get more details about the APIs and Jars. These third party libraries are compatible with Java SE 5 and above.

4.2 FACEBOOK JAR FILES

<http://Facebook-java-api.googlecode.com/files/Facebook-java-api-1.7.2.jar>
<http://Facebook-java-api.googlecode.com/files/json-1.0.jar>
<http://Facebook-java-api.googlecode.com/files/Facebook-util-1.7.2.jar>

If you are using a Java application server on top of Java SE 5, then you should download the following JAR files:

<http://Facebook-java-api.googlecode.com/files/jaxb-api-2.1.jar>
<http://Facebook-java-api.googlecode.com/files/jaxb-impl-2.1.jar>
<http://Facebook-java-api.googlecode.com/files/jsr173-api-1.0.jar>

The three JARs above are not required if you are using Java SE 6 or later.

After downloading these JARs, you need to incorporate them in your web or stand alone application. These JARs are made to provide API access to the client application for different purpose.

The following example will show the integration part:

```
package com.home.social;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;

import net.sf.json.JSONObject;
import net.sf.json.JsonConfig;
```

```

import Facebook4j.Facebook;
import Facebook4j.FacebookException;
import Facebook4j.FacebookFactory;
import Facebook4j.Post;
import Facebook4j.ResponseList;
import Facebook4j.conf.Configuration;
import Facebook4j.conf.ConfigurationBuilder;

public class FacebookIntegration {

    public static void main(String[] args) throws FacebookException {
        // Create conf builder and set authorization and access keys
        ConfigurationBuilder configurationBuilder = new
        ConfigurationBuilder(); configurationBuilder.setDebugEnabled(true);
        configurationBuilder.setOAuthAppId("xxxx");
        configurationBuilder.setOAuthAppSecret("xxxxx");
        configurationBuilder.setOAuthAccessToken("xxxx");
        configurationBuilder
            .setOAuthPermissions("email, publish_stream, id, name,
first_name, last_name, read_stream , generic");
        configurationBuilder.setUseSSL(true);
        configurationBuilder.setJSONStoreEnabled(true);

        // Create configuration and get Facebook instance
        Configuration configuration = configurationBuilder.build();
        FacebookFactory ff = new FacebookFactory(configuration);
        Facebook Facebook = ff.getInstance();

        try {
            // Set search string and get results
            String searchPost = "MACDonaldsIndia";

```

```

Date date = new Date();
SimpleDateFormat simpleDateFormat = new SimpleDateFormat(
    "dd_MM_yyyy_hh_mm");
String filename = "D:\\FacebookConfigFolder\\File\\" + searchPost
    + "_" + simpleDateFormat.format(date) + ".txt";
String results = getFacebookPostes(Facebook, searchPost);
File file = new File(filename);
if (!file.exists()) {
    file.createNewFile();
    FileWriter fw = new
    FileWriter(file.getAbsolutePath()); BufferedWriter bw
    = new BufferedWriter(fw); bw.write(results);
    bw.close();
    System.out.println("Completed");
}
} catch (IOException e) {
    e.printStackTrace();
}
}

```

// This method is used to get Facebook posts based on the search string set
// above

```

public static String getFacebookPostes(Facebook Facebook, String searchPost)
    throws FacebookException {
    String searchResult = "Item : " + searchPost + "\n";
    StringBuffer searchMessage = new StringBuffer();
    ResponseList<Post> results =
    Facebook.getPosts(searchPost); for (Post post : results) {
        System.out.println(post.getMessage());
        searchMessage.append(post.getMessage() + "\n");
        for (int j = 0; j < post.getComments().size(); j++) {

```

```

        searchMessage.append(post.getComments().get(j).getFrom()
            .getName()
            + “, “);
        searchMessage.append(post.getComments().get(j).getMessage()
            + “, “);

searchMessage.append(post.getComments().get(j).getCreatedTime()
            + “, “);

        searchMessage.append(post.getComments().get(j).getLikeCount()
            + “\n”);
    }
}

String feedString = getFacebookFeed(Facebook, searchPost);
searchResult = searchResult + searchMessage.toString();
searchResult = searchResult + feedString;
return searchResult;
}

```

// This method is used to get Facebook feeds based on the search string set
// above

```

public static String getFacebookFeed(Facebook Facebook, String searchPost)
    throws FacebookException {
    String searchResult = “”;
    StringBuffer searchMessage = new StringBuffer();
    ResponseList<Post> results =
    Facebook.getFeed(searchPost); for (Post post : results) {
        System.out.println(post.getMessage());
        searchMessage.append(post.getFrom().getName() + “, “);
        searchMessage.append(post.getMessage() + “, “);
        searchMessage.append(post.getCreatedTime() + “\n”);
    }
}

```

```

        searchResult = searchResult + searchMessage.toString();
        return searchResult;
    }

    // This method is used to create JSON object from data string
    public static String stringToJson(String data) {
        JsonConfig cfg = new JsonConfig();
        try {
            JSONObject jsonObject = JSONObject.fromObject(data, cfg);
            System.out.println("JSON = " + jsonObject.toString());
        } catch (Exception e) {
            e.printStackTrace();
        }
        return "JSON Created";
    }
}

```

4.3 INTEGRATION WITH TWITTER

In order to integrate your application with Twitter, we need to use the library – Twitter4j. This is a well documented library that helps any Java developer to integrate his/her application with Twitter. As a developer you need to follow certain steps:

Send a request to Twitter asking for a token. This request should carry both a consumer key and a secret key.

Store the response received from Twitter.

Once the response is received, the authentication URL is extracted from the response. The user then needs to be redirected to the authentication URL, so that he can sign in. User signs in and gets a Personal Identification Number or PIN.

User then enters PIN in the application.

Once the PIN is entered the application should ask Twitter for a security token, providing consumer parameters as above as well as the previously stored request token and the PIN.

Once the token is received, every request going to Twitter should have this token along with the PIN

```
package com.home.social;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.util.List;

import Twitter4j.Query;
import Twitter4j.QueryResult;
import Twitter4j.Status;
import Twitter4j.Twitter;
import Twitter4j.TwitterFactory;
import Twitter4j.conf.ConfigurationBuilder;

public class TwitterIntegration {
    public static void main(String[] args) throws Exception{
        // Create configuration builder and set key, token etc
        ConfigurationBuilder cb = new ConfigurationBuilder();
        cb.setOAuthConsumerKey("xxx");
        cb.setOAuthConsumerSecret("xxxx");
        cb.setOAuthAccessToken("xxxxx");
        cb.setOAuthAccessTokenSecret("xxxx");

        // Create Twitter instance
        Twitter Twitter = new TwitterFactory(cb.build()).getInstance();

        // Create file writer and buffer writer
        FileWriter fstream = new FileWriter("Twitterstream.txt",true);
```

```

BufferedWriter out = new BufferedWriter(fstream);

// Create Query object and set search
string Query query = new Query("");
query.setQuery("#USAirways");

// Get query result
QueryResult qr = Twitter.search(query);

// Get tweets and write in the file
while(qr.hasNext()){
    qr.nextQuery();
    List<Status> tweets = qr.getTweets();

    for (Status t: tweets){
        System.out.println(t.getId() + " - " + t.getCreatedAt() + ": " +
t.getText());

        out.write("\n"+t.getId()+");");
        out.write("\t"+t.getText()+");");
        out.write("\t"+t.getUser()+");");
    }
}
try{
    Thread.sleep(1000*60*15);
}catch(Exception e) {}
}
}

```

Table 5.1 – Comparison of Performance of Facebook and Twitter API

Keyword Length	Facebook API	Twitter API
3	0.0273738	0.0128833
4	0.0331	0.0211
5	0.09333	0.0866373
6	0.06788	0.0567272
7	0.089999	0.083333
8	0.1292393	0.10112
More than 8		

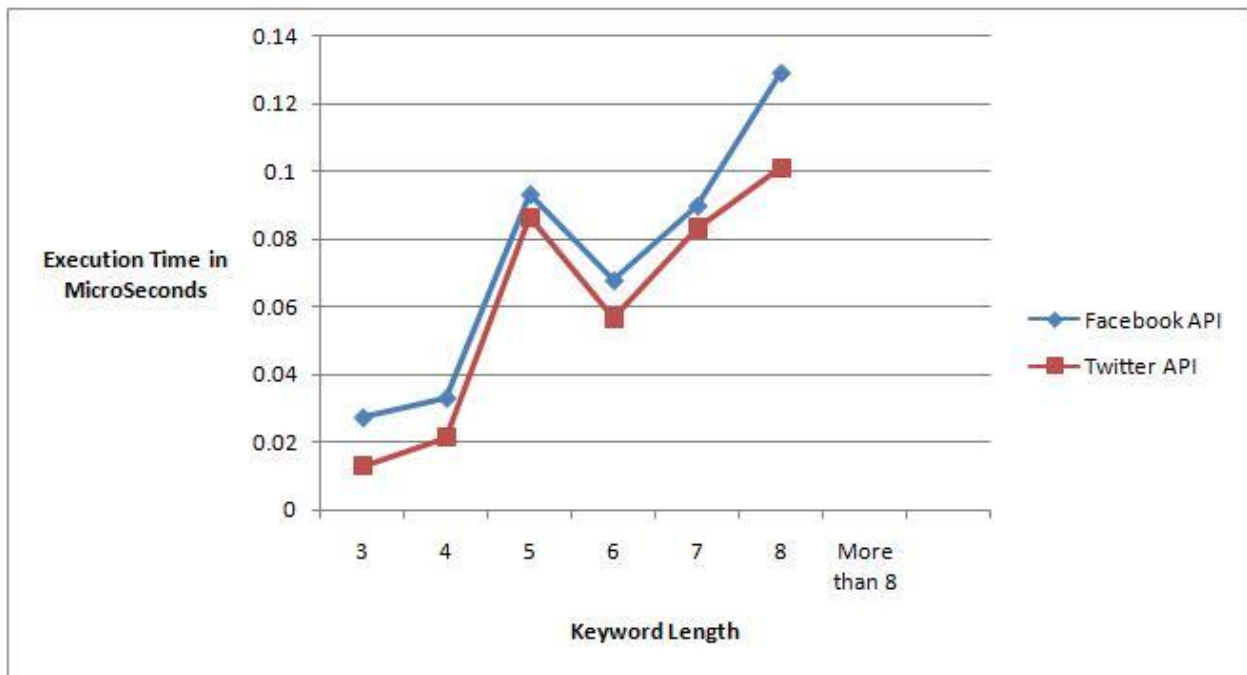


Figure 4.1 – Comparative Analysis between FACEBOOK and TWITTER Analytics

CHAPTER 5

CONCLUSION

Opinion mining is a type of natural language processing for tracking the mood of the public about a particular product. Opinion mining, which is also called sentiment analysis, involves building a system to collect and categorize opinions about a product. Automated opinion mining often uses machine learning, a type of artificial intelligence (AI), to mine text for sentiment. Opinion mining can be useful in several ways. It can help marketers evaluate the success of an ad campaign or new product launch, determine which versions of a product or service are popular and identify which demographics like or dislike particular product features. For example, a review on a website might be broadly positive about a digital camera, but be specifically negative about how heavy it is. Being able to identify this kind of information in a systematic way gives the vendor a much clearer picture of public opinion than surveys or focus groups do, because the data is created by the customer. There are several challenges in opinion mining. In this work the opinion mining and comparative analysis is done using Facebook and Twitter APIs and performed the pragmatic review.

REFERENCES / BIBLIOGRAPHY

- [1] Cataldi, M., Caro, L. D., & Schifanella, C. (2010). Emerging topic detection on Twitter based on temporal and social terms evaluation In Proceedings of the Tenth International Workshop on Multimedia Data Mining table of contents (pp. A4). New York, NY: ACM Press.
- [2] Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, K. P. (2010). Measuring user influence in Twitter: The million follower fallacy. In Proceedings of the International AAAI Conference on Weblogs and Social Media (pp. Retrieved August 9, 2010 from: http://an.kaist.ac.kr/~mycha/docs/icwsm2010_cha.pdf).
- [3] Choudhury, M. D., Sundaram, H., John, A., & Seligmann, D. D. (2008). Can blog communication dynamics be correlated with stock market activity? In Proceedings of the Nineteenth ACM conference on Hypertext and Hypermedia (pp. 55-60). New York, NY: ACM Press.
- [4] Culotta, A. (2010). Detecting influenza outbreaks by analyzing Twitter messages. arxiv.org, Retrieved September 1, 2010 from: http://arxiv.org/PS_cache/arxiv/pdf/1007/1007.4748v2011.pdf.
- [5] Diakopoulos, N. A., & Shamma, D. A. (2010). Characterizing debate performance via aggregated twitter sentiment. In Proceedings of the 28th international conference on Human factors in computing systems (pp. 1195-1198). New York, NY: ACM Press.
- [6] Dodds, P. S., & Danforth, C. M. (2010). Measuring the happiness of large-scale written expression: Songs, blogs, and presidents. *Journal of Happiness Studies*, 11(4), 441-456.
- [7] Efron, M. (2010). Hashtag retrieval in a microblogging environment. In Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval (pp. 787-788). New York, NY: ACM Press.
- [8] Enyon, R., Schroeder, R., & Fry, J. (2009). New techniques in online research: Challenges for research ethics. *21st Century Society*, 4(2), 187-199.
- [9] Fox, E. (2008). *Emotion science*. Basingstoke: Palgrave Macmillan.
- [10] Gamon, M., Aue, A., Corston-Oliver, S., & Ringger, E. (2005). Pulse: Mining customer opinions from free text. *Lecture Notes in Computer Science*, 3646, 121-132.

- [11] Allan, J., Papka, R., & Lavrenko, V. (1998). On-line new event detection and tracking In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval (pp. 37-45). New York, NY: ACM Press.
- [12] Archak, N., Ghose, A., & Ipeirotis, P. G. (2007). Show me the money!: Deriving the pricing power of product features by mining consumer reviews. In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 56-65). New York, NY: ACM Press.
- [13] Balog, K., Mishne, G., & Rijke, M. d. (2006). Why are they excited? Identifying and explaining spikes in blog mood levels. 11th Meeting of the European Chapter of the Association for Computational Linguistics (EACL 2006), Retrieved July 8, 2010 from: <http://staff.science.uva.nl/~mdr/Publications/Files/eacl2006-moodsignals.pdf>.
- [14] Bansal, N., & Koudas, N. (2007). BlogScope: a system for online analysis of high volume text streams. In Proceedings of the 33rd international conference on Very large data bases (pp. 1410-1413). New York, NY: ACM Press.
- [15] Bassett, E. H., & O'Riordan, K. (2002). Ethics of Internet research: Contesting the human subjects research model *Ethics and Information Technology*, 4(3), 233-247.
- [16] Bifet, A., & Frank, E. (2010). Sentiment knowledge discovery in Twitter streaming data. In Proc 13th International Conference on Discovery Science (pp. 1-15). Berlin: Springer.
- [17] Blumler, J. G., & Katz, E. (1974). The uses of mass communications: Current perspectives on gratifications research. Beverly Hills, CA: Sage.
- [18] Bollen, J., Pepe, A., & Mao, H. (2009). Modeling public mood and emotion: Twitter sentiment and socioeconomic phenomena. arXiv.org, arXiv:0911.1583v0911 [cs.CY] 0919 Nov 2009.
- [19] boyd, d., Golder, S., & Lotan, G. (2009). Tweet, tweet, retweet: Conversational aspects of retweeting on Twitter. Proceedings of HICSS-43, Retrieved November 12, 2009 from: <http://www.danah.org/papers/TweetTweetRetweet.pdf>.
- [20] Bruza, P., & Weeber, M. (2008). Literature-based discovery. Berlin: Springer.
- [21] Case, D. O. (2002). Looking for information: A survey of research on information seeking, needs, and behavior. San Diego, CA: Academic Press.

- [22] Gaver, W. W. (1991). Technology affordances. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 79-84). New York: ACM.
- [23] Gibson, J. J. (1977). The theory of affordances. In R. Shaw & J. Bransford (Eds.), *Perceiving, acting, and knowing: Toward an ecological psychology* (pp. 62-82). Hillsdale, NJ: Lawrence Erlbaum Associates.
- [24] Gibson, J. J. (1986). *The ecological approach to visual perception*. Hillsdale, NJ: Lawrence Erlbaum.
- [25] Gilbert, E., & Karahalios, K. (2010). Widespread worry and the stock market. Fourth International AAAI Conference on Weblogs and Social Media, Retrieved August 9, 2010 from: <http://www.aaai.org/ocs/index.php/ICWSM/ICWSM2010/paper/view/1513/1833>.
- [26] Gruhl, D., Chavet, L., Gibson, D., Meyer, J., & Pattanayak, P. (2004). How to build a WebFountain: An architecture for very large-scale text analytics. *IBM Systems Journal*, 43(1), 64-77.
- [27] Gruhl, D., Guha, R., Kumar, R., Novak, J., & Tomkins, A. (2005). The predictive power of online chatter. In R. L. Grossman, R. Bayardo, K. Bennett & J. Vaidya (Eds.), *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (pp. 78-87). New York, NY, USA: ACM Press.
- [28] Gruhl, D., Guha, R., Liben-Nowell, D., & Tomkins, A. (2004). Information diffusion through Blogspace, WWW2004. New York, Retrieved July 5, 2010 from: <http://www.tackelberz.de/wp-content/uploads/2009/10/5-gruhl.pdf>.
- [29] Guralnik, V., & Srivastava, J. (1999). Event detection from time series data. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 33-42). New York, NY: ACM Press.
- [30] Hamilton, J. D. (1994). *Time series analysis*. Princeton, NJ: University Press
- [31] Honeycutt, C., & Herring, S. C. (2009). Beyond microblogging: Conversation and collaboration via Twitter. In *Proceedings of the 42nd Hawaii International Conference on System Sciences* (pp. 1-10). Los Alamitos, CA: IEEE.
- [32] Hookway, N. (2008). Entering the 'blogosphere': some strategies for using blogs in social research. *Qualitative Research*, 8(1), 91-113.

- [33] Huang, J., Thornton, K. M., & Efthimiadis, E. N. (2010). Conversational tagging in twitter. In Proceedings of the 21st ACM conference on Hypertext and hypermedia (pp. 173-178). New York, NY: ACM Press.
- [34] Katz, E. (1959). Mass communication research and the study of popular culture. *Studies in Public Communication*, 2(1), 1-6.
- [35] Kerbel, M. (2000). *If it bleeds, it leads An anatomy of television news*. Boulder, CO: Westview Press.
- [36] Kinsinger, E. A., & Schacter, D. L. (2008). Memory and emotion. In M. Lewis, J. A. Haviland-Jones & L. Feldman Barrett (Eds.), *Handbook of emotions* (3 ed) (pp. 601-617). New York, NY: The Guildford Press.
- [37] Kramer, A. D. I. (2010). An unobtrusive behavioral model of "Gross National Happiness". *Proceedings of CHI 2010*, 287-290.
- [38] Krishnamurthy, B., Gill, P., & Arlitt, M. (2008). A few chirps about twitter. In *Proceedings of the first workshop on Online social networks table of contents*: (pp. 19-24). New York, NY: ACM Press.
- [39] Kumar, R., Novak, J., Raghavan, P., & Tomkins, A. (2003). On the bursty evolution of blogspace. In *WWW2003*. Budapest, Hungary, <http://www2003.org/cdrom/papers/refereed/p477/p477-kumar/p477-kumar.htm>.
- [40] Kwak, H., Lee, C., Park, H., & Moon, S. (2010). What is Twitter, a social network or a news media? In *Proceedings of the 19th international conference on world wide web* (pp. 591-600). New York, NY: ACM Press.