FILIET: An Information Extraction System

for Filipino Disaster-Related Reports

Thesis Document

Presented to

the Faculty of the College of Computer Studies

De La Salle University Manila

In Partial Fulfillment

of the Requirements for the Degree of

Bachelor of Science in Computer Science

by

DELA CRUZ, Kyle Mc Hale B.

GARCIA, John Paul F.

KALAW, Kristine Ma. Dominique F.

LU, Vilson E.

REGALADO, Ralph Vincent

Adviser

August 9, 2014

**Abstract**

The Philippines, being a disaster-prone country and the social media capital of the world, uses the social media to report the current status of their areas. Given these sources of information, relevant details could be extracted and utilized in order to provide more information for decision-makers. This could help the government and other institutions in deciding where to deploy relief resources. However, social media comes in different forms such as news feeds, blogs, and social networking sites. This means that it is harder to extract information due to the lack of structure. Also another problem lies with the nature of the Filipino language being morphologically rich and having other variations, thus, makes it difficult for the system to extract information. The information extraction system must be able to extract information from various sources like social networking sites, news feeds, and blogs, taking into consideration the different variations of the Filipino languages.

Keywords: Information extraction, disaster management, social media

**Table of Contents**

[1.0 Research Description 1-1](#_Toc395181674)

[1.1 Overview of the Current State of Technology 1-1](#_Toc395181675)

[1.2 Research Objectives 1-2](#_Toc395181676)

[1.2.1 General Objective 1-2](#_Toc395181677)

[1.2.2 Specific Objectives 1-2](#_Toc395181678)

[1.3 Scope and Limitations of the Research 1-3](#_Toc395181679)

[1.4 Significance of the Research 1-4](#_Toc395181680)

[1.5 Research Methodology 1-4](#_Toc395181681)

[1.5.1 Investigation and Research Analysis 1-5](#_Toc395181682)

[1.5.2 System Design 1-5](#_Toc395181683)

[1.5.3 Sprints 1-5](#_Toc395181684)

[1.5.4 Sprint Planning Meetings 1-6](#_Toc395181685)

[1.5.5 Scrum Meetings 1-6](#_Toc395181686)

[1.5.6 System Development 1-6](#_Toc395181687)

[1.5.7 System Integration and Testing 1-6](#_Toc395181688)

[1.5.8 System Evaluation 1-6](#_Toc395181689)

[1.5.9 Documentation 1-6](#_Toc395181690)

[1.5.10 Calendar of Activities 1-7](#_Toc395181691)

[2.0 Review of Related Works 2-1](#_Toc395181692)

[2.1 Machine Learning-Based Information Extraction Systems 2-1](#_Toc395181693)

[2.2 Rule-Based Information Extraction Systems 2-2](#_Toc395181694)

[2.3 Other Information Extraction Systems 2-4](#_Toc395181695)

[3.0 Theoretical Framework 3-1](#_Toc395181696)

[3.1 Information Extraction 3-1](#_Toc395181697)

[3.1.1 Information Extraction Modules 3-2](#_Toc395181698)

[3.2 Information Extraction Architecture 3-6](#_Toc395181699)

[3.2.1 Template-Based Architecture 3-6](#_Toc395181700)

[3.2.2 Adaptive Architecture 3-7](#_Toc395181701)

[3.3 Twitter 3-14](#_Toc395181702)

[3.3.1 Use of Twitter 3-14](#_Toc395181703)

[3.3.2 Twitter and Disasters 3-14](#_Toc395181704)

[3.4 Evaluation Metrics 3-16](#_Toc395181705)

[3.4.1 F-measure 3-16](#_Toc395181706)

[3.4.2 Kappa Statistics 3-17](#_Toc395181707)

[3.5 Tools 3-17](#_Toc395181708)

[3.5.1 Apache OpenNLP (OpenNLP, 2011) 3-17](#_Toc395181709)

[3.5.2 ANNIE (Cunningham et al., 2002) 3-18](#_Toc395181710)

[3.5.3 Twitter NLP Tools (Ritter et al., 2011) 3-19](#_Toc395181711)

[3.5.4 Weka (Weka 3, n.d.) 3-21](#_Toc395181712)

[3.5.5 TwitIE (Bontcheva et al., 2013) 3-21](#_Toc395181713)

[4.0 The FILIET System 4-1](#_Toc395181714)

[4.1 System Overview 4-1](#_Toc395181715)

[4.2 System Objectives 4-1](#_Toc395181716)

[4.2.1 General Objective 4-1](#_Toc395181717)

[4.2.2 Specific Objectives 4-1](#_Toc395181718)

[4.3 System Scope and Limitations 4-1](#_Toc395181719)

[4.4 Architectural Design 4-3](#_Toc395181720)

[4.4.2 Data Sources 4-4](#_Toc395181721)

[4.4.3 Crawler Module 4-5](#_Toc395181722)

[4.4.4 Preprocessing Module 4-6](#_Toc395181723)

[4.4.5 Feature Extraction Module 4-8](#_Toc395181724)

[4.4.6 Disaster Classifier 4-10](#_Toc395181725)

[4.4.7 Information Extraction Module 4-11](#_Toc395181726)

[4.4.8 Template Filler Module 4-12](#_Toc395181727)

[4.5 Physical Environment and Resources 4-13](#_Toc395181728)

[4.5.1 Minimum Software Requirements 4-13](#_Toc395181729)

[4.5.2 Minimum Hardware Requirements 4-13](#_Toc395181730)

[5.0 References 5-1](#_Toc395181731)

[6.0 Appendix 6-1](#_Toc395181732)

[6.1 Appendix A 6-1](#_Toc395181733)

[6.2 Appendix B 6-2](#_Toc395181734)

**List of Tables**

[Table 1‑1. Timetable of Activities (April 2014 - April 2015) 1-7](#_Toc395094917)

[Table 2‑1. Summary of Reviewed Information Extraction Systems 2-6](#_Toc395094918)

[Table 3‑1. Examples of official government institution Twitter accounts and unified hashtags 3-14](#_Toc395094919)

[Table 3‑2. Examples of disaster-related tweets with extractable information 3-15](#_Toc395094920)

[Table 3‑3. Confusion Matrix (Davis and Goadrich, 2006) 3-16](#_Toc395094921)

[Table 4‑1. Sample Entries of Tweets in CSV File 4-4](#_Toc395094922)

[Table 4‑2. Sample Gazetter for Storm Names (Philippines) 4-4](#_Toc395094923)

[Table 4‑3. Sample Rules stored in the database 4-5](#_Toc395094924)

[Table 4‑4. Template for General Information 4-5](#_Toc395094925)

[Table 4‑5. Template for Earthquakes 4-5](#_Toc395094926)

[Table 4‑6. Template for Storms 4-5](#_Toc395094927)

[Table 4‑7. Template for Floods 4-5](#_Toc395094928)

[Table 4‑8. Sample Input/Output for Text Normalizer 4-6](#_Toc395094929)

[Table 4‑9. Sample Input/Output Tokenizer 4-7](#_Toc395094930)

[Table 4‑10. Sample Input/Output POS Tagger 4-7](#_Toc395094931)

[Table 4‑11. Sample Input/Output for Filipino NER 4-8](#_Toc395094932)

[Table 4‑12. Sample Input/Output Disaster Tagger 4-8](#_Toc395094933)

[Table 4‑13. Sample Features and Values 4-10](#_Toc395094934)

[Table 4‑14. Sample Input/Output for Disaster Classifier 4-11](#_Toc395094935)

[Table 4‑15. Sample Generated Rules 4-12](#_Toc395094936)

[Table 4‑16. Sample Template for Earthquake 4-12](#_Toc395094937)

[Table 6‑1. Results of the study conducted by University McCann 6-1](#_Toc395094938)

[Table 6‑2. Example of Filipino Morphemes 6-2](#_Toc395094939)

**List of Figures**

[Figure 1‑1. Research Methodology Phases 1-5](#_Toc395181735)

[Figure 3‑1. Structure of an Information Extraction System 3-2](#_Toc395181736)

[Figure 3‑2. StaLe Lemmatization Process 3-5](#_Toc395181737)

[Figure 3‑3. Poibeau's General Architecture 3-7](file:///V:\Users\Vin\Dropbox\Awesome%20Nerds\THESIS\Gridlock\THESIS\thesis-document%20(8-06-2014%201321).docx#_Toc395181738)

[Figure 3‑4. Architecture of IE2 Adaptive Information Extraction System 3-8](#_Toc395181739)

[Figure 3‑5. Architecture of LearningPinocchio 3-9](#_Toc395181740)

[Figure 3‑6. Rule Induction Step 3-10](#_Toc395181741)

[Figure 3‑7. Algorithm for Choosing the Best Rules 3-11](#_Toc395181742)

[Figure 3‑8. Information Extraction Process of LearningPinocchio 3-11](#_Toc395181743)

[Figure 3‑9. SOMIDIA's Architecture 3-13](#_Toc395181744)

[Figure 4‑1. System Architecture of FILIET 4-3](#_Toc395181745)

# Research Description

This chapter introduces the research which will be undertaken in the field of Text Classification and Information Extraction (IE) in Natural Language Processing (NLP) for disaster management. This chapter is divided into four sections. The first section will talk about the motivations and the problem that needs to be addressed. The second section will discuss the objectives of the research. The third section will discuss the scope and limitations of the study. Lastly, the fourth section will tackle the significance of the research with regards to the Philippine society.

## Overview of the Current State of Technology

According to a report of the United Nations International Strategy for Disaster Reduction (UNISDR) Scientific and Technical Advisory Group, disasters have destroyed lives as well as livelihood across the world. Just between 2000 and 2012, about 2 million people died and an estimate of US$ 1.7 trillion of damage were sustained in disasters. In the same report, the UNISDR posits the use and research of new scientific and technological advancements in disaster management (Southgate et al., 2013).

Social media are online applications, platforms, and media which aim to facilitate interaction, collaboration and the sharing of content. In a study of Universal McCann and an analysis of 24/7 Wall St., LLC about social media, the Philippines got a high rank in most of the categories. This led to the country being dubbed as the “Social Media Capital of the World” (Universal McCann, 2008; Stockdale & McIntyre, 2011). Social media plays a vital role in disaster management. After the Haiti earthquake in 2010, numerous posts and photos were published in various social media sites. 48 hours later, the Red Cross has received a donation of US$8 million. Social media has enabled the generation of community crisis maps and interagency maps, a map that works as an intermediary between the public and relief organizations (Gao, Barbier & Goolsby, 2011). Patrick Meier, a crisis mapper, makes use of social media to improve the efficiency of relief efforts. He launched the website MicroMappers[[1]](#footnote-2), that quickly sort through online data, from tweets to uploaded photos, and then display the information on satellite maps, to assist in relief efforts during the disaster of Super Typhoon Haiyan (also called Yolanda) in the Philippines (Howard, 2013).

Social Media Monitoring for Disasters (SOMIDIA) (Cheng et al., 2011) is “a crisis mapping system that focuses on plotting authentic crisis events on an interactive map in near real time.” It uses data from news sites, blogs, web forms, SMS, and social networking sites (i.e. Facebook and Twitter) to know where the disasters are in near real-time. SOMIDIA extracts information like the type of disaster, date and time of the disaster, and the location of the disaster, so that the system can display the disasters in a map in near real-time. However, the data comes from different sources. Extracting information from different sources could be difficult for the computer as these data are unstructured. SOMIDIA needs an information extraction module to be able to collect the needed information.

Information extraction is defined as “extracting structured data from unstructured data as provided, for example, text document” (Feilmayr, 2011). Information extraction systems have been used in different fields such as clinical narratives – MedEx (Xu, & et. al., 2010), healthcare – VnHIES (Dung & Kamayama, 2007), and legal documents - Legal TRUTHS (Cheng et al., 2009). On the current implementation of SOMIDIA, it already has an information extraction system for English and Filipino texts. The problem is that SOMIDIA’s information extraction module for Filipino has difficulty in extracting the disaster and location from a given text. When compared to the information extracted from English text, the English dataset performed significantly better than the Filipino dataset (Cheng et al., 2011).

One of the main problem with information extraction when it comes to the Filipino or “Taglish” language is the characteristic of the two aforementioned languages. By nature, the Filipino language, in general, is a morphologically rich language. Morphologically rich languages (MRL) are languages that have words which are composed of the combination of the root word and a number of morpheme components epentheses (the addition of suffixes to add new meanings to the word), metatheses (the addition of suffixes that may change the spelling of the entire word), replacement (the addition of suffixes that may change the sound of certain parts of the root word), infixation (the addition of infixes splits the root word), and reduplication (repeating a combination of letters from the root word to give it a new meaning) (Stone, 2004). With these, systems that are being built for analyzing/translating the Filipino language are having problems when it comes to identifying the most concise translation of the Filipino word into other languages. But there have been researches that made use of Dependency Parsing and Part-Of-Speech (POS) Tagging to analyse a given Tagalog statement. In the Dependency Parsing approach, a dependency structure will be constructed based on the input statement and then, identify the syntactic head of the given statement from the constructed tree and lastly, analyze it by linking the head and its dependents. (Manguilimotan & Matsumoto, 2011). In the POS Tagging approach, the input statement is chopped into words and then these words are tagged or labeled according to their function in the statement and then given syntactical analysis (Manguilimotan & Matsumoto, 2009).

The existing information extraction system present in SOMIDIA has limitations when it comes to handling text because it can only handle English text. The system still has problems when handling Filipino text for the reason that the information extraction done on text written in Filipino is using keyword-based searching. Currently, there are no stable tools that can handle Filipino text (Cheng et. al., 2011). Social media data in the Philippines shows inconsistency because when Filipinos communicate or post in social networking sites they use different languages such as English, Filipino, Taglish and “TXTSPK”, especially today, when people with different language and culture communicate on the same medium (social networking sites) (Ghedin, 2011). Another problem with social media data is the existence of micro-posts, or short posts, such as tweets which do not have much contextual information and tend to be less grammatical. (Maynard et. al., 2012).

## Research Objectives

This section presents the general and specific objectives of the proposed research.

### General Objective

To develop an information extraction system that extracts relevant information from disaster-related texts from social media and takes into consideration the different available variations in the Filipino language.

### Specific Objectives

The following are the specific objectives of the research.

1. To review different information extraction systems;
2. To identify data source that will be used for the information extraction system;
3. To review different natural language processing techniques that will pre-process data for the information extraction system;
4. To review different information extraction techniques;
5. To evaluate existing tools and resources which could be incorporated in the information extraction components of the system;
6. To determine the metrics for evaluating the information extraction system;

## Scope and Limitations of the Research

The research aims to design an information extraction system for the Filipino language. It will also cover the review of various information extraction systems in order to know the different approaches on implementing them. Different existing domain-independent, domain-dependent information extraction systems will be reviewed in order to understand the architectures, implementation and components of an information extraction system. It will also review information extraction for MRL in order to understand the techniques used to extract from MRL since the Filipino language is considered to be an MRL.

The research will identify the source where the data will be collected. Example of data source would be Facebook and Twitter. Being the social media capital of the world, Filipinos post status of their lives in popular social networking sites like Facebook and microblogging sites like Twitter. There are also news from online newspaper. Identifying the data source that will be used in the information extraction will help in choosing appropriate preprocessing techniques and algorithms.

In order for the information extraction system to perform better, the research will review different natural language processing techniques that will preprocess the data before feeding it to the information extraction system. Examples of the NLP techniques that will be reviewed are text classification and text normalization. Text classification is the process of automatically assigning a text or document into a predefined category based on their content (Özsu & Liu, 2009). Texts may need to be classified according to categories so that the system can use appropriate algorithm to extract the information. Text normalization is the transforming of ill-formed words into their canonical forms (Han & Baldwin, 2011). The information extraction system will need a text normalizer as data coming from social media sites (Twitter and Facebook) are noisy. Most of the text has no structure, incorrectly spelled words, and invented terms.

The research will review different information extraction techniques that will be used for the information extraction systems. Some of the techniques that will be reviewed are Named Entity Recognition (NER), lexical analysis, and conference analysis. Lexical analysis involves splitting up sentences into words and performing Part-Of-Speech tagging to each word (Grishman, 1997). NER is the classification of each word into a category (Zhou & Su, 2002). Conference analysis is the resolving of references for the pronouns (Grishman, 1997).

Existing tools that will be used in building the information extraction system will be reviewed and evaluated. Example of NLP tools are OpenNLP and Lingpipe. OpenNLP is a machine learning based toolkit for the processing of natural language text that can support a number of common NLP tasks like tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and co-reference resolution (Apache Software Foundation, 2010). On the other hand, Lingpipe is a toolkit for processing text using computational linguistics that can perform certain tasks like finding names of people/organizations/event, classify Twitter data, and check spellings (Alias-I, 2011).

In order to evaluate the information extraction system, the research will determine the metrics to measure the system’s performance.

## Significance of the Research

Being the social media capital of the world, the Philippines generates a lot of diversified information that cannot be easily tapped because of the limited capabilities and tools that are available in processing the language unto which these information are written in, the Filipino language. With an information extraction system that is specifically built for the Filipino language, people can explore more possibilities and opportunities with regards to effectively utilizing this information from the web.

In the disaster management standpoint, respective stakeholders can collect disaster-related information in a way that is less strict because with an information extraction system specifically built for the Filipino language, these stakeholders can effortlessly accept and process information that are written in a much more natural and open way. With this, they can reach out to more people and to more places because they can have a system that understands how Filipinos speak and communicate through the different social media platforms available.

In the case of SOMIDIA, improving the on-board information extraction system can further enhance the usability and accuracy of the information presented in the existing SOMIDIA system. With an enhanced and improved information extraction system, SOMIDIA can accept and process information that are written in a more open and unstructured way, or simply, information that are written in the format of the different variations in the Filipino language like the ‘TXTSPK’, ‘Jejemon’ and the ‘Bekimon’. With an improved information extraction algorithm, the new information extraction system will be able to increase the probability of accurately and precisely understanding the normal Filipino language and at the same time, it will include support for the different variations like the ‘TXTSPK’, ‘Jejemon’ or ‘Bekimon’ since most of the information that will be used by the SOMIDIA system are taken from the different social media platforms and they are written in a very open and informal way.

## Research Methodology

This section discusses the different activities that will be performed throughout the research. Scrum-based methodology, an iterative software development life cycle, will be applied in the course of this research in order to ensure that the research will be able to adapt to changes in requirements.

Figure 1-1 shows a diagram of the phases the research will undergo. The phases are as follows: investigation and research analysis, system design, system development, system integration and testing, system evaluation, and documentation. Regular consultation with the thesis adviser will also be conducted in order to keep the research on track for the whole duration of the thesis.

Figure 1‑1. Research Methodology Phases



### Investigation and Research Analysis

This phase involves the study and understanding of the fundamental knowledge of the concepts, algorithms, techniques, and tools which can be used to implement the system as well as identifying the modules and requirements of the system to be developed. The main key activity involved in this phase is various literature reviews of related works. From those related works, the pre-processing techniques, information extraction techniques, tools, and evaluation metrics used are identified. The listed techniques, tools, and metrics are then compared and evaluated to see which ones can be adopted to the system.

### System Design

In this phase, the system will be designed according to the information gathered during the course of the Investigation and Research Analysis phase. It is in this phase where appropriate architectures, algorithms, information extraction techniques, and other necessary tools shall be identified so that they can be effectively utilized in the making of the system. Also, it is in this phase where necessary modules for the system will be identified based on the different processes and features that will be built into the system. This phase will cover the designs of the User Interfaces and the basic architecture for the databases that will store the data that will be gathered and used by the system. Finally, this phase will also cover the identification of the source of the data that will be used and processed by the system. And once the data sources have been identified, data collection will immediately commence.

### Sprints

A two-week timeframe for each sprint will be used. This is to ensure that there is progress in the research. Each member is expected to produce a working output based on the tasks assigned to him during the sprint planning meetings. The tasks may vary from developing a part of the system or to conduct further study regarding a certain concept.

### Sprint Planning Meetings

At the beginning of each sprint, a sprint planning meeting is conducted. Tasks that must be accomplished for the current sprint will be discussed here. Included in these meeting is the assignment and division of the tasks among the members of the team. Also, the evaluation of the tasks in the previous sprint are done here. If there are any unmet tasks, these will be carried over to the next sprint.

### Scrum Meetings

Scrum meetings of 10-15 minutes in duration will be conducted daily. The purpose of this is to update each member what has or has not been accomplished yet in the assigned task. This ensures that there is daily progress and if there are issues that hinders a member from accomplishing his assigned task.

### System Development

In this phase, actual development of the system will be done. It will follow the design made during the System Design phase. Data collection will also be done in this phase. Each member of the team will be assigned to modules. The development of the system will follow a scrum-based methodology wherein the system is developed in an iterative manner. Daily and weekly meetings, as well as regular consultations with the adviser, are conducted in order to assess the progress of the thesis and to plan the succeeding tasks.

### System Integration and Testing

In this phase, all the modules that have been developed during the System Development phase will be integrated into one system. This phase will cover unit testing processes for each module to ensure that there will be no significant bugs that can be found after integration processes are completed. After finishing the integration process, the system will be subjected to another round of tests to check for any faulty integration and bugs that may have arose during the integration process.

### System Evaluation

In this phase, the system's performance will be evaluated based on the metrics that were chosen. As of the moment, the metrics that will be used in this phase will be the Precision, Recall and F-measure results of the information extracted by the system. The information that were extracted by the system will be subjected to a number of tests that will test its Precision, Recall and F-measure when compared to the information that were extracted manually and to those that are extracted from the training set. Although, the set of metrics that will be used might change during the course of the research as these metrics will be modified to fit the needs in accurately measuring the performance of the system to be developed.

### Documentation

Every activity or methodology that is performed will be fully documented so that they can be monitored when it comes to the modifications and progress that are made in accomplishing the documents and the system proposed in this research. Also, the documentation will be used for further references, in case there is a need to validate or cross-reference any future work that is in mind.

### Calendar of Activities

Table 1-1 shows a Gantt chart of the activities for the thesis period. Each bullet represents one week worth of activities

Table 1‑1. Timetable of Activities (April 2014 - April 2015)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Activities** | **Apr**  **(2014)** | **May** | **Jun** | **Jul** | **Aug** | **Sept** | **Oct** | **Nov** | **Dec** | **Jan** | **Feb** | **Mar** | **Apr**  **(2015)** |
| Investigation and Research | \_ \* \_ \* | \_ \_ \* \* | \_ \* \* \* | \* \* \* \* | \* \* \* \* |  |  |  |  |  |  |  |  |
| System Design |  |  |  |  | \_ \_ \* \* | \_ \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \_ \_ | \_ \* \* \* | \* \* \_ \_ |  |  |
| System Development |  |  |  |  | \_ \_ \* \* | \_ \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \_ \_ | \_ \* \* \* | \* \* \_ \_ |  |  |
| System Integration and Testing |  |  |  |  | \_ \_ \* \* | \_ \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \_ \_ | \_ \* \* \* | \* \* \_ \_ |  |  |
| System Evaluation |  |  |  |  | \_ \_ \* \* | \_ \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \_ \_ | \_ \* \* \* | \* \* \_ \_ | \* \* \* \* | \* \_ \_ \_ |
| Documentation | \_ \* \_ \* | \_ \_ \* \* | \_ \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \* \* | \* \* \_ \_ | \_ \* \* \* | \* \* \* \* | \* \* \* \* | \* \_ \_ \_ |

# Review of Related Works

This chapter discusses the features capabilities, and limitations of existing research, algorithms, or software that are related or similar to the thesis.

## Machine Learning-Based Information Extraction Systems

This part discusses information extraction systems that use machine learning-based techniques.

Machine Learning for Information Extraction in Informal Domains (Freitag, 2000)

The researchers of the paper explored one of variation of the slot-filling problem and that is to find the best unbroken fragment of text to fill a given slot in the answer template. There is a definite template that is given to an IE task. The template consists of fields that need to be filled with instances from the text source. The researchers set two ways of simplifying how to study the behaviour of the algorithms to be developed: to isolate each field learning problem and focus on fields that is not instantiated or have a unique instance in a text source. With this, they found two primary aspects: multi-strategy learning and feature engineering. Multi-strategy learning because they believed that there is no single representation for all IE problems. Feature engineering because ML of a feature set is needed to help adapt to domains containing novel structures since they will target informal domains. The researchers used four ML components: rote learning, term-space learning, learning abstract structure with grammatical inference, and relational learning for information extraction. They did experiments to gauge the performance of the four learners.

To conclude, the researchers found out that it is possible to perform IE from informal domains found in the internet. Also, they stated that ML is a rich source of ideas for different algorithms that can be trained to perform IE. They have shown that with the right ML techniques it is possible to train effective extractors with very simple document representations.

TOPO - Information Extraction System for Natural Disaster Reports from Spanish Newspaper Article (Téllez-Valero, 2005)

This information extraction system extracts information related to natural disasters from newspaper articles written in Spanish. The system extracts the following information: (1) information related to the disaster itself (date, place and magnitude), (2) information related to buildings (number of destroyed buildings, affected houses), (3) information related to people (number of dead, missing or wounded), (4) information related to infrastructure (number of affected hectares, economic lost). It is able to extract information on natural disaster like hurricanes, forest fires, inundations, droughts and earthquake.

The system uses general information extraction system architecture. First, the document is turned into Boolean vectors representing the presence and absence of certain words. This is the document feature extraction stage. In order to limit the dimension, they used information gain technique. After it is turned into a Boolean vector, it will now be classified. They used Support Vector Machine (SVM), Naïve Bayes (NB), C4.5, k-Nearest Neighbours (kNN). After it has been classified, it needed to select text that might contain relevant information. This is the candidate text selection stage. They used grammar to select the text and a dictionary of names and number to treat grammar exceptions. Then the output will be candidates of relevant information. Then, the system will now select which of the information will be used. This uses the same algorithms in the text classification stage. They used different classifier for different output.

This architecture boasts its portability because it is language independent and domain adaptive. It is language independent because its training features and candidate text segments are based on simple lexical rules. It is domain adaptive because it only needed to change the training corpus.

The text filtering stage was evaluated on 134 news reports on the metrics of precision, recall and F-measure. The algorithm that produced the best result was the SVM. They got an F-measure from 72% to 88% on classification of news reports. The information extraction stage was evaluated on 1353 text segments that consist of names, dates and quantities randomly taken from 365 news reports. The best classifier for name and quantities was SVM, while kNN for dates. The overall system got an average of 72% on F-Measure.

## Rule-Based Information Extraction Systems

This part discusses information extraction systems that use rule-based techniques.

Vietnamese Real Estate (VRE) Information Extraction (Pham & Pham, 2012)

The Vietnamese Real Estate (VRE) Information Extraction system extracts information from Vietnamese Real Estate Advertisements. It collects information like the type of estate, category of the estate, area, zone, price, name of the author, and contact details. The system uses the GATE framework for its architecture.

For its data, it has to pass certain criteria before it is fed into the system. First, it must be news articles related to real estate advertisement. Second, only one advertisement from each input data file. Lastly, it must be strip off of all its HTML tags. After the data has met all the criteria, it will now go to data normalization first. The data normalization helps reduce ambiguity and helps the human in annotation. First, it must add the necessary punctuation at the end of the sentence. Second, it merges multiple paragraphs into one. Third, normalize the punctuations, remove redundant spaces and capitalizes the first character after each punctuation. Then lastly, normalize the telephone, price, area and zone to a common pattern. After the data is normalized, it will now be manually annotated using Callisto, an annotation software.

After it has been annotated, it is now ready to go to the information extraction system. It will go first through the tokenizer. The tokenizer will output two types of annotations, Word and Split. The Word annotations contains the part-of-speech, the word, checks if the first letter is capitalized, and other features (kind and nation). This will be used to create the Java Annotation Pattern Engine (JAPE) rules. The Split annotation contains the delimiter. After it goes through the tokenizer, it will now go through the Gazetteer. The gazetteers are dictionaries that are created during the system development. It contains dictionaries for potential named entities (person, location) or categories, phrases uses in contextual rules (name prefix or verbs that are likely to follow a person’s name), and potential ambiguous entities. The output of the gazetteer is a lookup annotation covering the specific semantics. After the gazetteer, it will now be passed to the JAPE transducer. The JAPE transducer is responsible for extracting the information. It uses JAPE rules to recognize the entities that will be needed to extract. The output is the annotated documents.

The system has been tested in a lenient and strict criterion. An entity that is recognized correctly when the type is correct but the span overlap in the annotated corpus is called the lenient criteria. On the other hand, an entity that is recognized correctly when the type and span are the same in the annotated corpus is called strict criteria. On the lenient criteria on test data, it measured 96% in F-measure. While on the strict criteria, it measured 91% in F-measure. The problem is on the data. The writing styles of the people are very diverse. The system has problem in recognizing some of the entities like the zone entity because some of the zone entity are very long and does not use capitalization.

Business Specific Online Information Extraction from German Websites (Lee & Geierhos, 2009)

The Business Specific Online Information Extraction System is a system that extracts information from the information pages of a German business website like its company profile, contact page, imprint and then identifies relevant business specific information. The system concentrates on the extraction of specific business information like company names, addresses, contact details, names of CEOs, etc. With regards to the way how the researchers pre-process their chosen input data, they interpret the HTML structure of documents and analyse some contextual facts to transform the unstructured web pages into structured forms. The approach applied by the researchers is quite robust in variability of the DOM (for the web pages), upgradeable and keeps data up-to-date. The evaluation metrics showed high efficiency of information access to the generated data. In their conclusion, they stated that the developed technique is also adaptive to non-German websites with slight language-specific modifications, and experimental results from real-life websites confirm the feasibility of their approach.

In their proposed system, the researchers had two main modules for processing and extracting information from the German Information Web Pages: one for establishing a relational database storing company information and the other is for providing a query module. Within these two modules are three sub process that are done to further process the input data: (A) Localization of the Information Pages on the Web; (B) Document Analysis and Information Extraction; lastly, (C) Query Processing. In sub process A (Localization of the Information Page), a web crawler is fed with the URL’s of the web pages that are stored in the specialized database and then it fetches them from the web. Afterwards, the proposed system will then retrieve the document by following the anchor tags that lead to the information pages. On the other hand, in sub process B (Document Analysis and Information Extraction), the fetched Information Pages are sent to an ‘info analyser’ module which examines the HTML content of the page and then extracts the needed information bits. Here, the system exploits the internal structure of the named entities and uses sublanguage-specific contexts or attribute classes to identify the attribute-value pairs. Lastly, in sub process C, the user of the system is given the right to query the database for the information bits that he/she needs and then add these bits to the index.

For the Information Page Analyser (info analyser) in sub process B, the input data has to further go through a number of processes to finally extract the information needed by the user. When given an Information Page, the analyser starts by pre-processing the frame structure and existing JavaScript of the page. And before creating the expressive DOM Tree, the HTML file of the page has to be validated and corrected, if needed, by using a special tool called ‘tidy’. After doing so, the system will now be able to locate the minimal data region (or the data region of the information bit searched for) surrounded by a number of HTML tags which contain the information record being searched for. By doing a depth- first traversal of the expressive DOM tree, the desired sub tree can be isolated based on the headings of the data record like the following: “Herausgeber” (publisher), “Betreiber” (operator), “Anbieter” (provider) and etc. The system was programmed to disregard domain name irrelevant information; thus, the analyser will work further on with a pruned DOM tree. After identifying the minimal data region, all information bits that are relevant to the domain name are extracted by using the Named-Entity Recognition technique and the attribute-value process (each attribute has a corresponding value that is indicated by the structure of the HTML file it is in) with respect to its external contexts and internal features. The system’s analyser module considers about 20 attribute classes and searches their corresponding values on the information page of business websites. The following are some of the attribute classes that are considered by the analyser: company name, address, phone and fax number, e-mail, CEO, management board, domain owner, contact person, register court, financial office, register number, value added tax number (VAT ID), and etc. After extracting the information bits needed from the pruned DOM trees, the information bits are then normalized to make sure that all information are consistent. The following are the classes that are affected by the normalization process: company names, legal form, register number, address (street, zip code, city), contact (phone and fax number, email), person name, and legal notification (tax number, VAT ID).

To conclude, the system performed surprisingly accurate with an average precision score of 99.1% and a recall score of 91.3% from a small test corpus that is composed of approximately 150 business web pages. The only encountered problem by the system is when value for certain attributes is erroneously represented like text in phone numbers and etc.

Ontology-Based Information Extraction (OBIE) System for French Newspaper Articles (Nebhi, 2012)

Since most of the information extraction systems are based on the English language, it poses a problem on other languages for there are not much tools available. In order to address this problem, the system maps the extracted entities to the ontology.

This system extracts person, location and organization on a French newspaper article. It collects data from LeMonde.fr. The system uses the GATE framework for annotation of entities in text and maps them to the ontology. It uses DBpedia databank that is based on Wikipedia projects. It contains 3,220,000 instances and is organized into a hierarchy of 320 classes and 1650 different properties. The system consists of 4 parts: pre-processing, gazetteer, rule-based semantic annotation, and the output. First, the system will pre-process the text. It will perform tokenizer, sentence splitter and POS tagger using the GATE application. After it is pre-processed, it will now go to the gazetteer. It will perform a lookup for the named entity recognition. After it passes through the gazetteer, grammar rules will be applied to create semantic annotation. The rules are written in JAPE which is part of the GATE framework. The system contains approximately 100 rules.

The system is evaluated using the Balance Distance Metrics (BDM) to take account the ontological similarity. It is also evaluated with the gold standards. They manually annotated the documents using concepts on DBpedia ontology, and then compared it with the gold standard. They only evaluated person, organization and location named entities. The system scored an average of 0.94 on the BDM and achieved a 91% F-Measure

## Other Information Extraction Systems

This part discusses information extraction systems that use other techniques.

SOMIDIA - Social Monitoring for Disaster Management (Cheng et al. 2011)

SOMIDIA is a crisis mapping system that focuses on plotting disaster on an interactive map in near real time. SOMIDIA collects data from different sources like news feeds, posts, SMS, blogs and microblogs. One of the main component of SOMIDIA is its information extraction module. It extracts from both Filipino and English texts.

For the information extraction module, first, documents go through a tokenizer. They used OpenNLP to tokenize the document, then it will go to a sentence splitter. The sentence splitter accepts the list of tokens and annotation list. It has a list of abbreviations so that the system can distinguish abbreviation periods from a period. The goal of the sentence splitter is to separate sentences by adding appropriate ending markers (period). The system used OpenNLP’s sentence splitter for its sentence detection. After the document has been split into sentences, it will go through a language guesser. They needed to differentiate English text from Filipino text because the language has different extraction techniques. They used frequency distribution of the words to detect the language. The output of the language guesser is the document with added metadata of the language. If the text is in English, the language guesser will passed the document to the POS tagger. Else, it would be passed to a Filipino NER.

For the English information extraction module, first it will go through the POS tagger. It uses the OpenNLP’s POS tagger function. The output is a list of token with its corresponding POS tags. After the POS tagger, it will go through a chunker. The chunker groups the tokens to their corresponding part-of-speech tag. This will be used to determine noun and verb phrases. It uses OpenNLP’s noun and verb chunker. After chunking, it will passed through the English NER. The NER only focuses on proper nouns. It uses LingPipe because of flexibility. LingPipe’s NER uses three types of approaches, dictionary-based, rule-based and statistic based approaches. After the NER, it will go through coreference resolution. The coreference resolution will find the noun counterpart of the pronouns. It uses Russian Mitkov algorithm for the resolution and WordNet for the lexicon. The normalization (standardizing data, collapsing of same sentences) will be done in this phase. The last step is now the information extraction phase. It uses JAPE rules to extract the information. The rules are paired with the two-tiered bootstrapping algorithm. The first tier bootstrapping algorithm starts with a small seed of words or rules. Then from the seed, it will try to learn the extraction pattern. The learned pattern will be used to generate new extraction pattern. The process will then repeat. The second-tiered bootstrap is responsible for keeping the most relevant extraction pattern.

For the Filipino extraction module, the document will go through the Filipino NER. They created their own NER because there are no existing Filipino NER tool. It uses a dictionary-based and rule-based approaches for their NER. After it has been tagged, it will now go through the Filipino extractor, the Filipino extractor has pre-defined rules (e.g. <event> sa <location>) that will extract the needed information.

The system is evaluated using precision, recall and F-measure. They evaluated it on Tweets and news feeds. For English tweets, it scored a 75.17% F-measure on extracting disaster and 62.83% on extracting location. For Filipino Tweets, it scored 82.13% F-measure on disaster and 56.32% on extracting location. For news feeds, it scored 45.40% F-measure on English news feeds, while 38.82% on Filipino news feeds. The tweets scored higher because it is much easier to extract patterns on shorter text. The needed information will most likely be located near the text. On longer texts, the information needed might be located far away.

Table 2-1 shows a summary of all the reviewed information extraction system. The table indicates the system name, the language and type of data it can extract, the domain, NLP pre-processing techniques, information extraction techniques, and evaluation metrics used by the system.

Table 2‑1. Summary of Reviewed Information Extraction Systems

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **System** | **Language** | **Type of Data** | **Domain** | **Pre-processing Techniques** | **Information Extraction Techniques** | **Evaluation Metrics** |
| Machine Learning for Information Extraction in Informal Domains  (Freitag, 2000) | N/A | Documents (i.e. email) | Informal Domain | Not mentioned | Machine Learning-  Based | Precision, Recall |
| TOPO - Information Extraction System for Natural Disaster Reports From Spanish Newspaper Article (Téllez-Valero, 2005) | Spanish | Free-text | Natural Disasters | Text Classification, Document Feature Extraction | Machine Learning- Based | Precision,  Recall,  F-measure |
| VRE Information Extraction System (Pham & Pham, 2012) | Vietnamese | Free text | Real Estate Advertisement | Text Normalization | Rule-Based | Precision,  Recall,  F-measure |
| Business Specific Online Information Extraction from German Websites  (Lee & Geierhos, 2009) | German | Structured Text | Business Specific Information | Named Entity Recognition, Text Normalization, Attribute-Value Process | Rule-Based | Precision, Recall |
| Ontology-Based Information Extraction (OBIE) System  (Nebhi, 2012) | French | Free text | News article | Tokenization, POS Tagging, Sentence Splitter | Rule-Based, Ontology | Precision, Recall,  F-measure,  BDM |
| Social Monitoring for Disaster Management  (Cheng et al. 2011) | English, Filipino | Free text | News article, tweets | Tokenization, Sentence Splitter, Language Guesser | Machine-Learning Based | Precision, Recall, F-measure |

# Theoretical Framework

This chapter presents a discussion on the different theoretical concepts associated to information extraction systems, and as well as common architectures, approaches, modules, and resources needed in developing such systems.

## Information Extraction

There is already huge amount information that is freely available in the internet. The problem is that people could not process them these information because of the huge volume. It becomes more difficult as the information is written in natural language, which can be ambiguous. However using an information extraction system, it can now automatically collect information from different sources like news, papers, and journals. Information extraction is the identification of class of events or relationship and the extraction of relevant arguments of the event or relationship inside a natural language. It involves the creation of a structured representation of the facts that will be extracted. An information extraction system can only extract those facts that are represented (Grisham, 1997).

Basically, an information extraction system is divided into two parts, local text analysis and discourse analysis. The local text analysis is responsible for extracting the information from a text document. It consists of lexical analysis, name recognition, partial syntactic analysis and scenario pattern analysis. The lexical analysis is responsible for splitting up the text into tokens. After splitting the text, it looks up a dictionary to fill up the part of speech and features of each token. After the lexical analysis, it goes through name recognition. Name recognition is responsible for identifying proper nouns, aliases, and other special forms (dates and currency). It uses regular expressions that are stated in the POS, syntactic features and orthogonal features to identify names. It also uses a dictionary that contains the list of proper nouns like company to identify the names. After going through name recognition, it passes through a partial syntactic analysis to identify some of the syntax of the text. It is responsible for identifying some of the like noun groups and verb groups. However, some system does not implement a syntactic analysis. After syntactic analysis, it goes through scenario pattern matching. Scenario pattern matching is the extraction of related events or relationship relevant to the scenario. The output of the scenario pattern matching is two clauses. The first clause is a reference to an event structure while the second clause is a reference to a created entity (Grisham, 1997).

After going through the phases of local text analysis, it can now pass through the discourse analysis. The discourse analysis is the one who will combine all the information extracted during the local text analysis and who will format the information. Under the discourse analysis are coreference analysis and inference. Coreference analysis tries to resolve anaphoric references (pronouns and definite noun phrases). To determine which entity is referenced, the most recent previous mention of the entity is the anaphoric reference. After the coreference analysis, it will now go to inference and event merging. Inference is responsible for making implicit information explicit. It uses system production rules to implement the inference module. After the inference, it can now be place in the data representation. Figure 3-1 shows the general flow of an information extraction system (Grisham, 1997).

Figure 3‑1. Structure of an Information Extraction System



### Information Extraction Modules

This section explains the different modules that are commonly used in information extraction systems.

#### Text Classification

Text classification is the automatic classification of text into different categories based on their content. It consists of several important components: document representation, dimensionality reduction, classification algorithm, and performance evaluations (Shen, 2010). This will be useful as different type of text may need different type of extraction techniques.

##### Document Representation

Classification algorithms cannot understand the text directly. The text must be converted into some form that can be easily understood by the algorithm. There are different methods that could be used to represent documents. The traditional representation of documents is the Bag-of-Words (BOW) representation, which is based on Vector Space Model. The use of Bag-Of-Words may vary as it can have different representation. (Shafiei et al., 2007)

One is the word representation. In word representation, each word in the document is considers as a feature. The problem with word representation is the curse of dimensionality because text documents has a lot of unique words. (Shafiei et al., 2007)

Another representation is the term representation. Here, it uses multi-words or phrases as its feature. This drastically reduce the number of features. However, there has been mixed results on the experiment results. (Shafiei et al., 2007)

Character N-gram is another feature representation that could be used. Character N-gram takes n characters as the feature. Instead of focusing on the word, the character n-gram uses the characters. This makes model language independent. It is less susceptible to typographical errors and grammatical errors. It also does not require any linguistic preprocessing. (Shafiei et al., 2007)

##### Dimensionality Reduction (Feature Selection)

The problem with text classification is the huge number of features present in the vector space. This huge number of features could drastically reduce the performance of the algorithm. It is important that reduce the number of features without sacrificing accuracy. The reduction of feature is called feature selection. There are different methods that could be used in feature selection.

Document Thresholding (DF) counts all the occurrences of each word in the document, then all the words that did not reach the specified threshold will be removed. The rationale behind this is that those words that has few occurrences are irrelevant (Wei et al., 2010).

Information Gain (IG) measures the bits of information that could be gained in the document. The information gain of a word (w) is defined as:

Where is the set of all possible categories, is the probability of a document classified to the category. This will be computed for all the words in the documents. Then, remove the words that did not reached the specified threshold. (Wei et al., 2010)

Mutual Information (MI) is the modelling of the word to a category. The mutual information criterion between term *t* and category *c* is defined as:

And is estimated using:

Where,

A = number of times *t* and *c* co-occurs

B = number of times *t* occurs without *c*

C = number of time *c* occurs without *t*

N = number of documents

##### Classification

There are different classification algorithms that could be used in classifying text. One of which is the Bag-of-Word technique. In Sriram et al. (2010) work, they classified short-text messages (Tweets) into news (N), events (E), opinions (O), deals (D) and private message (PM). They used Bag-Of-Words to classify the tweets. First, they were able to extract 8 features: author, presence of shortening of words, slangs, time-event phrases, opinioned words, emphasis on words, and currency and percentages. They used the author feature to determine the type of user. Corporate tweeters composed their message in a professional way. It uses less slangs, emotions and shortening for they wanted to convey their message clearly. On the other hand, personal tweets contains usage of slangs, emotions and shortening. These features can be used to determined corporate tweeters to personal tweeters. They collected 5407 English tweets. It contains 2107 N, 625 O, 1100 D, 1057 (E), and 518 (PM). It contained 6747 unique words. For the classification, they tried different setups: BOW, BOW and author feature (BOW-A), BOW and the seven features (BOW-7F), the 8 features (8F), and BOW and the 8 features (BOW-8F).

Another type of classification that could be used is k-nearest neighbor (k-NN). k-NN is an instance-based lazy learner. It means it only trains when a new instance comes in. k-NN computes for the k nearest instances (neighbors). Then, k-NN will use the neighbors’ categories to determine the class of the unknown instance. There are several ways to compute for the distance between the neighbors and the instances, Euclidean distance and Manhattan distance are some examples (Wajeed & Adilakshmi, 2011).

#### Tokenizer

Tokenizer is the module which segments a given text to tokens for further use in the natural language process. Tokens are usually the elements between spaces in the given input string. This module of natural language processing has a lot of difficulties to address such as tokenizing, email adresses, and uniform resource locators (URLs). Tokenizers today can identify that "15MB" is interpreted as "15 megabytes" even there is no space between '15' and 'MB', and words with punctuation marks are also read tokenized correctly. But these tokenizers, face two major problems, first is that the tokenizer performs its task independent of any knowledge contained in the system. Another problem is that tokenizers are hard coded in the system. Thus, systems using these tokenizers end up tokenizing the input text without even caring whether the output of the tokenization made sense.

The researchers invented a tokenizer that validates the proposed output of the tokenization in a linguistic knowledge component, and this proposal validation repeats until there is no more possible segmentation or the text is validated. Lastly, the tokenizer invented also includes a language specific data that contains a precedence hierarchy for punctuation (Bradlee et. al., 2001).

#### Sentence Splitter

The sentence splitter is a cascade of finite-state transducers which segments the text into sentences and this module is used for the POS tagger (Cunningham et al., 2002). This module uses the set of regular expression-based rules that define sentence breaks like using periods, exclamation marks and question marks. (Zeng et al., 2006)

#### Normalizer

Presence of text speaks, slangs and lingos are very high in SMS, social networks and microblog sites. The presence of these makes it difficult for the information extraction. In Aw and colleague’s work (2006), they viewed text normalization as a specialized machine translation problem, called SMS Normalization. They see that text speaks, slangs and lingos are just a variant of the English language. However, applying general machine translation will not work against SMS Machine Translation. General machine translation are based on non-standard words that have been well studied. However, with SMS, most of the lingoes, for example “b4” (before) and “bf” (boyfriend) are not formally defined by linguistics yet. These words can still evolved as time passed by and more new text speaks, slangs and lingos will be created by the younger generation.

There are two types of approach that was used in Aw and colleague’s paper (2006): basic word-based model and phrase based model. In basic word model, an SMS word will be mapped to exactly one word. In phrase based model, the SMS text will be split into k-phrases and the English will also be split into k-phrases. Then, it will map the SMS phrase to an English phrase.

#### POS Tagger

The tagger produces a part-of-speech tag as an annotation on every word or symbol. These annotations produced can be used by the grammar in order to increase its power and coverage (Cunningham et al., 2002).

#### Gazetteer

The gazetteer contains lists of cities, organizations, days of the week, etc. It does not only contain entities, but also names of useful indicators, such as typical company designators (e.g. ‘Ltd.’), titles, etc. The gazetteer lists are collected into finite state machines, which can match tokens (Cunningham et al., 2002).

#### Lemmatizer

Lemmatization is the reduction of inflectional forms and sometimes derivationally related forms of a word to a common base form. It uses vocabulary and morphological analysis to remove inflectional ending and return the root word (Manning et al., 2008). The traditional method of lemmatizing is to use morphological rules and dictionaries. However, with the presence of new words, it will be very difficult for the lemmatizer. Statistical method needs a large training corpus. StaLe is a lightweight statistical lemmatizer. In StaLe, the system produces result tokens based on the rules. Figure 3-2 shows StaLe’s lemmatization process. Each token will be ranked according to their confidence factor and then pruned according to their candidate check-up phase. Those who pass will be the lemma of that word. However, if no token passed the candidate check-up phase, the input word will be the lemma. The problem with StaLe is that it sometimes produces nonsense word resulting to a poorer precision than a traditional dictionary-based lemmatizer. The tested the lemmatizer.

Figure 3‑2. StaLe Lemmatization Process



#### Coreference Resolution

This module consists of a main module and a set of submodules. The main module is responsible for initializing the submodules and execute them in respective order then combine the results generated from the submodules and eventually perform some post processing over the result. There are three submodules in the main module: quoted text module, pleonastic it module and pronoun resolution module. The quoted text module submodule recognizes the quoted fragments inside the text. The identified fragments are used by the pronoun resolution submodule. The next module is the pleonastic it submodule, it is responsible for finding pleonastic occurences of “it”. The last and the main function of the coreference resolution module is in the pronoun resolution submodule. This submodule uses the results of the other two submodules after execution. The module works following an algorithm, first it inspects the appropriate context for all candidate antecedents for this kind of pronoun and choose the best antecedent if there is any. Then it creates the coreference chains from the individual anaphor/antecedent (this step is performed from the main coreference module) (Dimitrov, 2005).

#### Named Entity Recognition

Named entity recognition (NER) involves the automatic or semi-automatic processing of a series of words and then extracting or recognizing tokens in the text that refer to named entities (Lim et al., 2007). Named entities are phrases that contain the names of persons, organizations and locations.

## Information Extraction Architecture

This section discusses the different architectures that can be applied in an information extraction system.

### Template-Based Architecture

A template-based information extraction system uses templates to extract information. A template-based information extraction will only be able to extract the information that is deemed important by the user. The performance of the information extraction will now base on how the user created the templates (Corney et al., 2008).

#### An Open Architecture for Multi-Domain Information Extraction (Poibeau, 2001)

Thierry Poibeau has provided a general architecture for developing information extraction systems regardless of its domain (Poibeau, 2001). In his paper, he proposed an information extraction architecture that takes advantage of the capabilities of machine learning to help researchers define new templates (this is where the extracted information is being filled in) with respect to the IE system’s domain.

Poibeau’s architecture is divided into 5 main modules: (1) the module for extracting information from the structure of the text; (2) the module for named entity recognition which is responsible for recognizing places/dates/etc.; (3) the module for the semantic filters; (4) the module for the extraction of specific domain-dependent information; and lastly (5) the module for filling in a result template.

In module 1, a number of information is extracted from the structure of the input text. It is in this module where information that is embedded in the structure of the text is extracted like those that are written in HTML or XML formats. On the other hand, in module 2, relevant information is extracted/recognized through linguistic analysis. This module is responsible for recognizing the different named entities present in the input text like names, places, and dates. Poibeau made use of the finite-state tool *Intex* to develop this module. Furthermore, in module 3, text categorization is performed on the set of so-called “semantic signatures” that were produced from a semantic analysis of the input text. Poibeau made use of the French system Intuition™ to develop this module. In addition, in module 4, specific information like the specific relationships between named entities is extracted by applying a grammar of transducers or extraction patterns on the input text. Lastly, in module 5, all the information extracted from the input text are linked together to fill in a specific result template(s) that present(s) a summarized view of the extracted information. Figure 3-3 illustrates the general architecture proposed by Poibeau.

Figure 3‑3. Poibeau's General Architecture



### Adaptive Architecture

The problem with some information extraction system (knowledge based system) is that it is not portable and highly dependent to the domain. With sources rapidly growing and more diverse, it will be very hard for the information extraction system to extract as these text are unstructured, especially natural language. Another problem is that error propagates as it goes through each module, as the modules in information extraction architecture are cascaded. The use of machine learning techniques tries to solve these problems. Adaptive Information Extraction systems use machine learning techniques in order to automatically learn rules that will extract certain information (Turmo et al., 2006).

#### IE2 (Aone et al., 1998)

Aone and his team of researchers (1998) have presented an adaptive Information Extraction system that can be used to extract information from different type of texts like unstructured, structured and semi-structured texts. In their paper, they presented the architecture that they used in building the system. Aone’s IE system has six main modules in its architecture. Module 1 is responsible for the named-entity recognition part of the IE system. For this module, they used a commercial tool called NetOwl Extractor 3.0 to recognize general named-entity types. It is in this module where time/numerical expressions, names (persons/places/organizations), acronyms (organization names/locations), and semantic subtypes (country/city) are being recognized and extracted. Moving on, Module 2 or the Custom NameTag module is responsible for the recognition of restricted-domain named-entities by using pattern matching. The output phrases for this module are SGML-tagged (Standardized Generalized Markup Language) into the same input document. On the other hand, Modules 3 & 4 are responsible for SGML-tagging the phrases in the sentences that are considered to be values for the slots defined in the templates and works hand-in-hand. Module 3 or the PhraseTag module works by applying syntactico-semantic rules identify the noun phrases in the previously recognized/extracted named-entities. Module 4 or the EventTag module works by applying a set of custom-built syntactico-semantic multi-slot rules to recognize/extract events from the input sentence. Moving on, Module 5 or the Discourse Analysis Module is responsible for coreference resolution or the merging of the previously extracted noun phrases. This module was implemented by using three different strategies so that it can be modified to reach optimal performance regardless of the extraction scenario. Strategy A or the Rule-Based Strategy uses a set of custom-built rules to resolve definite noun phrases and singular personal pronoun coreference. Strategy B or the Machine Learning-Based Strategy uses a decision tree that has been formed from learning a corpus tagged with coreferences. Strategy C or the Hybrid Strategy uses Strategy A to filter false antecedents and then uses Strategy B to rank the remaining antecedents. In general, Module 5 is just merging the partial templates formed by the previous module. Lastly, Module 6 or the TempGen Module is responsible for the completion of the templates generated from the previous module by considering the consistency of the values in the slots of the event templates after resolving the noun phrases conferences and the generation of the output in the desired format. Figure 3-4 illustrates the architecture of the system proposed by Aone et al.

Figure 3‑4. Architecture of IE2 Adaptive Information Extraction System



#### EVIUS (Turmo & Rodriguez, 2000)

EVIUS is a multi-concept learning system for free text that follows a multi-strategy constructive learning (MCL) approach. The system also supports insufficient amounts of training corpora. M-TURBIO is the multilingual IE system where EVIUS is a component. The system’s input is both a partially-parsed semantically-tagged training corpus and a description of the desired target structure. The system’s approach to learn is by using MCL with constructive learning, closed-loop learning and deductive restructuring (Ko, 1998). EVIUS decides which concepts to learn and, updates the IE rule sets continuously. The system uses FOIL (First-Order Induction Learning) (Quinlan, 1990) to create an initial rule set from a set of positive and negative examples. Positive examples can be selected using a friendly environment either as text and ontology relations. Negative examples are automatically selected. If there are any uncovered positive examples remains after using FOIL, this is because there are insufficient examples. The system tries to develop recall by growing the positive examples with artificial examples (pseudo-examples). Combining the uncovered example vector and a randomly selected covered vector makes a pseudo-example. This is done as follows: for each dimension, one of both possible values is randomly selected as value for the pseudo-example. The new set of positive examples is now executed again using FOIL, the resulting set will be combined with the first rule set.

#### LearningPinocchio (Ciravegna & Lavelli, 2004)

LearningPinocchio is an adaptive information extraction systems that uses induction rules to extract information. Machine learning techniques are used to learn the rules over the training examples marked by XML tags. LearningPinocchio has two parts, preprocessor and modules. The preprocessor performs tokenization, lemmatization, POS tagging and Gazetteer lookup. After doing the preprocessor, it can now go to the modules. This is where the tags will be annotated. The modules may consists of NER, text zonings and other IE tasks. Figure 3-5 illustrates the architecture used by LearningPinocchio.

Each module has three modes: training mode, testing mode and production mode. Training mode is responsible for inducing the rules and learn how to apply IE rules in a specific scenario. The training mode accepts two inputs. First it needs the module definition that includes set of system parameters. Second is the preprocessed training corpus that has been tagged with XML. The output of the training mode is a set of rules that will be used in testing and production mode. The testing mode is for testing on unseen tagged corpus. This mode tells how well the module performed in a certain applications. The input for testing mode is a module with induced rules and a test corpus that has been tagged with information that needed to be extracted. In this mode, it is still possible to retrain the model by adjusting the parameters to improve performance. The output is corpus tagged with XML and a statistics on the performance of the module and details of the mistakes. The production mode is the one who will receive the tagged corpus and the XML tags to the corpus.

Figure 3‑5. Architecture of LearningPinocchio



For inducing rules, LearningPinocchio uses (LP)2, a covering algorithm specially for user-defined IE, to learn from training corpus marked with XML tags. It is a two-step process to induce the rules that will add XML tags to the text. Figure 3-6 shows the process of the inducing process of (LP)2. First, it induces tagging rules that will add preliminary tags. Second, it improves on the tagged rules by inducing correction rules.

Figure 3‑6. Rule Induction Step



A tagging rule is consist of a left-hand side, which is the pattern of conditions of sequence of words, and a right-hand side, which is the action that will insert the tags in the text. The rule induction algorithm uses positive examples to learn the rules. Positive examples are instances that has been manually tagged by the person. For each positive examples, the algorithm first initialize rules. Then, it will generalize the rules. Lastly, it will keep the best rules. The algorithm will repeat for each positive examples. Information, like word window, lexical items, lemma, lexical category, lexical case, and user-defined semantic classes could be used as condition in the initial rules. After getting the generalizations, it will be tested on training corpus to see if they will be accepted as best rules or contextual rules.

Best rules are rules that are highly dependable because they are able to cover most of the cases and their error rate is less than the threshold. These rules are sorted in decreasing number of covered cases. If the rules have the same number of matches, it is sorted according to their error rate. However, if they have the same number of matches and error rate, the one who has the generic condition is preferred. The algorithm only keeps *k* generalizations. Although best rules can correctly tag the information, the problem is the low recall. The role of the contextual rules is to increase the recall without sacrificing precision. Contextual rules are additional rules that will correct the problem.

Correction inducing rules are almost the same as the inducing rules. The difference is that the left hand side of the correction inducing rules contains the text and the tags and the right hand side, instead of adding tags, is shifting the misplaced tags. To select and apply the correction rules, the same algorithm as the inducing rules are used. Figure 3-7 illustrates the algorithm used by LearningPinocchio for choosing the best rules.

Figure 3‑7. Algorithm for Choosing the Best Rules



The information extraction process of LearningPinocchio consists of four (4) steps: initial tagging, contextual tagging, correction, and validation. The initial tagging will first tag the text. Next, the contextual tagging will further tags those that are missed by the initial tagging, until no more tags can be placed. The third step will correct the errors. Last step will validate the tags. Figure 3-8 shows the process of the information extraction.

Figure 3‑8. Information Extraction Process of LearningPinocchio



LearningPinocchio was tested in two languages, English and Italian. They trained the system on a corpus and tested the induced rules on unseen texts. They tested the system in two tasks: CMU Seminar announcements and Austin job announcements.

On CMU Seminar announcements, they performed tokenization and POS tagging. They did not do a gazetteer for a fair comparison. The IE must be able to extract speaker name, start time, end time, and location. They compared it to Rapier, symbolic-based (Califf, 1998), BWI, symbolic based, (Freitag & Kushmerick, 2000), SRV, WHISK (Soderland, 1999), and HMM, statistic-based (Freitag & McCallum, 1999). Based on the results, (LP)2 was able to achieve the highest score among the IE systems. (LP)2 was able to accurately extract the start time and end time, with 99.0% and 95% f-measure, respectively. However, it had difficulty in location and speaker name with f-measure 77.6% and 75.1%, respectively. Overall, (LP)2 has the highest performance in All Slots with score 86.0%.

On Austin job announcements, the IE systems must be able to extract message id, job title, salary offered, company offering the job, recruiter, state, city and country where the job is offered programming language, platform, application area, required and desired years of experience, required and desired degree and posting date. They performed the same preprocessing as on the CMU Seminar announcements. Based on the results, (LP)2 outperformed Rapier in almost all the aspects. Rapier was able to outperformed (LP)2 in salary, desired year, and desired degree. But in the overall performance, (LP)2 has a higher performance in All Slots with score 84.1%.

SOMIDIA (Chua, et al., 2013)

SOMIDIA uses an adaptive information extraction system that extract relevant information (English and Filipino) from different sources (i.e. blogs, social media sites, news articles). After crawling the internet for documents, the documents are fed to the information extraction system. First, it performs a tokenizer. They used OpenNLP to do the tokenization (OpenNLP, 2013). Then, it goes through the sentence splitter. It accepts a tokenized document. The system will now split the document into sentences. They used OpenNLP for the sentence detection (OpenNLP, 2013). After the sentence splitter, the document will be classified into English documents or Filipino documents. This is done because different information extraction modules will be applied for English and Filipino. For English, they used POS Tagger, Chunker, English NER, Coreference Resolution and English Extractor. For Filipino, they used Filipino NER and Filipino Extractor. The English information extraction process has POS Tagger, Chunker, English NER, Coreference Resolution and English Extractor. The Filipino information extraction process has Filipino NER and Filipino Extractor. For the Filpino NER, they build their own gazetteer for there are no existing gazetteer for Filipino. They used dictionary-based and rule-based approach in implementing the NER. Figure 3-9 describes the architecture of SOMIDIA.

In order for SOMIDIA to adapt to new instances, the rules must also be able to adapt. SOMIDIA has a pattern extractor module. This module of SOMIDIA is mainly responsible for extracting different patterns from the set of documents and seed words so that they can be later used for the extraction process. SOMIDIA defines a document as any text that is related to the domain of the extraction system. Moving on, this module of the system works in this manner. For each document, it will identify first the seed words present in the document. Seed words are words that will be extracted. And for each of the seed words identified, the module will try to generate possible rules by using Windowing, a term to describe the section of the document that is considered for computation. The module experiments with all possible combination of tokens and window setups to produce as much rules by taking into consideration a number of windowing concepts like the  minimum window size, maximum left window size, and maximum right window size. The minimum window size is the minimum number of tokens that is included in the window. In addition, the maximum left window size is the maximum number of tokens included in the window that is found to the left of the seed word. On the other hand, the maximum right window size is the maximum number of tokens included in the window that is found to the right of the seed word. After generating all possible rules from the combination of tokens and various window setups, it then stores the generated rules for that specific seed word in a HashMap together with the number of times the rules were generated. This process is done continuously until rules are generated for all the seed words in the document and until all of the documents are completely processed.

After the process of generating rules, the module will do some optimization of the rules generated to further improve the efficiency of the extraction module. The module will minimize rules by removing rules that fall into these two scenarios: (1) rules that occur only once because they are too specific and they would only work with a very small percentage of the documents and (2) rules that were able to extract more than its corresponding occurrence because these rules are too general and may have the tendency to extract irrelevant data.

Figure 3‑9. SOMIDIA's Architecture



## Twitter[[2]](#footnote-3)

Twitter is a microblogging social media platform wherein users may post messages of up to 140 characters long. Each of these posts are known as "tweets". Mainly, these tweets are expressions of a moment or idea. They may contain text, photos, and videos. Millions of tweets are shared in real time, every day.

A tweet may be replied to, retweeted, favorited, and may contain hashtags. A "reply" to a tweet is when another user comments or joins in the conversation of a tweet. A "retweet" is where you share the tweet of another user. A "favorite" indicates that a user likes the tweet. "Hashtags" assign a topic to the tweet. So if one searches for #WorldYouthDay, the search results will contain all tweets with related topics about World Youth Day. When a Twitter user "follows" another user, this means that they subscribe to the tweets posted by that user (Twitter, n.d.).

### Use of Twitter

Aside from Twitter’s social media aspect, Twitter has been used as a source of data for various fields, one of which is in disaster management (Imran et al., 2013). Other fields that Twitter data has contributed to linguistics (Mocanu et al., 2013), prediction (Tumasjan et al., 2010; Choy et al., 2012), real-time event detection (Sakaki et al., 2010), marketing (Jansen et al., 2009; Bollen et al., 2011), sentiment analysis and opinion mining (Pak et al., 2010), education (Grosseck et al., 2008; Junco et al., 2011), news casting (Phelan et al., 2009), medicine (Hawn, 2009; Chew & Eysenbach, 2010), business processes (Culnan et al., 2010)

### Twitter and Disasters

During disasters, the Filipino Twitter users tend to retweet tweets about request for help and prayer. Other tweets pertain to traffic updates, weather updates, observations, and class suspensions. While some users have a preference to post in English, there is still a larger number of user that use their native language when tweeting during disasters (Lee et al., 2013).

Table 3‑1. Examples of official government institution Twitter accounts and unified hashtags

|  |  |  |
| --- | --- | --- |
| **Category** | **Official Government Institution**  **Twitter Account** | **Unified Hashtag** |
| Typhoon | @dost\_pagasa | #(storm name)PH  (i.e. #YolandaPH, #GlendaPH) |
| Flood | @PAGASAFFWS, @MMDA | #FloodPH |
| Volcanic activities, earthquakes, and tsunamis | @phivolcs\_dost | #EarthquakePH |
| Relief and rescue efforts | @PIAalerts, @PIANewsDesk, @NDRRMC\_Open, @pcdspo, @DSWDserves | #ReliefPH  #RescuePH |
| Suspension of classes | @DepEd\_PH | #walangpasok |

As part of the disaster management of the Philippines for natural calamities, the government has released an official newsletter indicating the official social media accounts and hashtags (Official Gazette of the Republic of the Philippines, 2012). Table 3-1 shows some of the official twitter accounts of government institutions as well as the official hashtags being used during disasters. Table 3-2 shows the extractable information from the tweets per disaster.

Table 3‑2. Examples of disaster-related tweets with extractable information

|  |  |  |
| --- | --- | --- |
| **Type of Disaster** | **Tweet** | **Extractable Information** |
| Typhoon | @ANCALERTS:  NDRRMC says 77 dead, 220 injured, 5 missing due to Typhoon Glenda #GlendaPH | * 77 dead * 220 injured * 5 missing * Typhoon Glenda |
| Typhoon | @ABSCBNChannel2:  Bagyong Glenda patuloy na nagbabanta sa Luzon. #GlendaPH pic.twitter.com/2ygRWj6Z3D | * Typhoon Glenda * Luzon |
| Typhoon | @rapplerdotcom:  #GlendaPH: Marikina River now at alert level 1 [rplr.co/1mSTdnR](http://t.co/rqpfnzcLza) [pic.twitter.com/mECHfZfiyK](http://t.co/mECHfZfiyK) | * Marikina River * Alert level 1 |
| Typhoon | [@ABSCBNNews](https://twitter.com/ABSCBNNews/):  200 families in Laguna lose homes due to 'Glenda' [bit.ly/UfEDeO](http://t.co/6Kn0frqBsJ) [#southAlerts](https://twitter.com/search?q=%23southAlerts) [#GlendaPH](https://twitter.com/search?q=%23GlendaPH) | * 200 families * Laguna * Glenda |
| Earthquake | @dswdserves:  DSWD Region 11 prepositioned 12,170 food packs&55,206 assorted food for victims of recent quake in Davao Occ. [#EarthquakePH](https://twitter.com/search?q=%23EarthquakePH) [@dinkysunflower](https://twitter.com/dinkysunflower/) | * DSWD Region 11 * 12,170 food packs * 55,206 assorted food * Davao Occ |
| Earthquake | [@phivolcs\_dost](https://twitter.com/phivolcs_dost/):  No expected damage from 6.1-magnitude [#earthquakePH](https://twitter.com/search?q=%23earthquakePH) off Davao Occidental; aftershocks expected: [bit.ly/1ra30ZZa](http://t.co/IRX5SMSr3h) | * 6.1 magnitude * Davao Occidental |
| Earthquake | @manila\_bulletin:  BREAKING: 6.1 magnitude quake felt, east of Davao at 3:59PM. #EarthquakePH | * 6.1 magnitude * Davao * 3:59pm |
| Earthquake | @seanbofill:  Magnitude 6.1 earthquake recorded in Davao earlier today. #EarthquakePH | * Magnitude 6.1 * Davao |
| Flood | [@saabmagalona](https://twitter.com/saabmagalona/):  Ortigas st across La Salle GH ankle-deep [#floodph](https://twitter.com/search?q=%23floodph) | * Ortigas st * La Salle GH * Ankle-deep |
| Flood | [@MMDA](https://twitter.com/MMDA/):  [#FloodPH](https://twitter.com/search?q=%23FloodPH): As of 11:12 am, Orense to Estrella Southbound, leg deep, not passable to light vehicles | * 11:12am * Orense * Estrella Southbound * Leg deep * Not passable to light vehicles |
| Flood | @rqskye:  @MovePH MT @PIAalerts 5m: #FLOODPH ALERT: Greenhills, La Salle Street, San Juan, Metro Manila: Knee-high. #TrafficPH | * Greenhills * La Salle Street * San Juan * Metro Manila * Knee-high |
| Flood | @rqskye:  @MovePH MT @MakatiTraffic 11:27am: Flooded area in Brgy. Pio del Pilar: Medina St. corner... tl.gd/n\_1s2geia |#FloodPH #TrafficPH | * 11:27am * Brgy. Pio del Pilar * Medina St. corner |

## Evaluation Metrics

This section discusses the different metrics that will evaluate the performance of the information extraction system.

### F-measure

Precision and recall are the two primary metrics. Given a subject and a gold standard, precision is the percentage of cases that the subject was correctly classified as positive or true in the gold standard.

Recall is the percentage of cases in the gold standard that was correctly classified as positive or true by the subject. The two metrics are often combined as their harmonic mean known as the F-measure (Hripcsak and Rothschild, 2005).

Table 3‑3. Confusion Matrix (Davis and Goadrich, 2006)

|  |  |  |
| --- | --- | --- |
|  | **Actual Positive** | **Actual Negative** |
| **Predicted Positive** | True Positive | False Positive |
| **Predicted Negative** | False Negative | True Negative |

The True positive category means a positive instance is correctly predicted as positive while the False positive category denotes a negative instance is predicted as positive. Then, the True negative category signifies a negative instance is predicted correctly as negative while the False negative means a positive instance is predicted as negative (Davis and Goadrich, 2006). Table 3-3 show a confusion matrix of this.

### Kappa Statistics

The common way of summarizing interrater agreement among observers is the kappa statistics. The kappa allows to measure agreement not only by chance alone. The kappa is the observed agreement beyond chance divided by the maximum agreement beyond chance that is possible for the dataset. The general kappa formula is

where and are the observed and expected proportions of agreement (Malpica et al., 2005).

## Tools

This section discusses the different NLP tools that could be used in implementing the information extraction system.

### Apache OpenNLP (OpenNLP, 2011)

Apache OpenNLP is a Java-based library used for commonly used modules in NLP. It has tokenization, sentence segmentation, part-of-speech (POS) tagging, named entity recognition (NER), chunking, parsing and coreference resolution. It also uses maximum entropy and perceptron based machine learning to train the model. OpenNLP is extendable so that the module could be customized to suit the requirements.

#### Sentence Detector

The OpenNLP Sentence Detector knows if a punctuation mark signifies the end of the sentence or not. They defined the sentence as the longest whitespace trimmed character sequence between two punctuation marks. Their module looks for the first non-whitespace character. This signifies the beginning of the sentence. The last non-whitespace character signifies the end of the sentence. It can also be trained be providing it a model. It also has an evaluation module in which it computes the precision, recall and f-measure. The limitation of the sentence detector is that it cannot identify the sentence boundary based on the content.

#### Tokenizer

The OpenNLP tokenizer split the character sequence into word, punctuation marks, and numbers. OpenNLP has three implementations of the tokenizer: white space tokenizer, simple tokenizer and learnable tokenizer. The whitespace tokenizer just uses the whitespace as the boundary to identify as a token. The simple tokenizer uses the sequence of character class to detect the token. The learnable tokenizer uses a machine learning technique, maximum entropy, to detect the boundary. It is based on probability models. The tokenizer can also generate the probability of each tokens. It can also be trained by providing it a model.

#### Named Entity Recognition (NER)

The OpenNLP NER is used to detect names in the sentence. It accepts a tokenized string as input. OpenNLP offers pre-trained models that are trained on freely available corpora. The limitation is that it can only detect the names based on the corpora it used to train the model. It is highly recommended that it uses a custom model. For the training data, it should contain at least 15,000 sentence in order for the model to be trained. It can also evaluate the model by measuring the precision, recall and f-measure.

#### Document Categorization

The OpenNLP document categorizer uses maximum entropy to classify the document to a pre-defined category. The module needs a model because the requirements varies for different users. So, there are no pre-defined models.

#### Part-Of-Speech (POS) Tagging

The OpenNLP POS Tagging marks a word based on the word type and the context. A word can have multiple tags. OpenNLP POS Tagger uses probability model to correctly tag the word. To limit the number of possible tags, they used a dictionary tags. This contains the list of tags that will be used by the POS tagger. The use of the dictionary has two positive effect. First, it limits the assignment of inappropriate tags to the token is now limited. Second, OpenNLP POS tagging uses beam search algorithm to search for the most appropriate tags. By using a dictionary tags, it now limits the possibilities. In result, it search faster. The POS tagger has a built in evaluator that measures the accuracy of the model.

#### Chunker

The OpenNLP chunker groups syntactically correlated words. The chunker needs a tokenized and tagged sentence in order to perform the chunker. The limitation is that it only tells what words are in the same group. It does not tell the internal structure and the context. The chunker needs to be trained as the performance of the chunker is not that good outside the predefined model (news). It has an evaluation module where it can be tested using test dataset or cross validation. It measures the performance of the model using precision, recall and f-measure.

#### Parser

The OpenNLP parser is responsible for turning the input text into a tree. OpenNLP has two implementation, chunking parser and tree insert parse. The tree insert is still on experimental stage. The parser expect a whitespace tokenized sentence as inputs.

#### Coreference Resolution

The OpenNLP Coreference resolution is not fully developed yet. It is still very much limited as its implementation is only for noun mentions.

### ANNIE (Cunningham et al., 2002)

ANNIE or A Nearly-New IE System is a system that contains different modules for NLP tasks. ANNIE is part of the GATE framework. ANNIE uses finite state transducers and JAPE rules to implement the modules. ANNIE has a tokenizer, gazetteer, sentence splitter, semantic tagger and name matcher.

#### Tokenizer

ANNIE tokenizer uses rules to split the sentence into words, numbers and punctuations. The tokenizer can also differentiate lower case and upper case words as well as special punctuations. It uses grammar rules to tokenize the strings. By using grammar rules, the tokenizer will be more flexible.

There are different types of tokens that the tokenizer can use for the rules. First is the word tokens. A word is defined as any set of contiguous upper or lower case (including and only hyphen). The word has an attribute called “orth”, which can have the following values: upperInitial (first letter is upper case, the rest are lower case), allCaps (all uppercase letters), lowercase (all lower case), mixedCaps (mixture of upper and lower case letters). The second type of tokens are numbers. This are combinations of consecutive digits. Third type is the symbol. The symbol has two categories, currency (“$”,”₱”, “₤”) and symbol (“&”, “^”). Fourth type is the punctuation. The punctuation has three categories, start punctuation (‘(‘, ‘[‘), end punctuation (‘)’, ‘]’) and other punctuations (‘:’). The tokenizer consider a punctuation as an individual token. The fifth type of token is the SpaceToken. This are the whitespaces. There are two types of whitespace, space and control. Any set of contiguous space and control characters is considered as a SpaceToken.

#### Gazetteer

The gazetteer just contains the list of names, organizations, cities, days of the weeks, and others in plain text. It uses an index files to access the lists which will be compiled in the finite state machines.

#### Sentence Splitter

The sentence splitter uses a finite state transducers to split the text into sentences. It uses gazetteer to check if punctuation is part of an abbreviations or signals the end of the sentence. The sentence are annotated with the type “Sentence”, the breaks with “Split”. The sentence splitter is domain and application independent.

#### Part-Of-Speech (POS) Tagger

ANNIE POS Tagger uses a modified version of Brill Tagger. It uses lexicons and ruleset that has been trained in the Wall Street Journal corpus. However, the lexicon and rulesets can be changed based on the requirements. There are two additional lexicons, the lexicon for all caps and the lexicon for lowercase.

#### Semantic Tagger

The semantic tagger uses JAPE rules to annotate the entities. The grammar could be designed in such a way that it would recognize the entities. The output of the semantic tagger is the annotated text, which will be needed by the orthographic coreference.

#### Orthographic Coreference (OrthoMatcher)

The orthographic coreference is responsible for finding relationship between named entities. It tries to classify those unclassified entities. To prevent an entity from being recategorized, the matching rules makes sure that the two entities have the same category. It uses a lookup table to categorized different strings but same entity (i.e. Coca-Cola – Coke, IBM – Blue). It also has table called table of spurious matches. Here, strings matches but represents different entities.

### Twitter NLP Tools (Ritter et al., 2011)

The performance of standard NLP tools is degraded when used with Twitter data. This is because the standard NLP tools are trained with a structured news corpora. Twitter corpus, on the other hand, is noisy and informal in nature. Classifying named entities in tweets is challenging because of two things. The first is that tweets may contain a lot of named entity types and the second is that due to Twitter’s 140 character limit, it is difficult to determine the context of the named entity. The tool addresses the issue by rebuilding the NLP pipeline from POS up to NER. Testing and results show that each module of the tool performs better than its standard NLP counterpart.

The system, called T-NER, is composed of five modules: POS Tagger (T-POS), Chunker (T-CHUNK), Capitalization (T-CAP), Name Entity Segmentation (T-SEG), and Named Entity Classification (T-CLASS). The first three modules were trained with a dataset of 800 randomly sampled tweets and a 4-fold cross validation. The latter two were trained with 2,400 randomly sampled tweets and a 4-fold cross validation also. T-NER doubles the F1 score of the Stanford NER.

#### Part-of-Speech Tagging (T-POS)

Standard POS taggers suffers because of the nature of the Twitter corpora’s style, vocabulary, and lexical variations. To overcome the problems pertaining to style and vocabulary, they manually annotated the 800 tweets with tags from the Penn TreeBank tag set. They also added new tags which are Twitter-related such as “retweet”. To overcome the lexical variations, they performed a hierarchal clustering using Jcluster on 52 million tweets and applied Brown clusters to capture the lexical variations. T-POS uses Conditional Random Fields (CRF). T-POS shows a 41% reduction in error against the Stanford POS tagger.

#### Shallow Parsing or Chunking (T-CHUNK)

To create the model for T-CHUNK, they annotated the 800 tweets with tags from CoNLL Shared Task and used the set of shallow parser features described by Sha and Pereira (2003). They also included the Brown clusters and the POS tag features predicted by T-POS. T-CHUNK also applies CRF and has a 22% reduction in error over the OpenNLP chunker.

#### Capitalization (T-CAP)

One of the key features in recognizing named entities is the capitalization but this is unreliable in tweets. To address this problem, they incorporated information based on the entire content of the tweet to determine if the capitalization is informative. To train the classifier, they manually tagged the 800 tweets if it has an “informative” or “uninformative” capitalization. The classifier is a Support Vector Machine with features such as the fraction of words in the tweet which are capitalized, the fraction which appear in a dictionary of frequently lowercase/capitalized words but are not lowercase/capitalized in the tweet, the number of times the word ‘I’ appears lowercase and whether or not the ﬁrst word in the tweet is capitalized. Results show that T-CAP’s precision, recall, and F1 score is 0.77, 0.98, and 0.86 versus the baseline’s 0.70, 1.00, and 0.82

#### Segmenting the Named Entities (T-SEG)

T-SEG utilizes the features of T-CAP and is trained by manually tagging the 2,400 tweets with named entities. @usernames were not considered as a named entity. To implement T-SEG, they used a sequence-labeling task using IOB encoding for representing segmentations, CRF, Freebase, and T-POS, T-CHUNK, T-CAP, Brown clusters for feature generation.

#### Classifying Named Entities (T-CLASS)

Freebase is used as a baseline source of distant supervision of entities. To model the unlabeled entities, they made use of a Distant Supervision with Topic Models that applies LabeledLDA. They annotated the 2,400 tweets with the ten types which are popular on twitter: PERSON, GEO-LOCATION, COMPANY, PRODUCT, FACILITY, TV-SHOW, MOVIE, SPORTSTEAM, BAND, and OTHER. Note that a tweet can be annotated with two or more types and that the annotation is used for evaluation purposes. For training, they ran T-SEG on 60 million tweets and kept the entities that appeared 100 or more times. For each entity, they collected the words occurring in a context window of three words. The results show that T-CLASS outperforms the baseline and achieves a 25% increase in F1 score over the co-training algorithm of Collins and Singer (1999).

### Weka (Weka 3, n.d.)

Waikato Environment Knowledge Analysis (Weka) is a Java-based open source collection of machine learning algorithms that are used in data mining tasks. It contains various tools for preprocessing, classification, regression, clustering and visualization. It provides a library that could be used. It is also flexible as users can extend the API to customize the machine learning algorithms (Weka 3, n.d.).

### TwitIE (Bontcheva et al., 2013)

Bontcheva and her team of researchers have proposed an information extraction system that is specifically targeted for extracting relevant information from texts coming from microblogs. Bontcheva et al made use of the GATE ANNIE architecture in developing the system and took advantage of some of its built-in tools to further streamline the process of information extraction. In their paper, they presented how they designed the architecture of TwitIE and how they used the existing tools from ANNIE.

ANNIE offers a wide array of information extraction tools like tokenizer, sentence splitter, POS tagger, gazetteer lists, finite state transducer (from GATE’s built-in regular expression over annotation language), orthomatcher and coreference resolver but in the case of TwitIE, Bontcheva et al only reused the sentence splitter and name gazetteer because the other components/tools have to be modified to fit the specifics of microblog texts like noisiness, brevity, idiosyncratic language and social context.

Overall, TwitIE has the following main components: Language Identifier, Tokenizer, Gazetteer, Sentence Splitter, Normalizer, POS Tagger, and the Named Entity Recognizer. For the Language Identifier, TwitIE made use of the TextCat language identification algorithm, which heavily relies on n-gram frequency models to identify languages. For the Tokenizer, TwitIE followed Ritter’s Tokenization Scheme to treat abbreviations and URL’s as one token and hashtags and mentions as two token. This scheme also features orthography and capitalization preservation. For the Gazetteer, TwitIE used the unmodified version from ANNIE, which compiles lists of entities into finite state machines that can match text tokens. For the Sentence Splitter, TwitIE still used the unmodified version from ANNIE, which is a cascade of finite-state transducers that segments text into sentences. For the Normalizer, TwitIE made use of a combination of a generic spelling-correction dictionary and a social media-specific dictionary. The list of variations is also dynamic by using heuristics to correct spellings. For the POS Tagger, TwitIE made use of the same technique used by a Stanford Tagger, which was trained on tweets that were tagged using the Penn TreeBank Tagset. The improved tagger also included tag labels to support retweets, mentions, URL’s, hashtags and user mentions. Lastly, for the Named Entity Recognizer, TwitIE made use of the existing ANNIE NER as its pattern and just added some features that would help it support recognition of entities in social media texts. The following are the main functionalities of TwitIE:

* Custom Language Identifier to support language identification for social media/Twitter data;
* Twitter Tokenizer to enable the system to properly handle smilies, usernames, hashtags, URL’s and etc.;
* Twitter POS Tagger (also available as a standalone tool) to enable the system to properly perform POS tagging in Twitter data; lastly,
* Text Normalization PR to enable the system to perform correction of slangs and misspellings in the tweet.

# The FILIET System

This chapter presents the proposed system. It is divided into six sections. The first section will discuss the system overview. The second section outlines the objectives the system must be able to achieve. The third section tackles the scope and limitations of the system based on the outlined objectives. The fourth section presents the architectural design. The fifth section discusses the front-end and back-end features. Lastly, the sixth section will present the resources that will be used in implementing the system.

## System Overview

Filipino Information Extraction for Twitter (FILIET) is a hybrid information extraction system that incorporates the architectures of an adaptive IE system and a rule-based IE system for Filipino disaster related reports. The FILIET system will work with extracting information from Tweets that were written in the different variations of the Filipino language specifically the Taglish (Code Switching) and the TXTSPK variations. The system will follow the methodology described below. The disaster-related tweets will be loaded into the system. Then the system will then classify the type of disaster within each of the tweets. After undergoing the classification process, the tweets will now proceed to the information extraction engine of the system wherein the system will apply various information extraction techniques to extract the relevant information from the tweets with regards to its given type of disaster. Extracted information from the given tweets will vary based on the type of disaster it was classified as since each type of disaster has different set of information that can be extracted. For instance, information like intensity, signal number, and wind speeds can be further extracted from typhoon-related tweets while information like magnitude, epicenter and duration can be further extracted from earthquake-related tweets.

## System Objectives

This section will discuss the objectives of the system.

### General Objective

To develop an information extraction system that extracts relevant information from disaster-related texts from Twitter data and takes into consideration the different available variations of the Filipino language.

### Specific Objectives

The following are the specific objectives of the system:

1. To be able to preprocess the tweets;
2. To be able to extract relevant features from the tweets;
3. To classify the type of disaster for each tweet;
4. To extract relevant information common among the types of disaster (i.e. location);
5. To extract disaster-specific information from the tweet given the type of disaster;
6. To be able to handle TXTSPK and code-switching variations of the Filipino language;

## System Scope and Limitations

The system to be developed in this research is expected to be able to do a number of tasks that is within the scope of extracting information from Filipino disaster-related reports. These tasks include the following: Text Preprocessing, Feature Extraction, Disaster Classification, and actual Information Extraction.

The system must be able to perform some preprocessing techniques onto the input tweet. These preprocessing techniques shall only be limited to the following: (1) text normalization to include support for input tweets that were written in the TXTSPK format; (2) text tokenization, to enable word level analysis of the input tweet; (3) part-of-speech tagging, to enable semantic level analysis of the input tweet; (4) named-entity recognition for Filipino, to enable proper identification of Filipino named-entities; and lastly, (5) disaster keyword tagging, to enable proper recognition of disaster words in the input tweet.

Moving on, the system must be able to extract features from the input tweet. The features that will be extracted from the input tweet are categorized into two: (1) binary features, features that have discrete values 0 and 1; and (2) nominal features, features that have continuous values. For the binary features, they will only be limited to the following: (1) Presence features (presence of keywords like disaster words, mentions, hashtags, emoticons, retweets, and Code Switching), and (2) Trusted feature (whether the input tweet came from a trusted source). On the other hand, the nominal features will only be limited to the following: (1) Tweet length; (2) User; and lastly, (3) Location.

In addition, the system must be able to classify the input tweet based on the type of disaster that is present in the said input tweet. The system will only be limited to classification of tweets that involves natural disasters. Furthermore, these natural disasters will only be limited to the following: (1) Typhoons; (2) Floods; and lastly, (3) Earthquakes.

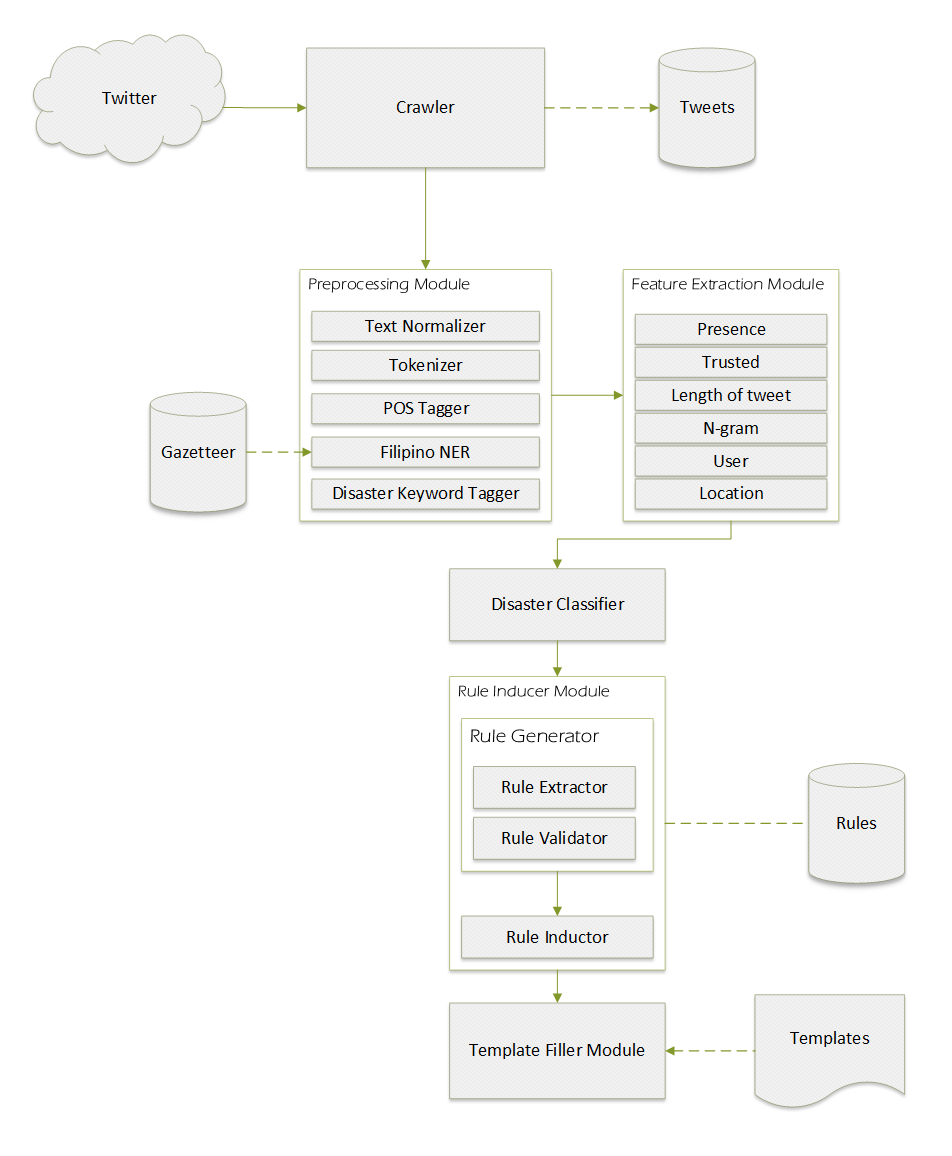
The system must be able to extract two types of information from the given input tweet. The two main types of information include the following: (1) General Disaster Information; and the, (2) Disaster-Specific Information. For the General Disaster Information, only the Date/Time and Location of the disaster shall be extracted from the input tweet. On the other hand, for the Disaster-Specific Information, the following shall be extracted from the input tweet: (a) for Typhoon-related tweets: the typhoon name, signal number, and wind speeds; (b) for Earthquake-related tweets: the magnitude; and lastly, (c) for Flood-related tweets: how deep the flood is and if the flood is passable to vehicles or not.

Lastly, by looking at the initial data, it was observed that TXTSPK and code-switching were the variations of the Filipino language evident in the tweets. This is why the system must be able to handle those two types of variations.

The data that will be used in the development of the system will come from the Twitter Web Crawler developed by the De La Salle University - College of Computer Studies as well as from the crawler to be developed by the group. The system will only be processing data that has already been filtered to be factual and disaster-related tweets.

## Architectural Design

Figure 4‑1. System Architecture of FILIET



### Data Sources

#### Twitter / Tweets

The data that will be collected will come from the tweets of the trusted Twitter accounts. Some of it will be provided by the Twitter Web Crawler developed by the De La Salle – College of Computer Studied, while the rest will come from the Crawler module to be discussed in the next section. The list of trusted Twitter accounts is based on the list provided by SOMIDIA.

The output of the crawler will be saved in a CSV file. Each entry in the CSV file will have the following content: <tweet ID>,<username>,”<tweet>”,”<date and time it was tweeted>”,<longitude>,<latitude>. Table 4-1 shows a sample of what can be seen in the CSV file.

Table 4‑1. Sample Entries of Tweets in CSV File

|  |  |
| --- | --- |
| **#** | **Sample Output** |
| 1 | 413991910886240256,BakaCarmiyan,"Kawawa naman nilindol sa Antique. <//33","Fri Dec 20 19:18:57 CST 2013",14.613499,121.0558286 |
| 2 | 414017377517326337,Ehmai123,"""@ANCALERTS: Magnitude 4.3 quake jolts Antique, Boracay http://t.co/c2BczJEa6Y"" Lindol everywhere :3","Fri Dec 20 21:00:09 CST 2013",14.527157,121.0033549 |

#### Gazetteer

The gazetteer is a text file that contains the list of names and locations to identify the proper nouns in the tweets. This will be used for the Filipino NER module. The plan is to updateand use SOMIDIA’s gazetteer. Table 4-2 shows a sample gazetteer for the storm names in the Philippines.

Table 4‑2. Sample Gazetter for Storm Names (Philippines)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Agaton  Amang  Ambo  Auring  Basyang  Bebeng  Bising  Butchoy  Caloy  Chedeng  Cosme  Crising  Dante  Dindo  Dodong  Domeng  Egay  Emong  Enteng  Ester | Falcon  Feria  Florita  Frank  Gener  Gloria  Goring  Gorio  Hanna  Helen  Henry  Huaning  Igme  Inday  Ineng  Isang  Jolina  Juan  Juaning  Julian | Kabayan  Karen  Katring  Kiko  Labuyo  Lando  Lawin  Luis  Marce  Maring  Milenyo  Mina  Nando  Neneng  Nina  Nonoy  Ofel  Ompong  Ondoy  Onyok | Pablo  Paeng  Pedning  Pepeng  Quedan  Queenie  Quiel  Quinta  Ramil  Ramon  Reming  Rolly  Santi  Seiang  Sendong  Siony  Tino  Tisoy  Tomas  Tonyo | Udang  Unding  Ursula  Usman  Venus  Vinta  Violeta  Viring  Waldo  Weng  Wilma  Winnie  Yayang  Yolanda  Yoyong  Yoyoy  Zeny  Zigzag  Zoraida  Zosimo |

#### Rules

The rules generated by the Rule Generator module will be stored in the database. Then, the Rule Inductor module will access the database to retrieve the rules. Table 4-3 shows a sample of the rules.

Table 4‑3. Sample Rules stored in the database

|  |  |
| --- | --- |
| Rules | Occurences |
| <string: naman><disaster><string:sa> AS Disaster | 1 |
| <string: naman><disaster><string:sa> AS Disaster | 4 |
| <string: magnitude><number>AS Intensity | 5 |
| <POS: NNS><location><POS: PSNS>AS Location | 2 |

#### Templates

For the templates, there will be two types of templates: general and specific. General templates are templates that will contain the following fields: (1) time, (2) location, and (3) type of disaster. For each type of disaster, there is a corresponding template. For typhoons, the template will have fields for: the typhoon name, signal number, and wind speeds. For earthquakes, the template will have a field for the magnitude. Lastly, for floods, the template will have a field for how deep the flood is and if the flood is passable to vehicles or not. Table 4-4 shows the general template while Tables 4-5 to 4-7 show disaster-specific templates.

Table 4‑4. Template for General Information

|  |  |
| --- | --- |
| Location |  |
| Time |  |

Table 4‑5. Template for Earthquakes

|  |  |
| --- | --- |
| Magnitude |  |

Table 4‑6. Template for Storms

|  |  |
| --- | --- |
| Typhoon Name |  |
| Signal Number |  |
| Wind Speed |  |

Table 4‑7. Template for Floods

|  |  |
| --- | --- |
| Depth of Flood |  |
| Is road passable |  |

### Crawler Module

This module will be crawling Twitter to retrieve tweets. The system will continuously collect the tweets using Twitter’s Stream API. The system will use a Twitter account where it follows the trusted users. The system will use a library called, Twitter4j, to implement the crawler method. Twitter4j is an unofficial Java library that integrates Twitter API (Twitter4j, n.d.).

### Preprocessing Module

This module will be responsible for preprocessing the input tweets before it is passed on to the information extraction module. This module will include the following text processing techniques: text normalizer, tokenizer and POS Tagger. After going through this module, the preprocess tweets will then be passed on the Information Extraction module.

#### Text Normalizer

The first step in preprocessing the input tweets is text normalization. The main responsibilities of the text normalizer are the following: (1) to convert the TXTSPK format of the tweets into full-word format so that the information when extracted will be consistent; (2) to correct the misspellings found in the tweets; (3) remove emoticons, links, and hashtags. The text normalizer will accept a text as input. The output of this module is the normalized tweets where the TXTSPK is converted to its full form, and links and emoticons are removed. For this module, the researchers will use two specific normalization tools from the existing pool of IE tools mentioned in the previous sections like OpenNLP’s built-in Normalizer (OpenNLP, 2011),TwitIE’s Text Normalization PR (Bontcheva, 2013), and NormAPI. Table 4-8 shows the sample input and its corresponding output.

Table 4‑8. Sample Input/Output for Text Normalizer

|  |  |
| --- | --- |
| **Input** | **Output** |
| <tweet>  Kawawa naman nilindol sa Antique. <//33  </tweet> | <tweet>  Kawawa naman nilindol sa Antique.  </tweet> |
| <tweet>  “”@ANCALERTS: Magnitude 4.3 quake jolts Antique, Boracay http://t.co/c2BczJEa6Y"" Lindol everywhere :3</tweet> | <tweet>  Magnitude 4.3 quake jolts Antique, Boracay Lindol everywhere  </tweet> |

#### Tokenizer

After normalizing the tweets, the tokenizer will now then split the input tweets into tokens like numbers, punctuations, words, abbreviations and other special characters like emoticons, hashtags, mentions and the like. The tokenizer will take as an input the normalized tweet from the Text Normalizer. The tokenizer will output an array containing the tokenized tweet in a form that is similar this. Tokenized = {“@<username>”, “<punctuations>”, “#<hashtag>”...} or an array that would contain all the tokens in a given tweet. For this module, the researchers will use two specific tokenization tools from the existing pool of IE tools mentioned in the previous sections, OpenNLP’s built-in Tokenizer (OpenNLP, 2011) and TwitIE’s Twitter tokenizer (Bontcheva, 2013).  Table 4-9 shows the sample input and its corresponding output.

Table 4‑9. Sample Input/Output Tokenizer

|  |  |
| --- | --- |
| **Input** | **Output** |
| <tweet>  Kawawa naman nilindol sa Antique.  </tweet> | <tweet>  [“Kawawa”, “naman”, “nilindol”, “sa”, “Antique”, “.”]  </tweet> |
| <tweet>  Magnitude 4.3 quake jolts Antique, Boracay Lindol everywhere  </tweet> | <tweet>  [“Magnitude”, “4.3”, “quake”, “jolts”, “Antique”, “,”, “Boracay”, “Lindol”, “everywhere”]  </tweet> |

#### POS Tagger

After tokenizing the tweets, the POS tagger will accept the tokenized Filipino tweet as an input and then, it will tag each of a token with its corresponding part-of-speech. Each of the tokens can be tagged as a noun, a verb, an adjective, an adverb or others. After tagging the tokens, the POS tagger will then output the tokens with their corresponding POS tag in the form of a text. For the module, the researchers are considering the following tools: ANNIE (Cunningham et al, 2002), OpenNLP (OpenNL, 2011), Twitter NLP tools (Ritter et al., 2011), TwitIE (Bontcheva, 2013), and Filipino Tagger Dictionary (Oco & Borra, 2011). Table 4-10 shows the sample input and output of POS tagger

Table 4‑10. Sample Input/Output POS Tagger

|  |  |
| --- | --- |
| **Input** | **Output** |
| <tweet>  [“Kawawa”, “naman”, “nilindol”, “sa”, “Antique”, “.”]  </tweet> | <tweet>  [“Kawawa\_ADUN”, “naman\_NPRO”, “nilindol”, “sa\_DECN”, “Antique\_NN”, “.\_PSNS”]  </tweet> |
| <tweet>  [“Magnitude”, “4.3”, “quake”, “jolts”, “Antique”, “,”, “Boracay”, “Lindol”, “everywhere”]  </tweet> | <tweet>  [“Magnitude\_NN:U”, “4.3”, “quake\_NN”, “jolts\_NNS”, “Antique\_NN”, “,\_PSNS”, “Boracay”, “Lindol”, “everywhere\_RB” </tweet> |

#### Filipino NER

The Filipino NER will be the one who will identify those proper nouns in the tweets. The module will accept the tweets that has passed through the preprocessing module. The output of the NER are tagged proper nouns in the tweet. For the gazetteer, the plan is to use SOMIDIA gazetteer and update the gazetteer. Table 4-11 shows the sample input and its corresponding output.

Table 4‑11. Sample Input/Output for Filipino NER

|  |  |
| --- | --- |
| **Input** | **Output** |
| <tweet>  [“Kawawa”, “naman”, “nilindol”, “sa”, “Antique”, “.”]  </tweet> | <tweet>  [“Kawawa\_ADUN”, “naman\_NPRO”, “nilindol”, “sa\_DECN”, “<location=”Antique\_NN”>, “.\_PSNS”]  </tweet> |
| <tweet>  [“Magnitude”, “4.3”, “quake”, “jolts”, “Antique”, “,”, “Boracay”, “Lindol”, “everywhere”]  </tweet> | <tweet>  [“Magnitude\_NN:U”, “4.3”, “quake\_NN”, “jolts\_NNS”, <location=“Antique\_NN”>, “,\_PSNS”, <location=“Boracay”>, “Lindol”, “everywhere\_RB” </tweet> |

Disaster Keyword Tagger

The disaster keyword tagger module is the module responsible for enclosing the disaster seed words present in the input tweet with a disaster tag. The disaster tag will be tagged in this manner: <disaster=word>. For the list of seed words that will be used in this module, the researchers will make use of the existing list of seed words from SOMIDIA. Table 4-12 shows the sample input and output of the disaster keyword tagger.

Table 4‑12. Sample Input/Output Disaster Tagger

|  |  |
| --- | --- |
| **Input** | **Output** |
| <tweet>  [“Kawawa”, “naman”, “nilindol”, “sa”, “Antique”, “.”]  </tweet> | <tweet>  [“Kawawa\_ADUN”, “naman\_NPRO”, “<disaster=nilindol/>”, “sa\_DECN”, “<location=”Antique\_NN/”>, “.\_PSNS”]  </tweet> |
| <tweet>  [“Magnitude”, “4.3”, “quake”, “jolts”, “Antique”, “,”, “Boracay”, “Lindol”, “everywhere”]  </tweet> | <tweet>  [“Magnitude\_NN:U”, “4.3”, “quake\_NN”, “jolts\_NNS”, <location=“Antique\_NN”>, “,\_PSNS”, <location=“Boracay”>, <disaster=“Lindol”>, “everywhere\_RB” </tweet> |

### Feature Extraction Module

This module is responsible for extracting the feature from the tweet. The module will extract the presence of disaster words, tweet length, character n-gram, user, location, and trusted accounts. The Feature Extraction Module will take the preprocessed tweets as inputs, then output the tweet with the features. Table 4-13 shows a sample of the features and their respective values.

#### Presence

The Presence feature is a binary feature that indicates the presence of keywords like disaster words, mentions, hashtags, emoticons, retweets, and Code Switching in the input tweet. The value of “1” is given if the keyword is present, else it is given “0”.

#### Trusted

The Trusted feature is a binary feature that helps in determining whether the input tweet came from a reliable source or account. Reliable sources are Twitter accounts that were automatically verified by Twitter or trusted social media accounts (i.e. accounts of news reporters). The value of “1” is given if the tweet came from a Trusted account, else it is given “0”. For the complete list of reliable sources or accounts, this module will use the existing list of SOMIDIA as its main reference.

#### Tweet Length

The Tweet Length feature essentially counts the length of the input tweet.

#### N-gram

The N-gram feature is mainly responsible for generating/extracting the different n-grams for the input tweets, specifically, the bi-gram and the tri-gram of the input tweets. In order to accomplish the n-gram generation/extraction tasks, the module will make use of the SRILM tool, which is specifically built for generating/extracting n-gram models.

#### User

The User feature will help in determining the type of disaster. For example, @dost\_pagasa will tweet about typhoons.

#### Location

The location feature is where the disaster occurred. There are instances which are specific for certain disasters, for example the disaster is flood, and the location given is usually a street. It can be also be a region, city or province for typhoon or earthquake related tweets.

Table 4‑13. Sample Features and Values

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Description** | **Type** | **Sample 1** | **Sample 2** |
| Kawawa naman nilindol sa Antique. <//33 | “”@ANCALERTS: Magnitude 4.3 quake jolts Antique, Boracay http://t.co/c2BczJEa6Y"" Lindol everywhere :3 |
| Hashtags | Presence feature that indicates the presence of hashtags “#” in the input tweet | Binary | 0 | 0 |
| Links | Presence feature that indicates the presence of links or URLs in the input tweet | Binary | 0 | 1 |
| Emoticons | Presence feature that indicates the presence of emoticons in the input tweet | Binary | 1 | 1 |
| Mentions | Presence feature that indicates the presence of mentions via the “@” sign in the input tweet | Binary | 0 | 1 |
| Retweet | Presence feature that indicates if the input tweet is a retweet via the presence of “RT” | Binary | 0 | 0 |
| Codeswitch | Presence feature that indicates if the input tweet employs codeswitching | Binary | 0 | 1 |
| Trusted | Trusted feature that indicates if the input tweet | Binary | 0 | 0 |
| Tweet Length | The length of the tweet | Nominal | 39 | 102 |
| User | The user of the tweet | Nominal | BakaCarmiyan | Ehmai123 |
| Location | The location mentioned in the tweet. | Nominal | Antique | Antique, Boracay |

### Disaster Classifier

The tweets will be classified first to the type of disaster: typhoon, earthquakes, and flood. This is to determine the type of information that will be extracted from the tweets. The classifier will accept the tweet as input. The output will add a tag that will determine the disaster. The classifier can be implemented using k-NN or Bag of Words (BoW). Table 4-14 shows the sample input and its corresponding output.

Table 4‑14. Sample Input/Output for Disaster Classifier

|  |  |
| --- | --- |
| **Input** | **Output** |
| <tweet>  [“Kawawa”, “naman”, “nilindol”, “sa”, <location= “Antique”/>, “.”]  </tweet> | <tweet disaster=”earthquake”>  [“Kawawa\_ADUN”, “naman\_NPRO”, “<disaster=nilindol/>”, “sa\_DECN”, “<location=”Antique\_NN/”>, “.\_PSNS”]  </tweet> |
| <tweet>  [ “Magnitude”, “4.3”, “quake”, “jolts”, “<location= Antique/>, “,”, “<location=Boracay/>, “Lindol”, “everywhere”]  </tweet> | <tweet disaster=”earthquake”>  [“Magnitude\_NN:U”, “4.3”, “quake\_NN”, “jolts\_NNS”, <location=“Antique\_NN”>, “,\_PSNS”, <location=“Boracay”>, <disaster=“Lindol”>, “everywhere\_RB” </tweet> |

### Information Extraction Module

The Rule Generator module is the module that is responsible for the generation of rules that can be used in the information extraction process by making use of different patterns that can be found in the input tweets. This module is further subdivided into two modules: the Rule Extractor and the Rule Validator.

#### Rule Extractor

The Rule Extractor will extract all the possible rules for the tweets by making use of different patterns that can be found in the input tweet. To generate the rules, the system will identify the seed words or the words that will be extracted that are present in the tweets. With the seed words, the system will apply the windowing technique around the seed words to extract as much patterns of combination of words, which, in turn can be converted to become extraction rules. To generate/extract as much patterns, the researchers will try various Maximum Windows Sizes, W = 1, 2, 3. After generating/extracting the patterns, they will now be converted to extraction rules. The rules that were generated will be stored in a HashMap together with its respective occurrence count (the number of times the rule was generated/extracted) so that it can be used in validating the validity of the generated rules in the next submodule. Table 4-15 shows the rules that can be generated by the rule extractor.

Table 4‑15. Sample Generated Rules

|  |  |
| --- | --- |
| **Tweet** | **Sample Rules** |
| <tweet disaster=”earthquake”>  [“Kawawa\_ADUN”, “naman\_NPRO”, “<disaster=nilindol/>”, “sa\_DECN”, “<location=”Antique\_NN/”>, “.\_PSNS”]  </tweet> | <string: sa><location><string: “.”>AS Location |
| <string: naman><disaster><string:sa> AS Disaster |
| <tweet disaster=”earthquake”>  [“Magnitude\_NN:U”, “4.3”, “quake\_NN”, “jolts\_NNS”, <location=“Antique\_NN”>, “,\_PSNS”, <location=“Boracay”>, <disaster=“Lindol”>, “everywhere\_RB” </tweet> | <string: magnitude><number>AS Intensity |
| <POS: NNS><location><POS: PSNS>AS Location |
| <POS: PSNS><location><disaster> AS Location |
| <location><disaster><string: everywhere> AS Disaster |

#### Rule Validator

The Rule Validator module is responsible for cleaning up the excess rules. This is done to improve the performance of the information extraction rules. Rules that are too specific will be removed, because the rules can only affect small part of the dataset. Rules that are too general will also be removed, because they might extract unwanted information. Rules are specific if they only occurred once in the rule extractor. Rules are general if they extract more than the occurrence count.

#### Rule Inductor

The Rule Inductor module will accept a tokenized and tagged tweets. The tweets contain XML tags that tells the type of disaster, location, time and other information. The, the plan is to generate rules using machine learning techniques and apply the rules to extract the information needed.

### Template Filler Module

After the rule induction module, the extracted information will now be placed to a slot in the template. Each type of disaster will have a different type of template as they have different information that can be extracted. Table 4-16 shows the sample extracted information.

Table 4‑16. Sample Template for Earthquake

|  |  |
| --- | --- |
| **Sample 1** | |
| General Information | |
| **Disaster** | Earthquake |
| **Location** | Antique |
| **Time** |  |
| Specific Information | |
| **Intensity** |  |
| **Sample 2** | |
| General Information | |
| **Disaster** | Earthquake |
| **Location** | Boracay, Antique |
| **Time** |  |
| Specific Information | |
| **Intensity** | Magnitude 4.3 |

## Physical Environment and Resources

This section outlines the minimum software and hardware requirements of the system.

### Minimum Software Requirements

* Windows 7
* MySQL
* Java 1.7.0

### Minimum Hardware Requirements

* 2 GB RAM
* Server

# References

alias-i. (2011). What is lingpipe?. Retrieved from http://alias-i.com/lingpipe/

Aone, C., Halverson, L., Hampton, T., & Ramos-Santacruz, M. (1998, April). SRA: Description of the IE2 system used for MUC-7. In *Proceedings of the seventh message understanding conference (MUC-7)*.

Apache Software Foundation. (2010). Welcome to apache opennlp. Retrieved from https://opennlp.apache.org/

Asahara, M., & Matsumoto, Y. (2003, May). Japanese named entity extraction with redundant morphological analysis. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1 (pp. 8-15). Association for Computational Linguistics.

Aw, A., Zhang, M., Xiao, J., & Su, J. (2006, July). A phrase-based statistical model for SMS text normalization. In Proceedings of the COLING/ACL on Main conference poster sessions (pp. 33-40). Association for Computational Linguistics.

Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. Journal of Computational Science, 2(1), 1-8.

Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M.A., Maynard, D., Aswani, N. TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013).

Bradlee, D., Knoll, S., & Pentheroudakis, J. (2001). Tokenizer for a natural language processing system.

Cheng, H., Chua, J., Co, J., & Magpantay, A. B. (2013). Social media monitoring for disasters. Unpublished undergraduate thesis, De La Salle University, Manila, Philippines.

Cheng, T. T., Cua, J. L., Tan, M. D., Yao, K. G., & Roxas, R. E. (2009, October). Information extraction from legal documents. In Natural Language Processing, 2009. NLP'09. Eighth International Symposium on (pp. 157-162). IEEE.

Chew, C., & Eysenbach, G. (2010). Pandemics in the age of Twitter: content analysis of Tweets during the 2009 H1N1 outbreak. PloS one, 5(11), e14118.

Choy, M., Cheong, M., Laik, M. N., & Shung, K. P. (2012). US Presidential Election 2012 Prediction using Census Corrected Twitter Model. *arXiv preprint arXiv:1211.0938*.

Califf, Mary Elaine. 1998. Relational Learning Techniques for Natural Language Information Extraction. Ph.D. thesis, Univ. of Texas at Austin, http://www.cs.utexas.edu/users/mecaliff.

Ciravegna, F., & Lavelli, A. (2004). LearningPinocchio: Adaptive information extraction for real world applications. Natural Language Engineering, 10(02), 145-165.

Corney, D., Byrne, E., Buxton, B., & Jones, D. (2008). A logical framework for template creation and information extraction. In *Data Mining: Foundations and Practice* (pp. 79-108). Springer Berlin Heidelberg.

Culnan, M. J., McHugh, P. J., & Zubillaga, J. I. (2010). How large US companies can use Twitter and other social media to gain business value. MIS Quarterly Executive, 9(4), 243-259.

Cunningham, H. (2002). GATE, a general architecture for text engineering.Computers and the Humanities, 36(2), 223-254.

Cunningham, H., Maynard, D., Bontcheva, K., & Tablan, V. (2002, July). A framework and graphical development environment for robust NLP tools and applications. In ACL (pp. 168-175).

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Ursu, C., Dimitrov, M. & Aswani, N. (2002). Developing language processing components with GATE (a user guide). University of Sheffield, Sheffield UK, 5.

Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In Proceedings of the 23rd international conference on Machine learning (pp. 233-240). ACM.

Dimitrov, M. (2005). A Lightweight Approach to Coreference Resolution for Named Entities in Text Marin Dimitrov, Kalina Bontcheva, Hamish Cunningham and Diana Maynard. Anaphora Processing: Linguistic, cognitive and computational modelling, 263, 97.

Dung, T. Q., & Kameyama, W. (2007, March). A proposal of ontology-based health care information extraction system: Vnhies. In Research, Innovation and Vision for the Future, 2007 IEEE International Conference on (pp. 1-7). IEEE.

Farkas, R. (2009). Machine learning techniques for applied information extraction (Doctoral dissertation, University of Szeged).

Feilmayr, C. (2011, August). Text Mining-Supported Information Extraction: An Extended Methodology for Developing Information Extraction Systems. InDatabase and Expert Systems Applications (DEXA), 2011 22nd International Workshop on (pp. 217-221). IEEE.

Freitag, D. (2000). Machine learning for information extraction in informal domains. Machine learning, 39(2-3), 169-202.

Freitag, D., & Kushmerick, N. 2000. Boosted wrapper induction. In: Basili, R., Ciravegna, F., & Gaizauskas, R. (eds), ECAI2000 Workshop on Machine Learning for Information Extraction. http://www.dcs.shef.ac.uk/ fabio/ecai-workshop.html.

Freitag, D., & McCallum, A. 1999. Information Extraction with HMMs and Shrinkage.In: AAAI-99 Workshop on Machine Learning for Information Extraction.

Gao, H., Barbier, G., & Goolsby, R. (2011). Harnessing the crowdsourcing power of social media for disaster relief. Intelligent Systems, IEEE, 26(3), 10-14. doi: 10.1109/MIS.2011.52

Ghedin, G. (2011, November 16). A social media lesson. from the philippines. Retrieved from <http://www.youngdigitallab.com/en/social-media/a-social-media-lesson-from-the-philippines>

Grishman, R. (1997). Information extraction: Techniques and challenges. InInformation Extraction A Multidisciplinary Approach to an Emerging Information Technology (pp. 10-27). Springer Berlin Heidelberg.

Grosseck, G., & Holotescu, C. (2008, April). Can we use Twitter for educational activities. In 4th international scientific conference, eLearning and software for education, Bucharest, Romania.

Han, B., & Baldwin, T. (2011, June). Lexical normalisation of short text messages: Makn sens a# twitter. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 368-378). Association for Computational Linguistics.

Hawn, C. (2009). Take two aspirin and tweet me in the morning: how Twitter, Facebook, and other social media are reshaping health care. Health affairs, 28(2), 361-368.

Hripcsak, G., & Rothschild, A. S. (2005). Agreement, the f-measure, and reliability in information retrieval. Journal of the American Medical Informatics Association, 12(3), 296-298.

Imran, M., Elbassuoni, S., Castillo, C., Diaz, F., & Meier, P. (2013, May). Practical extraction of disaster-relevant information from social media. In*Proceedings of the 22nd international conference on World Wide Web companion* (pp. 1021-1024). International World Wide Web Conferences Steering Committee.

Intarapaiboon, P., Nantajeewarawat, E., & Theeramunkong, T. (2009). Information extraction from Thai text with unknown phrase boundaries. InAdvances in Knowledge Discovery and Data Mining (pp. 525-532). Springer Berlin Heidelberg.

Jansen, B. J., Zhang, M., Sobel, K., & Chowdury, A. (2009). Twitter power: Tweets as electronic word of mouth. Journal of the American society for information science and technology, 60(11), 2169-2188.

Junco, R., Heiberger, G., & Loken, E. (2011). The effect of Twitter on college student engagement and grades. Journal of Computer Assisted Learning, 27(2), 119-132.

Ko, H. (1998). Empirical assembly sequence planning: A multistrategy constructive learning approach. Machine Learning and Data Mining. John Wiley & Sons LTD.

Krishnamurthy, R., Li, Y., Raghavan, S., & Reiss, F. SystemT: A system for declarative information extraction. SIGMOD Record, 37. Retrieved May 28, 2014, from http://www.almaden.ibm.com/cs/projects/avatar/

Lim, N. R., New, J. C., Ngo, M. A., Sy, M., & Lim, N. R. (2007). A Named-Entity Recognizer for Filipino Texts. Proceedings of the 4th NNLPRS.

Lee, J. B., Ybañez, M., De Leon, M. M., Estuar, M., & Regina, E. (2013). Understanding the Behavior of Filipino Twitter Users during Disaster. GSTF Journal on Computing, 3(2).

Lee, Y. S., & Geierhos, M. (2009). Business specific online information extraction from german websites. In Gelbukh, A. (Eds.), CICLing (pp. 369-381). Germany: Springer-Verlag Berlin Heidelberg.

Loponen, A., & Järvelin, K. (2010). A dictionary-and corpus-independent statistical lemmatizer for information retrieval in low resource languages. In*Multilingual and Multimodal Information Access Evaluation* (pp. 3-14). Springer Berlin Heidelberg.

Malpica, A., Matisic, J. P., Niekirk, D. V., Crum, C. P., Staerkel, G. A., Yamal, J. M., ... & Follen, M. (2005). Kappa statistics to measure interrater and intrarater agreement for 1790 cervical biopsy specimens among twelve pathologists: qualitative histopathologic analysis and methodologic issues.Gynecologic oncology, 99(3), S38-S52.

Manguilimotan, E., & Matsumoto, Y. (2009). Factors affecting part-of-speech tagging for tagalog. In PACLIC (pp. 763-770).

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 1, p. 6). Cambridge: Cambridge university press.

Mocanu, D., Baronchelli, A., Perra, N., Gonçalves, B., Zhang, Q., & Vespignani, A. (2013). The Twitter of Babel: Mapping world languages through microblogging platforms. *PloS one*, *8*(4), e61981.

Maynard, D., Bontcheva, K., & Rout, D. (2012). Challenges in developing opinion mining tools for social media. Proceedings of@ NLP can u tag# user\_generated\_content.

Maynard, D., Peters, W., & Li, Y. (2006, May). Metrics for evaluation of ontology-based information extraction. In International world wide web conference.

McCallum, A. (2005). Information extraction: Distilling structured data from unstructured text. Queue, 3(9), 48-57.

Meier, P. (2013, September 18). [Web log message]. Retrieved from http://irevolution.net/2013/09/18/micromappers/

N´edellec C., Nazarenko A., & Bossy R. (2009). Information extraction. In S. Staab & R. Studer (Eds), Handbook on ontologies (pp 683-685). Dordecht: Springer.

Nebhi, K. (2012). Ontology-Based information extraction for french newspaper articles. In KI 2012: Advances in Artificial Intelligence (pp. 237-240). Springer Berlin Heidelberg.

Official Gazette of the Republic of the Philippines. (2012, July 21). Prepare for natural calamities: Information and resources from the government. Retrieved July 15, 2014, from http://www.gov.ph/crisis-response/government-information-during-natural-disasters/

OpenNLP, A. (2011). Apache Software Foundation. URL http://opennlp. apache. org.

[Özsu, M. T., & Liu, L. (2009). Text Categorization. Encyclopedia of database systems (p. 3044). New York: Springer.](http://www.bibme.org/)

Pak, A., & Paroubek, P. (2010, May). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In LREC.

Pham, L. V., & Pham, S. B. (2012, August). Information Extraction for Vietnamese Real Estate Advertisements. In Knowledge and Systems Engineering (KSE), 2012 Fourth International Conference on (pp. 181-186). IEEE.

Phelan, O., McCarthy, K., & Smyth, B. (2009, October). Using twitter to recommend real-time topical news. In Proceedings of the third ACM conference on Recommender systems (pp. 385-388). ACM.

Poibeau, T. An Open Architecture for Multi-Domain Information Extraction. IAAI-01. Retrieved May 28, 2014, from [www.aaai.org](http://www.aaai.org)

Quinlan, J. R. (1990). Learning logical definitions from relations. Machine learning, 5(3), 239-266.

Ritter, A., Clark, S., & Etzioni, O. (2011, July). Named entity recognition in tweets: an experimental study. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (pp. 1524-1534). Association for Computational Linguistics.

Sakaki, T., Okazaki, M., & Matsuo, Y. (2010, April). Earthquake shakes Twitter users: real-time event detection by social sensors. In Proceedings of the 19th international conference on World wide web (pp. 851-860). ACM.

Shafiei, M., Wang, S., Zhang, R., Milios, E., Tang, B., Tougas, J., & Spiteri, R. (2007, April). Document representation and dimension reduction for text clustering. In Data Engineering Workshop, 2007 IEEE 23rd International Conference on (pp. 770-779). IEEE.

Shen, D. (2009). Text Categorization. Encyclopedia of Database Systems, 3041-3044.

Soderland, S. 1999. Learning Information Extraction Rules for Semi-structured and Free Text. Machine Learning, **34**(1), 233–272.

Southgate, R., Roth, C., Schneider, J., Shi, P., Onishi, T., Wengner, D., Amman, W., Ogallo, L., Beddington J., & Murray, V. (2013). Using science for disaster risk reduction. Retrieved from www.preventionweb.net/go/scitech

Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., & Demirbas, M. (2010, July). Short text classification in twitter to improve information filtering. InProceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval (pp. 841-842). ACM.

Stockdale, C. & McIntyre, D.A. (2011, May 09). The ten nations where facebook rules the internet. Retrieved from <http://247wallst.com/technology-3/2011/05/09/the-ten-nations-where-facebook-rules-the-internet/>

Stone, R. (2004). Natural language processing challenges and advantages for philippine languages. Proceedings from 1st Natural Language Processing Research Symposium (pp.81-86).

Téllez-Valero, A., Montes-y-Gómez, M., & Villaseñor-Pineda, L. (2005). A machine learning approach to information extraction. In Computational Linguistics and Intelligent Text Processing (pp. 539-547). Springer Berlin Heidelberg.

Tumasjan, A., Sprenger, T. O., Sandner, P. G., & Welpe, I. M. (2010). Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment. ICWSM, 10, 178-185.

Twitter4J - A Java library for the Twitter API. (n.d.). *Twitter4J - A Java library for the Twitter API*. Retrieved July 29, 2014, from http://twitter4j.org/en/

Universal McCann. (2008). Power to the people: Social media tracker wave 3. Retrieved from <http://web.archive.org/web/20080921002044/http://www.universalmccann.com/Assets/wave_3_20080403093750.pdf>

Wajeed, M. A., & Adilakshmi, T. (2011). Using KNN Algorithm for Text Categorization. In *Computational Intelligence and Information Technology* (pp. 796-801). Springer Berlin Heidelberg.

Wei, G., Gao, X., & Wu, S. (2010, July). Study of text classification methods for data sets with huge features. In Industrial and Information Systems (IIS), 2010 2nd International Conference on (Vol. 1, pp. 433-436). IEEE.

Weka 3: Data Mining Software in Java. (n.d.). *Weka 3*. Retrieved July 15, 2014, from http://www.cs.waikato.ac.nz/ml/weka/

Xu, H., Stenner, S. P., Doan, S., Johnson, K. B., Waitman, L. R., & Denny, J. C. (2010). MedEx: a medication information extraction system for clinical narratives. Journal of the American Medical Informatics Association, 17(1), 19-24.

Zeng, Q. T., Goryachev, S., Weiss, S., Sordo, M., Murphy, S. N., & Lazarus, R. (2006). Extracting principal diagnosis, co-morbidity and smoking status for asthma research: evaluation of a natural language processing system. BMC medical informatics and decision making, 6(1), 30.

Zhou, G., & Su, J. (2002, July). Named entity recognition using an HMM-based chunk tagger. In proceedings of the 40th Annual Meeting on Association for Computational Linguistics (pp. 473-480). Association for Computational Linguistics.

Zhou, S., Ling, T. W., Guan, J., Hu, J., & Zhou, A. (2003, March). Fast text classification: a training-corpus pruning based approach. In Database Systems for Advanced Applications, 2003.(DASFAA 2003). Proceedings. Eighth International Conference on (pp. 127-136). IEEE.

# Appendix

## Appendix A

Table 6‑1. Results of the study conducted by University McCann

|  |  |  |
| --- | --- | --- |
| **Category** | **Rank** | **Margin from Rank 1** |
| Blog Readership | 2 (90.3%) | 1.8% (South Korea) |
| Starting a Blog | 4 (65.8%) | 5.9% (South Korea) |
| Social Networks | 1 (83.1%) | -- |
| Photo Sharing | 1 (86.4%) | -- |
| Uploading Videos | 2 (60.5%) | 7.8% (China) |
| Watching Videos | 1 (98.6%) | -- |
| Podcasts | 5 (61.8%) | 12.5% (China) |
| RSS | 6 (45.2%) | 11.4% (Russia) |

## Appendix B

Table 6‑2. Example of Filipino Morphemes

|  |  |  |  |
| --- | --- | --- | --- |
| **Morpheme Element** | **Root Word** | **Suffix** | **Filipino Word** |
| Elision | bigay | na- ; -an | nabigyan |
| Epenthesis | patay | -an | patayan |
| Metathesis | peteh (cebuano) | -en | pehten |
| Replacement | utos | -an | utusan |
| Nasal Assimilation | bigay | paN- | pamigay |
| Infixation | kain | -um- | kumain |
| Reduplication | matamis | - | matamistamis |

1. MicroMappers digital disaster response system. http://micromappers.com/ [↑](#footnote-ref-2)
2. http://www.twitter.com/ [↑](#footnote-ref-3)