FILIET: An Information Extraction System

For Filipino Disaster-Related Tweets

A 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

April 20, 2015

Abstract

The Philippines is considered the social media capital of the world, and the role of social media has become even more pronounced in the country during disasters. Twitter is among the many forms of social media. As experienced, information and data shared through Twitter have helped individuals, institutions, and organizations (government, public, and private) during emergency response, in making decisions, conducting relief efforts, and practically mobilizing people to humanitarian causes. However, extracting the most relevant information from Twitter is a challenge because natural languages do not have a particular structure immediately useful when programming. Another problem that information extraction faces is that some languages, like Filipino, is morphologically rich, making it even more difficult to extract information. Therefore, the goal of this research is to create Filipino Information Extraction Tool for Twitter (FILIET), a system that extracts relevant information from disaster-related tweets composed in Filipino.The system was able to classify the tweets to caution and advice (CA), casualties and damages (CD), Donations (D). After classification, it was able to extract relevant information by applying rules catered to the category the tweet was classified in. The extractor could extract information such as typhoon signals, suspension of classes, casualties, damages, and items donated with a 0.4 f-measure score.

***Keywords***: information extraction, disaster management, Twitter

Table of Contents

[Abstract i](#_Toc415573167)

[1.0 Research Description 1-1](#_Toc415573168)

[1.1 Overview of the Current State of Technology 1-1](#_Toc415573169)

[1.2 Research Objectives 1-2](#_Toc415573170)

[1.2.1 General Objective 1-2](#_Toc415573171)

[1.2.2 Specific Objectives 1-2](#_Toc415573172)

[1.3 Scope and Limitations of the Research 1-2](#_Toc415573173)

[1.4 Significance of the Research 1-3](#_Toc415573174)

[1.5 Research Methodology 1-4](#_Toc415573175)

[1.6 Investigation and Research Analysis 1-4](#_Toc415573176)

[1.6.1 System Design 1-4](#_Toc415573177)

[1.6.2 Sprints 1-5](#_Toc415573178)

[1.6.3 Sprint Planning Meetings 1-5](#_Toc415573179)

[1.6.4 Scrum Meetings 1-5](#_Toc415573180)

[1.6.5 System Development 1-5](#_Toc415573181)

[1.6.6 System Integration and Testing 1-5](#_Toc415573182)

[1.6.7 System Evaluation 1-5](#_Toc415573183)

[1.6.8 Documentation 1-5](#_Toc415573184)

[1.6.9 Calendar of Activities 1-6](#_Toc415573185)

[2.0 Review of Related Works 2-1](#_Toc415573186)

[2.1 Machine Learning-Based Information Extraction Systems 2-1](#_Toc415573187)

[2.2 Rule-Based Information Extraction Systems 2-2](#_Toc415573188)

[2.3 Template-Based Architecture 2-4](#_Toc415573189)

[2.4 Ontology-Based Information Extraction Systems 2-5](#_Toc415573190)

[2.5 Other Information Extraction Systems 2-6](#_Toc415573191)

[2.6 Disaster Management – Relief Operations 2-9](#_Toc415573192)

[2.7 Twitter and Disaster 2-9](#_Toc415573193)

[3.0 Theoretical Framework 3-1](#_Toc415573194)

[3.1 Information Extraction 3-1](#_Toc415573195)

[3.1.1 Information Extraction Modules 3-2](#_Toc415573196)

[3.2 Information Classification 3-4](#_Toc415573197)

[3.2.1 Document Representation 3-4](#_Toc415573198)

[3.2.2 Dimensionality Reduction (Feature Selection) 3-5](#_Toc415573199)

[3.2.3 Classification 3-5](#_Toc415573200)

[3.2.4 Term Frequency-Inverse Document Frequency (TF-IDF) 3-6](#_Toc415573201)

[3.3 Information Extraction Architecture 3-6](#_Toc415573202)

[3.3.1 Adaptive Architecture 3-6](#_Toc415573203)

[3.4 Ontology 3-12](#_Toc415573204)

[3.4.1 Ontology Design 3-13](#_Toc415573205)

[3.4.2 Ontology Population 3-13](#_Toc415573206)

[3.5 Twitter 3-14](#_Toc415573207)

[3.5.1 Use of Twitter 3-14](#_Toc415573208)

[3.5.2 Twitter and Disasters 3-15](#_Toc415573209)

[3.6 Evaluation Metrics 3-16](#_Toc415573210)

[3.6.1 F-measure 3-16](#_Toc415573211)

[3.6.2 Kappa Statistics 3-17](#_Toc415573212)

[3.7 Tools 3-17](#_Toc415573213)

[3.7.1 ANNIE (Cunningham et al., 2002) 3-17](#_Toc415573214)

[3.7.2 Weka (Weka 3, n.d.) 3-18](#_Toc415573215)

[3.7.3 JENA API (McBride, 2002) 3-18](#_Toc415573216)

[3.7.4 ArkNLP (Gimpel et al., 2011) 3-19](#_Toc415573217)

[3.7.5 NormAPI (Nocon et al., 2014) 3-19](#_Toc415573218)

[4.0 The FILIET System 4-1](#_Toc415573219)

[4.1 System Overview 4-1](#_Toc415573220)

[4.2 System Objectives 4-1](#_Toc415573221)

[4.2.1 General Objective 4-1](#_Toc415573222)

[4.2.2 Specific Objectives 4-1](#_Toc415573223)

[4.3 System Scope and Limitations 4-1](#_Toc415573224)

[4.4 Architectural Design 4-3](#_Toc415573225)

[4.4.1 Crawler Module 4-4](#_Toc415573226)

[4.4.2 Preprocessing Module 4-4](#_Toc415573227)

[4.4.3 Feature Extraction Module 4-6](#_Toc415573228)

[4.4.4 Category Classifier Module 4-6](#_Toc415573229)

[4.4.5 Rule Inductor Module 4-7](#_Toc415573230)

[4.4.6 Ontology Population Module 4-7](#_Toc415573231)

[4.4.7 Data Sources 4-8](#_Toc415573232)

[4.5 System Functions 4-10](#_Toc415573233)

[4.5.1 Tweet Retrieval 4-10](#_Toc415573234)

[4.5.2 Information Extraction 4-11](#_Toc415573235)

[4.5.3 Ontology Population 4-11](#_Toc415573236)

[4.5.4 Ontology Retrieval 4-12](#_Toc415573237)

[4.6 Physical Environment and Resources 4-12](#_Toc415573238)

[4.6.1 Minimum Software Requirements 4-12](#_Toc415573239)

[4.6.2 Minimum Hardware Requirements 4-12](#_Toc415573240)

[5.0 Design and Implementation Issues 5-1](#_Toc415573241)

[5.1 Resource Gathering 5-1](#_Toc415573242)

[5.1.1 Dataset Building 5-1](#_Toc415573243)

[5.1.2 Feature Words 5-1](#_Toc415573244)

[5.2 Crawler Module 5-1](#_Toc415573245)

[5.3 Preprocessing Module 5-2](#_Toc415573246)

[5.3.1 Text Normalizer 5-2](#_Toc415573247)

[5.3.2 Tokenizer 5-3](#_Toc415573248)

[5.3.3 POS Tagger 5-4](#_Toc415573249)

[5.3.4 Filipino NER 5-4](#_Toc415573250)

[5.3.5 Preprocessor Manager 5-5](#_Toc415573251)

[5.4 Feature Extraction Module 5-5](#_Toc415573252)

[5.5 Category Classifier Module 5-6](#_Toc415573253)

[5.6 Rule Inductor Module 5-7](#_Toc415573254)

[5.7 Ontology Population Module 5-8](#_Toc415573255)

[5.7.1 OntologyModule 5-12](#_Toc415573256)

[5.7.2 OntologyRetriever 5-14](#_Toc415573257)

[6.0 Results and Observation 6-1](#_Toc415573258)

[6.1 Evaluation of Word Feature Set 6-1](#_Toc415573259)

[6.1.1 Methodology 6-1](#_Toc415573260)

[6.1.2 Experiments 6-1](#_Toc415573261)

[6.2 Evaluation of Classifiers 6-6](#_Toc415573262)

[6.2.1 Methodology 6-6](#_Toc415573263)

[6.2.2 Multiple Binary Classifier 6-6](#_Toc415573264)

[6.2.3 Single Classifier 6-9](#_Toc415573265)

[6.3 Evaluation of Information Extraction 6-11](#_Toc415573266)

[6.3.1 Methodology 6-11](#_Toc415573267)

[6.3.2 Caution and Advice Category 6-12](#_Toc415573268)

[6.3.3 Casualties and Damage 6-14](#_Toc415573269)

[6.3.4 Donation Category 6-18](#_Toc415573270)

[6.3.5 Call for Help Category 6-20](#_Toc415573271)

[7.0 Conclusion and Recommendation 7-21](#_Toc415573272)

[7.1 Conclusion 7-21](#_Toc415573273)

[7.2 Recommendation 7-22](#_Toc415573274)

[7.2.1 Preprocessing Module 7-22](#_Toc415573275)

[7.2.2 Category Classifier Module 7-22](#_Toc415573276)

[7.2.3 Ontology Population Module 7-22](#_Toc415573277)

[Appendix A. References A-1](#_Toc415573278)

[Appendix B. Examples of Filipino Morphemes B-1](#_Toc415573279)

[Appendix C. List of Unified Hashtags Used in the Crawler Module C-1](#_Toc415573280)

[Appendix D. List of Filipino Stop Words D-1](#_Toc415573281)

[Appendix E. List of Irrelevant Words Removed from Top Scoring Word Features E-1](#_Toc415573282)

[Appendix F. Top 30% TFIDF Word Features F-1](#_Toc415573283)

[Appendix G. Top 30% Combined Word Features of Mario and Ruby Dataset G-1](#_Toc415573284)

[Appendix H. Extraction Rules H-1](#_Toc415573285)

[Appendix I. Representation of Ontology in OWL Format I-1](#_Toc415573286)

[Appendix J. Resource Persons J-1](#_Toc415573287)

[Appendix K. Personal Vitae K-1](#_Toc415573288)

List of Tables

[Table 1‑1. Timetable of Activities (April 2014 - April 2015) 1-6](#_Toc417059523)

[Table 2‑1. Summary of Reviewed Information Extraction Systems 2-8](#_Toc417059524)

[Table 2‑2. Four Main Categories of Vital Information 2-11](#_Toc417059525)

[Table 2‑3. Tweet Categories 2-12](#_Toc417059526)

[Table 2‑4. Informative Tweet Categories 2-12](#_Toc417059527)

[Table 2‑5. Extractable Information Nugget per Informative Tweet Category 2-13](#_Toc417059528)

[Table 3‑1. Examples of official government institution 3-15](#_Toc417059529)

[Table 3‑2. Examples of disaster-related tweets with extractable information 3-15](#_Toc417059530)

[Table 3‑3. Confusion Matrix (Davis and Goadrich, 2006) 3-17](#_Toc417059531)

[Table 4‑1. Sample Input/Output for Text Normalizer 4-4](#_Toc417059532)

[Table 4‑2. Sample Input/Output Tokenizer 4-4](#_Toc417059533)

[Table 4‑3. Sample Input/Output POS Tagger 4-5](#_Toc417059534)

[Table 4‑4. Sample Input/Output Gazetteer 4-5](#_Toc417059535)

[Table 4‑5. Sample Input/Output Category Classifier Module 4-7](#_Toc417059536)

[Table 4‑6. Sample Entries of Tweets in CSV File 4-8](#_Toc417059537)

[Table 4‑7. Sample Gazetteer for Storm Names (Philippines) 4-8](#_Toc417059538)

[Table 4‑8. Sample Extracted Rules 4-9](#_Toc417059539)

[Table 4‑9. Excerpts of the List of Seed Words 4-9](#_Toc417059540)

[Table 4‑10. Excerpts of the POS Dictionary 4-9](#_Toc417059541)

[Table 5‑1. Contents of Each Dataset 5-1](#_Toc417059542)

[Table 6‑1. Listing of Word Feature Sets 6-1](#_Toc417059543)

[Table 6‑2. Mario Dataset using top n% Mario-Combined Features (10-cross fold validation) 6-1](#_Toc417059544)

[Table 6‑3. Mario Dataset using top n% Mario-Combined Features (Split Testing) 6-2](#_Toc417059545)

[Table 6‑4. Ruby Dataset using top n% Ruby-Combined Features (10 cross-fold validation) 6-2](#_Toc417059546)

[Table 6‑5. Ruby Dataset using top n% Ruby-Combined Features (Split Testing) 6-3](#_Toc417059547)

[Table 6‑6. Mario Dataset using top n% Ruby-Combined features (10 cross-fold validation) 6-3](#_Toc417059548)

[Table 6‑7. Mario Dataset using top n% Ruby-Combined features (Split Testing) 6-3](#_Toc417059549)

[Table 6‑8. Ruby Dataset using top n% Mario-Combined features (10 cross-fold validation) 6-4](#_Toc417059550)

[Table 6‑9. Ruby Dataset using top n% Mario-Combined features (Split Testing) 6-4](#_Toc417059551)

[Table 6‑10. Mario-Ruby Dataset using top n% Mario-Combined features 6-5](#_Toc417059552)

[Table 6‑11. Mario-Ruby Dataset using top n% Ruby-Combined features 6-6](#_Toc417059553)

[Table 6‑12. Mario Dataset using top n% Mario and Ruby-Combined features 6-6](#_Toc417059554)

[Table 6‑13. Ruby Dataset using top n% Mario and Ruby-Combined features 6-7](#_Toc417059555)

[Table 6‑14. Summary of Cascading Classifier Dataset Contents 6-8](#_Toc417059556)

[Table 6‑15. Mario Dataset (CA) using top n% Mario-CA features 6-8](#_Toc417059557)

[Table 6‑16. Mario Dataset (CD) using top n% Mario-CD features 6-8](#_Toc417059558)

[Table 6‑17. Mario Dataset (CH) using top n% Mario-CH features 6-9](#_Toc417059559)

[Table 6‑18. Mario Dataset (D) using top n% Mario-D features 6-9](#_Toc417059560)

[Table 6‑19. Ruby Dataset (CA) using top n% Ruby-CA Features 6-10](#_Toc417059561)

[Table 6‑20. Ruby Dataset (CD) using top n% Ruby-CD Features 6-10](#_Toc417059562)

[Table 6‑21. Ruby Dataset (CH) using top n% Ruby-CH Features 6-10](#_Toc417059563)

[Table 6‑22. Ruby Dataset (D) using top n% Ruby-D features 6-11](#_Toc417059564)

[Table 6‑23. Mario Dataset using top 30% Mario-Combined features 6-11](#_Toc417059565)

[Table 6‑24. Ruby Dataset using top 30% Mario-Combined features 6-12](#_Toc417059566)

[Table 6‑25. Mario-Ruby Dataset using top 30% Mario-Combined features 6-12](#_Toc417059567)

[Table 6‑26. Ruby Dataset Location Extraction Statistics 6-13](#_Toc417059568)

[Table 6‑27. Ruby Dataset Location Extraction Statistics (Using NER) 6-14](#_Toc417059569)

[Table 6‑28. Mario Dataset Location Extraction Statistics 6-14](#_Toc417059570)

[Table 6‑294. Mario Dataset Location Extraction Statistics (Using NER) 6-14](#_Toc417059571)

[Table 6‑30. Sample Extraction of Location in Tweets 6-14](#_Toc417059572)

[Table 6‑31. Ruby Dataset Advice Extraction Statistics 6-15](#_Toc417059573)

[Table 6‑32. Mario Dataset Advice Extraction Statistics 6-15](#_Toc417059574)

[Table 6‑33. Sample Extraction of Caution/Advice in Tweets 6-15](#_Toc417059575)

[Table 6‑34. Ruby Dataset Location Extraction Statistics 6-16](#_Toc417059576)

[Table 6‑35. Mario Dataset Location Extraction Statistics 6-16](#_Toc417059577)

[Table 6‑36. Sample Extraction of Location in Tweets 6-17](#_Toc417059578)

[Table 6‑37. Ruby Dataset Object Name Extraction Statistics 6-17](#_Toc417059579)

[Table 6‑38. Mario Dataset Object Name Extraction Statistics 6-17](#_Toc417059580)

[Table 6‑39. Sample Extraction of Object Name in Tweets 6-18](#_Toc417059581)

[Table 6‑40. Ruby Dataset Object Details Extraction Statistics 6-18](#_Toc417059582)

[Table 6‑41. Mario Dataset Object Details Extraction Statistics 6-18](#_Toc417059583)

[Table 6‑42. Sample Extraction of Object Details in Tweets 6-19](#_Toc417059584)

[Table 6‑43. Ruby Dataset Victim Name Extraction Statistics 6-19](#_Toc417059585)

[Table 6‑44. Mario Dataset Victim Name Extraction Statistics 6-19](#_Toc417059586)

[Table 6‑45. Sample Extraction of Victim Names in Tweets 6-20](#_Toc417059587)

[Table 6‑46. Ruby Dataset Location Extraction Statistics 6-20](#_Toc417059588)

[Table 6‑47. Mario Dataset Location Extraction Statistics 6-20](#_Toc417059589)

[Table 6‑48. Sample Extraction of Location in Tweets 6-20](#_Toc417059590)

[Table 6‑49. Ruby Dataset Resource Name Extraction Statistics 6-21](#_Toc417059591)

[Table 6‑50. Mario Dataset Resource Name Extraction Statistics 6-21](#_Toc417059592)

[Table 6‑51. Ruby Dataset Resource Details Extraction Statistics 6-21](#_Toc417059593)

[Table 6‑52. Mario Dataset Resource Details Extraction Statistics 6-21](#_Toc417059594)

[Table 6‑53. Ruby Dataset Victim Name Extraction Statistics 6-22](#_Toc417059595)

[Table 6‑54. Mario Dataset Victim Name Extraction Statistics 6-22](#_Toc417059596)

List of Figures

[Figure 1‑1. Research Methodology Phases 1-4](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573425)

[Figure 2‑1. Poibeau's General Architecture 2-5](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573426)

[Figure 3‑1. Structure of an Information Extraction System 3-2](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573427)

[Figure 3‑2. StaLe Lemmatization Process 3-4](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573428)

[Figure 3‑3. Architecture of LearningPinocchio 3-7](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573429)

[Figure 3‑4. Rule-Induction Step 3-8](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573430)

[Figure 3‑5. Algorithm for Choosing the Best Rules 3-9](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573431)

[Figure 3‑6. Information Extraction Process of LearningPinocchio 3-10](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573432)

[Figure 3‑7. Architecture of IE2 Adaptive Information Extraction System 3-10](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573433)

[Figure 3‑8. SOMIDIA Architecture 3-11](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573434)

[Figure 3‑9. Process of Semi-Automatic Ontology Population 3-14](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573435)

[Figure 4‑1. FILIET Architectural Design 4-3](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573436)

[Figure 4‑2. FILIET Ontology 4-10](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573437)

[Figure 4‑3. Tweet Retrieval Screenshot 4-10](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573438)

[Figure 4‑4. Information Extraction Screenshot 4-11](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573439)

[Figure 4‑5. Ontology Population Screenshot 4-11](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573440)

[Figure 4‑6. Ontology Access Screenshot 4-12](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573441)

[Figure 5‑1. Process Diagram for Storing Information to the Ontology 5-12](file:///D:\Dropbox\Awesome%20Nerds\Thesis\_Main%20Document\FILIET%20v3.docx#_Toc415573442)

List of Code Listings

Code Listing 3‑1. Ontology Model Creation 3-18

Code Listing 3‑2. Ontology Class Creation 3-18

Code Listing 3‑3. Ontology Object Property Creation 3-19

Code Listing 3‑4. Ontology Instance Creation 3-19

Code Listing 3‑5. Tweet Tokenization 3-19

Code Listing 3‑6. Text Normalization with NormAPI 3-19

Code Listing 5‑1. Crawl for tweets with the specified keywords 5-2

Code Listing 5‑2. Store the Twitter status to the database 5-2

Code Listing 5‑3. Using NormAPI 5-2

Code Listing 5‑4. NormAPI Approach 5-3

Code Listing 5‑5. FILIET Normalizer Execution 5-3

Code Listing 5‑6. ArkNLP Tokenizer Approach 5-3

Code Listing 5‑7. OpenNLP Tokenizer Approach 5-3

Code Listing 5‑8. Tokenizer Implementations in FILIET 5-3

Code Listing 5‑9. OpenNLP Tokenizer Approach 5-4

Code Listing 5‑10. FILIET Tokenizer Execution 5-4

Code Listing 5‑11. POS Dictionary Look-up Approach 5-4

Code Listing 5‑12. FILIET POS Tagger Execution 5-4

Code Listing 5‑13. SOMIDIA’s NER Approach 5-5

Code Listing 5‑14. FILIET Filipino NER Execution 5-5

Code Listing 5‑15. FILIET Preprocessor Manager Initialization 5-5

Code Listing 5‑16. FILIET Feature Extraction Initialization 5-6

Code Listing 5‑17. FILIET Feature Extraction Execution 5-6

Code Listing 5‑18. FILIET Classifier Implementations 5-6

Code Listing 5‑19. Classifier Builder Initialization 5-7

Code Listing 5‑20. FILIET Classifier Execution 5-7

Code Listing 5‑21. Rule Inductor Initialization 5-7

Code Listing 5‑22. Sample Extraction Rules for CA Category 5-7

Code Listing 5‑23. Rule Inductor Initialization 5-8

Code Listing 5‑24. addCautionAndAdviceReport Initialization 5-8

Code Listing 5‑25. Prerequisite Class Instantiations 5-9

Code Listing 5‑26. Ontology Individual Creations 5-9

Code Listing 5‑27. Class-Individual Assertions 5-9

Code Listing 5‑28. Data Property Assertion 5-10

Code Listing 5‑29. Object Property Assertion 5-10

Code Listing 5‑30. Asserting Individual Instances as Class Instances 5-11

Code Listing 5‑31. Adding Data Properties to Instantiated Individuals 5-11

Code Listing 5‑32. OntologyModule Initialization 5-11

Code Listing 5‑33. OntologyModule Initialization 5-12

Code Listing 5‑34. Sample Initialization of Categorized Tweet Information Instance 5-13

Code Listing 5‑35. Storing Information in the Ontology 5-13

Code Listing 5‑36. OntologyRetriever Initialization 5-14

Code Listing 5‑37. RetrievedTweet Class Structure 5-14

Code Listing 5‑38. OntologyRetriever Execution 5-14

# Research Description

This chapter introduces the research undertaken in the field of Text Classification (TC) and Information Extraction (IE) in Natural Language Processing (NLP) for disaster management. This chapter is divided into four sections. The first section talks about the motivations and the problem that needs to be addressed. The second section discusses the objectives of the research. The third section details the scope and limitations of the study. Lastly, the fourth section tackles the significance of the research and its benefits to 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, properties, and livelihood across the world. Just between 2000 and 2012, about 2 million people have died during disasters and an estimated US$ 1.7 trillion in damages have been recorded. 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 and platforms that aim to facilitate interaction, collaboration, and sharing of content. Social media can be accessed by computers or by smart phones. In a study of Universal McCann and an analysis of 24/7 Wall St., LLC about social media, the Philippines received 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 play a vital role in disaster management. For example, after the Haiti earthquake in 2010, numerous posts and photos were published in various social media sites. Just 48 hours later, the Red Cross has raised US$8 million. Social media have also enabled the generation of community crisis maps and interagency maps. They are maps that work as intermediaries 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-1), that quickly sorts through online data, from tweets to uploaded photos, and then displays 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). To illustrate further how social media are significantly regarded, in a study commissioned by the American Red Cross[[2]](#footnote-2), it was revealed that 74% of the respondents expect response agencies to answer social media calls for help within an hour.

Twitter is a social media microblogging platform where users can post statuses in real-time. In times of disaster, Twitter is used to share information regarding the disaster including updates on response efforts. As part of the Philippine disaster management for natural calamities, the government has released an official newsletter detailing the official social media accounts and hashtags[[3]](#footnote-3). Filipino Twitter users tend to post tweets about requests for help and prayer. Other tweets pertain to traffic and weather updates, related observations, and class suspensions. While some users prefer to post messages in English, a large number of users also communicate with their native language when tweeting during disasters (Lee et al., 2013).

Knowing that various emergency response organizations aim to, as much as possible, attend to all requests for help, it would be very important and beneficial to have a system that is capable of extracting relevant disaster relief operation information from the contents that are posted by Filipino netizens in Twitter. Furthermore, it would be very helpful to have an information extraction system that is able to mine relevant information from the language that is dominant in the disaster-stricken areas, which, in the case of the Philippines, is the Filipino language and, at the same time, support the way how content is posted in Twitter like having certain formats (having #tags), writing style (TXTSPK and code-switched styles), etc. In general, having this system can open up opportunities for improving how disaster relief operations are planned and conducted in the Philippines, and eventually, can help save lives.

## 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 relief effort information from disaster-related tweets.

### Specific Objectives

The following are the specific objectives of the research:

1. To review different information extraction systems;
2. To identify the different types of disaster-related tweets and the relevant information needed in relief operations;
3. To review different NLP techniques that are applicable in pre-processing Twitter data;
4. To analyze different approaches used in implementing an information extraction system;
5. To evaluate existing tools and resources that could be incorporated in the information extraction components of the system;
6. To determine the metrics for assessing the performance or effectiveness of the information extraction system.

## Scope and Limitations of the Research

The research is about the design of an information extraction system for the Filipino language. Review of various information extraction systems to know the different approaches to implementation was covered. Different existing domain-independent, domain-dependent information extraction systems were also analyzed to understand their components, architectures, and implementation. Additionally, this study examined information extraction for MRL to grasp the techniques used to extract from MRL given that the Filipino language is considered an MRL.

For the system to extract relevant information, the research determined which information details are deemed relevant in times of disaster, especially in relief operations. The research also identified the different types of disaster-related Tweets to support the task of discerning relevant information from the given tweets. Other researches on the use of Twitter in disaster management were also evaluated to aid in the formulation of ontologies of the information extraction system developed in this study.

In terms of system performance, the research looked into different natural language processing techniques used for data preprocessing before feeding them into the information extraction system. Examples of the NLP techniques are text classification and text normalization. Text classification is the process of automatically assigning a text or document into a predefined category based on its 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 Twitter are noisy. Most of the text has no structure, incorrectly spelled words, and invented terms.

Different information extraction techniques were also examined. Some of these 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). Co-reference analysis is the resolving of references for the pronouns (Grishman, 1997).

Existing NLP tools for building information extraction systems were also reviewed. Examples of these 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).

Lastly, metrics were determined to measure system 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 Twitter being one of the most commonly used social media platforms in the country, a new level of information dissemination has been established. With an information extraction system that is built for the Filipino language and at the same time for supporting texts that are found in Twitter, respective stakeholders can explore more possibilities and opportunities with regard to effectively utilizing such information from the web and use them for disaster management purposes.

From a disaster management standpoint, there are a number of advantages to having an information extraction system that is specifically made to work with Twitter texts that are written in the Filipino language.

First, respective stakeholders can collect disaster-related information in a way that is less strict because with an information extraction system built for the two languages, 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 can extract information from how Filipinos speak and communicate through the different social media platforms available, and to be specific, in Twitter.

Second, with an information extraction system, respective stakeholders can easily make use of the information that is written in the format of the different variations of the languages like the ‘TXTSPK’ and ‘Code Switching’. With a custom-built information extraction algorithm, the information extraction system will be able to increase the probability of accurately and precisely extracting relevant information.

Third, the information that can be extracted from Twitter can be further utilized to help in disaster relief efforts. With a system that can further categorize tweets automatically can help in extracting more straightforward and meaningful information about the current state of disasters. Certain types of tweets can indicate a specific set of relevant information that can be extracted. Take, for instance, Disaster Information Tweets. Information that can be extracted from this kind of tweets can include, but not limited to, the type of disaster, location of disaster and etc. Or take, for instance, Casualty Report Tweets. Information like the number of casualties or the names of missing people can be extracted from this type of tweets.

Lastly, with an information extraction system that can organize the extracted relevant information, respective stakeholders can now expedite the process of conducting relief operations since they can be presented with information that has already been processed to be easily read and understood by the normal people. With this information extraction system, the process of consolidating necessary relevant disaster-related information can be more intuitive and faster.

## Research Methodology

Scrum-based methodology, an iterative software development life cycle, was applied in the course of this research.



Figure ‑. Research Methodology Phases

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.

## Investigation and Research Analysis

This phase involved the study and understanding of the fundamental knowledge of the concepts, algorithms, techniques, and tools that can be used to implement the system as well as identifying the modules and requirements of the system to be developed. The key activity involved in this phase is literature reviews of related works. From the review process, the techniques, tools, and metrics were then compared to verify which can be adapted to the system.

### System Design

With the information takeaways from the earlier phase, system design commences. In this phase, appropriate architectures, algorithms, information extraction techniques, and other necessary tools are identified so that they can be effectively utilized in system creation. In addition, in this phase, necessary modules for the system are also identified based on the different processes and features that will be built into the system. The designs of the User Interfaces and the basic architecture for the databases are also covered in this stage. Finally, this phase also addressed data source identification for use and processing by the system. Once the data sources have been identified, data collection immediately commences.

### Sprints

There is a two-week timeframe for each sprint 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 to conducting 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 are discussed here. Included in these meetings is the assignment and division of the tasks among the members of the team. The evaluation of tasks from the previous sprint is also 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 are conducted daily. The purpose is to update each member about what has or has not been accomplished yet in the assigned tasks. This ensures that there is daily progress and if there are issues that hinder members from accomplishing their assigned tasks so that they can be assisted.

### System Development

From the design phase, system development follows. Data collection will also be done in this phase. Each team member is assigned to modules. The development of the system follows 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 to assess the progress of the thesis and to plan the succeeding tasks.

### System Integration and Testing

All the modules that have been developed are integrated into one system. This phase is also about 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 integration, the system is then subjected to another round of tests to check again for any faulty integration and bugs.

### System Evaluation

This phase is system performance evaluation following the metrics selected and reviewed. The following metrics have been identified so far: Precision, Recall, and F-measure results of the information extracted by the system. A number of tests of information extracted manually and those from the training set are undertaken to compare and validate results. The metrics can also be modified as needed depending on additional tests and findings in the future.

### Documentation

Documentation of activities, methodologies, and of the system developed is important for monitoring and modification or improvement purposes. It will also be used for further reference, in case there is a need to validate or cross-reference any future work.

### 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 ‑. 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 applications that are related or similar to this research.

## 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 variation of the slot-filling problem, specifically how to find the best unbroken fragment of text to fill a given slot in the answer template. A definite template 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 behavior of the algorithms to be developed: to isolate each field, learn the problem and focus on fields that are 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 the ML of a feature set is needed to help adapt to domains containing novel structures because 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 conducted experiments to gauge the performance of the four components.

In summary, the researchers found 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, training effective extractors with very simple document representations is feasible.

##### 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 casualties, missing or wounded), (4) information related to infrastructure (number of affected hectares, economic losses. It is able to extract information on natural disasters 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 stage is the document feature extraction. To limit the dimension, they used information gain technique. After conversion to a Boolean vector, classification follows. They used Support Vector Machine (SVM), Naïve Bayes (NB), C4.5, and k-Nearest Neighbors (kNN). After classification, text that might contain relevant information is selected. This stage is the candidate text selection. This process uses grammar to select the text and a dictionary of names and number to treat grammar exceptions. Then the output becomes candidates of relevant information. The system will then select which of the information will be used. This system uses the same algorithms in the text classification stage. They used different classifiers for different outputs.

This architecture boasts of 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 needs to change the training corpus.

In this work, 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 obtained an F-measure from 72% to 88% on the 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 the name and quantities was SVM, while it was kNN for dates. The overall system obtained an average of 72% on the F-Measure.

##### 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 its 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 any uncovered positive examples remain 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 the 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.

## 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 specific criteria before it is fed into the system. First, input must be news articles related to real estate advertisements. Second, only one advertisement is allowed from each input data file. Lastly, it must be stripped off of all its HTML tags. After the data have met all the criteria, it will now go to data normalization. This process helps reduce ambiguity and assists in the annotation process. The necessary punctuation at the end of each sentence is also added. Second, it merges multiple paragraphs into one. Third, punctuations are normalized, redundant spaces are removed, and the first character after each punctuation is capitalized. Lastly, the telephone, price, area, and zone details are normalized to a common pattern. Upon completion, the data will now be manually annotated using Callisto, an annotation software.

After annotation, data are 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 annotation contains the part-of-speech, the word; it also checks if the first letter is capitalized, and has other features (kind and nation). This will be used to create the Java Annotation Pattern Engine (JAPE) rules. The Split annotation contains the delimiter. The next process is through the Gazetteer. Gazetteers are dictionaries that are created during system development and they include potential named entities (person, location) or categories, phrases used 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 this process, the text or data will now be passed on to the JAPE transducer. The JAPE transducer is responsible for extracting the information. It uses JAPE rules to recognize the entities that will need to be extracted. The annotated documents are the output.

The system has been tested using a lenient criterion and a strict criterion. An entity that is recognized correctly when the type is correct but the span overlaps in the annotated corpus is called the lenient criterion. 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 criterion. On the lenient criterion on test data, it registered 96% on the F-measure. While on the strict criterion, it registered 91% on the F-measure. The problem is on the data. The writing styles of the people are very diverse. The system has a problem in recognizing some of the entities like the zone entity because some of the zone entities are very long and do not observe 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, and 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 regard to how the researchers pre-process their chosen input data, they interpret the HTML structure of documents and analyze some contextual facts to transform the unstructured web pages into structured forms. The approach applied by the researchers is quite robust in the variability of the DOM (for the web pages); it is also upgradeable and keeps data up-to-date. The evaluation metrics showed high efficiency of information access to the generated data. In 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-processes 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 URLs 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 analyzer’ 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 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 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. 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. By doing a depth- first traversal of the expressive DOM tree, the desired subtree can be isolated based on the headings of the data record like the following: “Herausgeber” (publisher), “Betreiber” (operator), “Anbieter” (provider), etc. The system was programmed to disregard domain-name irrelevant information; thus, the analyser will work further 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 was composed of approximately 150 business web pages. The only encountered problem by the system was when value for certain attributes were erroneously represented like text in phone numbers, among others.

## 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 information that is deemed important by the user. Its performance is based 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 are 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 such as 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 are 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 2‑1 illustrates the general architecture proposed by Poibeau.

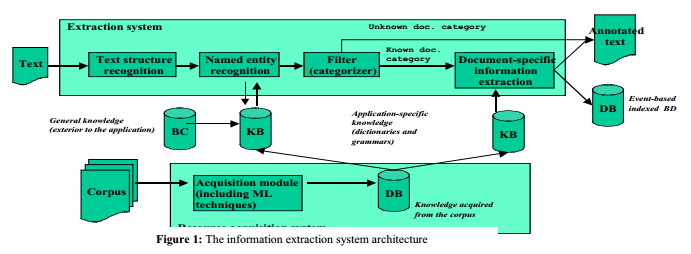


Figure ‑. Poibeau's General Architecture

## Ontology-Based Information Extraction Systems

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

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

As most information extraction systems are based on the English language, it poses a problem for other languages in terms of limited tools available. To address this problem, the system maps the extracted entities to the ontology.

This system extracts names of persons, locations, and organizations from French newspaper articles. 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 consider ontological similarity. 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., 2013)

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 components 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 use OpenNLP to tokenize the document, then it will go to a sentence splitter. The sentence splitter accepts a list of tokens and an 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 uses 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 pass the document to the POS tagger. Otherwise, it would be passed on 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 tokens with its corresponding POS tags. After the POS tagger, it will go through a ‘chunker’. The chunker groups the tokens into 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 pass through the English NER. The NER only focuses on proper nouns. It uses LingPipe because of its flexibility. LingPipe’s NER uses three types of approaches, dictionary-based, rule-based, and statistic-based approaches. After the NER, it will go through co-reference resolution. The co-reference resolution will find the noun counterpart of the pronouns. It uses the 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 the information extraction phase. It uses JAPE rules to extract the information, and 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 a new extraction pattern. The process will then be repeated. 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 is no existing Filipino NER tool. It uses dictionary-based and rule-based approaches for their NER. After tagging, 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 systems. The table lists 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 ‑. 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 |

## 

## Disaster Management – Relief Operations

This part discusses more about Relief Operations and the different information that are essential to this aspect of Disaster Management.

##### Humanitarian Knowledge Management (King, 2005)

This paper discusses the complexities and numerous challenges that many humanitarian organizations face whenever complex international humanitarian emergencies occur and how certain critical information in relation to disaster management activities, such as humanitarian assistance or relief operations can be utilized to help facilitate needed actions. King mentioned that the problem lies on the management of the data needed about these emergencies. In his paper, King stated that data management includes identifying, presenting, and disseminating critical information about the situation although such critical information, in itself, present a serious problem that could greatly affect data management. The problem lies in how this critical information is gathered: what information should be gathered and where should these be taken from? Upon efficiently identifying this in the early stages of these kinds of activities, as King mentioned, humanitarian organizations can more effectively make contingency plans and respond to natural disasters and complex emergencies and at the same time, potentially save a significant number of lives.

In the paper, a specific section was made to discuss what information are essential and crucial to different humanitarian organizations whenever they would conduct relief operations as a response to international complex emergencies like natural disasters and etc. According to King, humanitarian organizations like NGOs, UN agencies, local and national government, etc. need two specific types of information: (1) background and (2) situational information. Furthermore, information that is not within these types is more pertinent, relevant and critical to various specific personnel that are also within the said organizations. To support this claim, King gave an example through a scenario. He mentioned, “*policy makers want “big picture snapshot” analysis in order to understand the issues, to make decisions on providing assistance, and to be alerted to problems and obstacles*…*field personnel and project and desk officers in aid organizations, on the other hand, need more detailed operational and programmatic information in order to plan and implement humanitarian assistance and reconstruction programs.*”

With all of these, King listed down four main categories for the different vital information that is needed by organizations whenever they would conduct relief operations. Table 2‑2 lists the four categories as well as their description and guide question that helps in determining which category the information belongs to.

## Twitter and Disaster

This part discusses the uses of Twitter in times of disaster, the information that are useful during disasters, the information that can be extracted from disaster-related tweets and lastly, systems that make use of Twitter for disaster management procedures.

##### Extracting Relevant Information Nuggets from Disaster-Related Messages in Social Media (Imran et al., 2013)

This paper focuses on the extraction of relevant information from disaster-related tweets. The data set the authors worked with are Twitter data during hurricane Joplin on May 22, 2011 with #joplin. Their approach includes text classification and information extraction.

First, the tweets were classified into their respective categories. Table 2‑3 lists the categories for the tweets. After filtering the tweets, only those of the Informative category were used. The informative tweets were further categorized into the information types they contained, which is listed in Table 2‑4. The basis for the categories was from the ontology by Vieweg et al. (2010).

Table ‑. Four Main Categories of Vital Information

|  |  |  |
| --- | --- | --- |
| **Category** | **Description** | **Guide Questions** |
| Situational Awareness | Information about the latest situation on the ground and information about the conditions, needs, and locations of affected populations | * What is the latest/current humanitarian situation in the country? * What are the most recent severity indicators? (Death tolls, mortality rates, malnutrition rates, economic impact, infrastructure damage, etc.) * Who are the affected populations (refugees, IDPs, children and other vulnerable groups, resident populations, etc.); how many are there, and where are they located? * What are the conditions and humanitarian needs of the affected populations? * What is the assessment of damage to infrastructure? (Transport, buildings, housing, communications, etc.) * What is the latest/current security situation in the affected areas of the country? |
| Operational / Programmatic | Information necessary to plan and implement humanitarian assistance programs | * Where are and what are the conditions of the logistical access routes for delivering humanitarian assistance? * Who’s doing what where? What humanitarian organizations are working in the country, what are their programs, what are their capacities, and where are they working? * How is the host country/government responding and can it provide more? * What are the programmatic/financial needs of the humanitarian organizations? * What and how much are being provided to the humanitarian response organizations and who are the donors? |
| Background | Information about the unique history, geography, population, political and economic structure, infrastructure and culture of the country to be able to compare the emergency situation and conditions to previous normal conditions; and lastly | * What are the country’s population (national, province/state, city/town) and composition (ethnicity, religion, age cohorts, urban/rural, political, etc.)? * What is the geography of the country? * What are the country’s past disasters and natural hazards? * What are the most recent annual baseline health indicators for the population? (crude mortality rate, infant/child mortality rates, HIV adult prevalence, malnutrition, etc.) * What are the annual economic indicators? (GDP, GNP, agricultural/food production, staple food prices, etc.) |
| Analysis | Humanitarian information needs to be interpreted in context and related to other thematic information. Analysis can include evaluations of issues and responses, projections about the future, and recommendations for policies and actions | * What are the causes and contributing factors of the emergency? * What are the constraints to providing humanitarian assistance? (Insecurity, inaccessibility, government, interference, etc.) * How effective are humanitarian assistance programs and responses? * What are the future impacts of the emergency? * What are the options and recommendations for action? |

## 

Table ‑. Tweet Categories

|  |  |
| --- | --- |
| **Category** | **Description** |
| Personal Only | If a message is only of interest to its author and his/her immediate circle of family/friends and does not convey any useful information to other people who do not know the author. |
| Informative  (Direct) | If the message is of interest to other people beyond the author's immediate circle, and seems to be written by a person who is a direct eyewitness of what is taking place. |
| Informative  (Indirect) | If the message is of interest to other people beyond the author's immediate circle, and seems to have been seen/heard by the person on the radio, TV, newspaper, or other source. The message must specify the source. |
| Informative  (Direct or Indirect) | If the message is of interest to other people beyond the author's immediate circle, but there is not enough information to tell if it is a direct report or a repetition of something from another source. |
| Others | If the message is not in English, or if it cannot be classified. |

To classify the tweets into the categories mentioned, Naïve Bayesian classifiers were trained and implemented using Weka. Their features include binary features (if the tweet contains the ‘@’ symbol, hashtags, emoticons, links or URLs, and numbers), scalar features (the length of the tweet), and text features (unigrams, bigrams, POS tags, POS tag-bigrams, and VerbNet classes).

For each informative tweet category, various types of information, referred to as information nuggets, were extracted. Table 2‑5 shows the extractable information nugget per informative tweet category as well as that category’s type subsets. The location references, time references, and number of casualties were extracted using the Stanford Named Entity Recognizer. All the Twitter Handlers (i.e. all words starting with the ‘@’ symbol and URLs) were extracted from the tweet for the sources. Caution/Advice and Damaged Object were extracted using the Stanford Part of Speech Tagger and WordNet. For the intention of the tweet, another classifier was trained to determine if the tweet is a donation effort or a request for help. Lastly, the type information nugget pertains to the Type Subset column. For each informative tweet category, another classifier was trained to classify the category into its corresponding subset.

Table ‑. Informative Tweet Categories

| **Category** | **Description** |
| --- | --- |
| Caution and advice | If a message conveys/reports information about some warning or a piece of advice about a possible hazard of an incident.  Example: “Alerto sa Mayon Volcano, itinaas ng Phivolcs sa level 2” |
| Casualties and damage | If a message reports the information about casualties or damage done by an incident.  Example: “Bush fires destroy 50 hectares in Baler, Aurora – NDRRMC http://t.co/Oc70OMeung49” |
| Donations of money, goods or services | If a message speaks about money raised, donation offers, goods/services offered or asked by the victims of an incident  Example: “Repacking of Mineral waters! (@ Dano Residenza) http://t.co/iHUn4XA7jb” |
| People missing, found, or seen | If a message reports about the missing or found person affected by an incident or a celebrity seen visiting ground zero  Example: “@philredcross missing joahnna nicole juliana ortiz sn isidro sulat eastern samar maytigbao church evacuation http://t.co/PGLnSEOtmY” |
| Information source | If a message conveys/contains some information sources like photo, footage, video, or mentions other sources like TV, radio related to an incident.  Example: “VIDEO: Alert level 2, itinaas sa Mayon Volcano http://t.co/g6U5AziDFt” |

Table ‑. Extractable Information Nugget per Informative Tweet Category

|  |  |  |
| --- | --- | --- |
| **Informative Tweet Category** | **Information Nugget** | **Type Subsets** |
| Caution and advice | Location references  Time references  Caution/Advice  Source  Type | Warning issued or lifted  Siren heard  Shelter open or available  Disaster sighting or touchdown |
| Casualties and damage | Location references  Time references  Number of Casualties  Damaged Object  Source  Type | Infrastructure  Death  Injury  Unspecified  No Damage  Both Infrastructure and People |
| Donations of money, goods or services | Location references  Time references  Intention of Tweet  Source  Type | Money  Blood  Voluntary Work  Food  Equipment  Shelter  Discounts  Other |
| Information source | Location references  Time references  Source  Type | Photo  Video  Website  TV Channel  Radio Station  Unspecified |

##### **Practical Extraction of Disaster-Relevant Information from Social Media (Imran et al.,** 2013)

Based on their previous paper Extracting Relevant Information Nuggets from Disaster-Related Messages in Social Media, after classifying the tweets into the informative tweet category, they extracted the information by employing a different approach. This time, they used two datasets: (1) tweets during hurricane Joplin on May 22, 2010 with #joplin and (2) tweets during hurricane Sandy on October 29, 2012 with #sandy #nyc.

To detect class-relevant information, they treated it as a sequence-labeling task. For each token in the tweet, they labeled it as either part of the relevant information or not. The (+) label indicates that the token is part of the relevant information while the (-) label indicates that it is not. After labeling, they applied Conditional Random Fields (CRF) to extract the information. A tool they also used in this paper is ArkNLP, a Twitter-specific POS tagger.

##### Safety Information Mining - What can NLP do in a disaster (Neubig et al., 2011)

In the article presented by Neubig and his team of researchers, they described the efforts of researchers in the field of Natural Language Processing in creating an information extraction system that aided in the relief operations during the 2011 East Japan Earthquake. The system that was described was primarily built to ease the mining of information regarding the safety of those affected by the earthquake from one of the most prevalent information source during that time, that is, Twitter. The system included subsystems that work for the following NLP and IE techniques like word segmentation, named entity recognition, and tweet classification.

The development cycle of the IE system has two phases: (1) resource-building phase and the (2) actual IE system development phase. To begin the development of the information extraction system, the researchers first started out by making the prerequisite resources for the system (or the resource-building phase). The researchers first focused on developing the different Language Resources and Tweet Corpus of the system. These language resources included dictionaries (used to improve the performance of the different text analyzers and classifiers in the system) and a labeled corpus of tweets (this contains safety information about the disaster and was used for the extraction from unlabeled tweets).

For the creation of the dictionaries, the researchers made use of the “Balanced Corpus of Contemporary Written Japanese” and the “UniDic dictionary” for general domain languages while the “Mozc Japanese Input Method Dictionary” and other publicly available resources like the last names specific to northeast Japan and the database of postal code were used for the domain-specific language. An additional list containing station names and locations, landmarks, etc. were made to aid in the extraction process.

For the creation of the Tweet corpus, the researchers collected tweets that contain the word ‘earthquake’, and those that contains the following hashtags: #anpi (safety information), #hinan (evacuation), #j\_j\_helpme (help request) and #save\_<location>. To complete the corpus, the researchers tried to recognize the topic of the tweet (tweet classification) and the people mentioned in the tweet (named-entity recognition). To do so, the researchers defined nine classifications for the labels/topic of the tweets and are (1) I - Himself/Herself is alive; (2) L - Alive; (3) P - Passed away; (4) M - Missing; (5) H - Help request; (6) S - Information request; (7) O - Not safety information; (8) R - External link; and lastly, (9) U - Unknown.

After developing the prerequisite resources, the researchers proceeded with the actual development of the information extraction system. According to Neubig et al., the first step in IE for the Japanese language is Morphological Analysis. The MA is responsible for the tokenization and POS tagging of the tweets and for this, they made use of an open-source tool called KyTea. To accommodate the proper named-entity recognition in the Japanese language, the researchers trained the POS tagging model and replaced all proper nouns with subcategory tags (e.g. “first name”, “last name”, “place name”, and etc.) together with the introduction of a Conversational & News Text Corpus (containing a large list of Japanese first and last names). However, even though the POS tagging has been polished, the NER still failed to detect named entities that are grouped (NER still works on a word-by-word basis) that’s why the researchers made a simple rule-based system to accommodate the grouping of the Japanese named entities.

With all these, the researchers finally combined the two developed systems (the language resources and the MA system) to make the final information extraction system. The combination of the language resources with the MA system tends to increase the performance (accuracy) of the developed information extraction system by being able to accommodate the variations in the different styles in the different datasets that were used in this research.

# 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 freely available in the internet. The problem is that people could not process these information easily because of the huge volume. It becomes more difficult as the information are 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 the 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).

An information extraction system is divided into two parts, local text analysis and discourse analysis. 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. Lexical analysis is responsible for splitting up the text into tokens. After splitting the text, it looks up a dictionary to fill out the part of speech and features of each token. After 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 such as company names to easier identification. 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 systems do 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 outputs of the scenario pattern matching are 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. Discourse analysis is the combination of all the information extracted during the local text analysis, and the formatting of the information. Under the discourse analysis are co-reference analysis and inference. Co-reference analysis attempts 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 co-reference analysis, it will undergo 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).

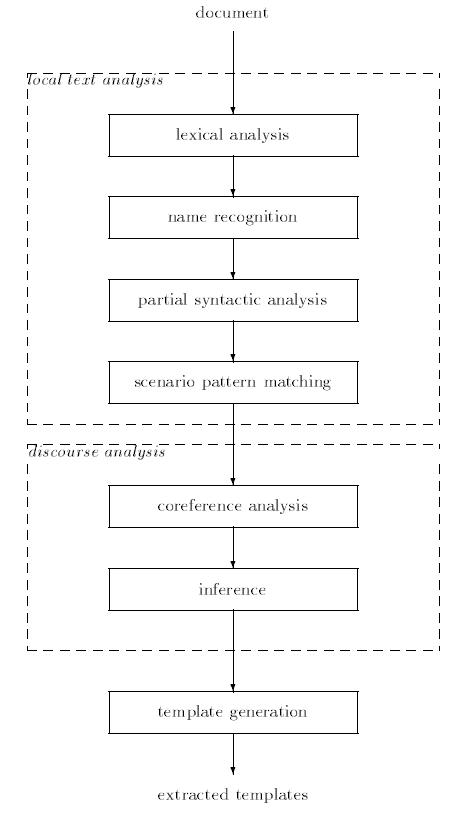


Figure ‑. Structure of an Information Extraction System

### Information Extraction Modules

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

#### Tokenizer

Tokenizer is the module that segments a given text into 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 encounters several difficulties that need to be addressed such as tokenizing, email addresses, and uniform resource locators (URLs). Tokenizers today can identify that "15MB" is interpreted as "15 megabytes" if even there is no space between '15' and 'MB', and words with punctuation marks are also read correctly if tokenized. However, 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 invented tokenizer also includes a language-specific data that contain a precedence hierarchy for punctuation (Bradlee et. al., 2001).

#### Sentence Splitter

The sentence splitter is a cascade of finite-state transducers that 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

The presence of text speaks, slangs, and lingos is very high in SMS, social networks, and microblog sites. This presence makes it difficult for 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 is based on non-standard words that have been well studied. However, with SMS, most of the lingos, for example “b4” (before) and “bf” (boyfriend) are not formally defined by linguistics yet. These words can still evolve as time passes by and more new text speaks, slangs, and lingos might be created by the younger generation.

There are two types of approaches 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 words 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 a grammar checking tool 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 its confidence factor and then pruned according to its 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 a nonsense word resulting to a poorer outcome than a traditional dictionary-based lemmatizer.

#### Co-reference Resolution

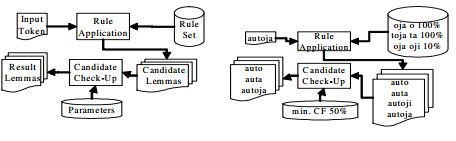


Figure ‑. StaLe Lemmatization Process

This module consists of a main module and a set of submodules. The main module is responsible for initializing the submodules, and executes them in a particular order, then combines the results generated from the submodules, and eventually performs 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 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 occurrences of “it”. The last and the main function of the co-reference 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 then chooses the best antecedent, if there is any. Then it creates the co-reference chains from the individual anaphor/antecedent (this step is performed from the main co-reference 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 Classification

Text classification or information 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 types of text may need different types of extraction techniques.

### Document Representation

Classification algorithms cannot understand texts 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 the Vector Space Model. The use of BOW may vary as it can have different representations (Shafiei et al., 2007), one of which is word representation. In word representation, each word in the document is considered as a feature. The problem with word representation is the ‘curse’ of dimensionality because text documents have a lot of unique words (Shafiei et al., 2007).

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

Character N-gram is another feature representation that could be used. Character N-gram takes n characters as a 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 when a number of features are reduced, accuracy is not sacrificed. The reduction of feature is called feature selection. There are different methods that could be used in feature selection.

Document Thresholding (DT) 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 have few occurrences are irrelevant (Wei et al., 2010).

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

where is the set of all possible categories and is the probability of a document classified into a category. This will be computed for all the words in the documents. Then, the words that did not reach the specified threshold are removed (Wei et al., 2010).

Mutual Information (MI) is the modeling 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 the work of Sriram et al., (2010), they classified short-text messages (Tweets) into news (N), events (E), opinions (O), deals (D), and private messages (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, opinion words, emphasis on words, 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 because they wanted to convey their message clearly. On the other hand, personal tweets contain usage of slangs, emotions, and shortening. These features can be used to distinguish corporate tweeters from personal tweeters. They collected 5407 English tweets, broken down into N = 2107, O = 625, D = 1100, E = 1057, and PM = 518. They also 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 the 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).

### Term Frequency-Inverse Document Frequency (TF-IDF)

Term Frequency – Inverse Document Frequency (TF-IDF) is a term weighting scheme that uses term frequency (TF) on the given document and its importance in relation to the whole collection or inverse document frequency (IDF) (Sammut, 2011; Zsu, 2009).

where,

N: number of documents in the collection

dj: number of documents containing term j

fij: frequency of term j in document i

Wij: is the weight of term j in document i

## Information Extraction Architecture

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

### Adaptive Architecture

The problem with some information extraction systems (knowledge-based systems) is that they are not portable and are highly dependent to the domain. With sources rapidly growing and becoming more diverse, it will be very hard for an information extraction system to extract as these texts are unstructured, especially given the natural language used. Another problem is that an error may propagate 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 to automatically learn rules that will extract certain information (Turmo et al., 2006).

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

LearningPinocchio is an adaptive information extraction system 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 preprocessing, information can proceed to the modules. This is where the tags will be annotated. The modules may consist of NER, text zonings, and other IE tasks. Figure 3‑3 illustrates the architecture used by LearningPinocchio.

Each module has three modes: training, testing, and production. Training mode is responsible for inducing the rules and learning how to apply IE rules in a specific scenario. The training mode accepts two inputs. First, it needs the module definition that includes a 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 the testing and production modes. The testing mode is for testing on unseen tagged corpus. This mode tells how well the module performed in a certain application. The input for the 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 statistics on the performance of the module, and details of the mistakes as well. The production mode handles receiving the tagged corpus and the XML tags to the corpus.

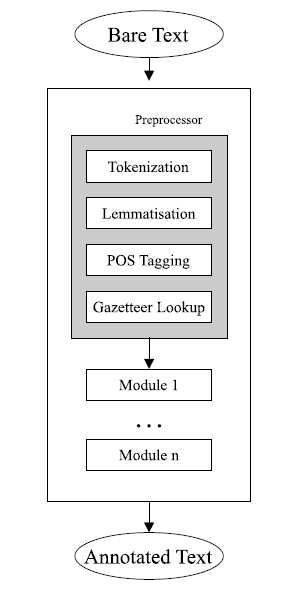


Figure ‑. Architecture of LearningPinocchio

For inducing rules, LearningPinocchio uses (LP)2 a covering algorithm especially 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 3‑4 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.

A tagging rule consists of a left-hand side, which is the pattern of conditions of a 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 have been manually tagged by a specialist. For each positive example, the algorithm first initializes rules. Then, it will generalize the rules. Lastly, it will keep the best rules. The algorithm will repeat for each positive example. Information, like word window, lexical items, lemma, lexical category, lexical case, and user-defined semantic classes could be used as a condition in the initial rules. After getting the generalizations, they 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, they are sorted according to their error rate. However, if they have the same number of matches and error rate, the one with 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.

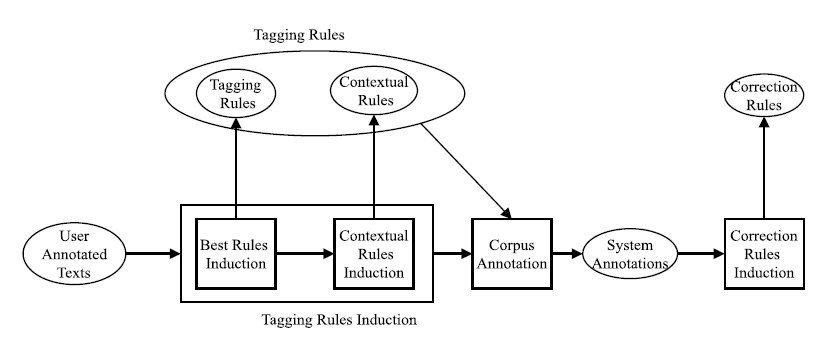


Figure ‑. Rule-Induction Step

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 3‑5 illustrates the algorithm used by LearningPinocchio 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 tag those that are missed during initial tagging, until no more tags can be placed. The third step will correct the errors. The last step will validate the tags. Figure 3‑6 shows the process of the information extraction.

LearningPinocchio was tested in two languages, English and Italian. They trained the system on a corpus and tested the induced rules on unseen texts. The system was tested in two tasks: CMU Seminar announcements and Austin job announcements.On CMU Seminar announcements, tokenization and POS tagging were performed. A gazetteer was not done for a fair comparison. The IE must be able to extract the speaker’s 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-measures, respectively. However, it had difficulty in extracting the location and speaker’s name with F-measures 77.6% and 75.1%, respectively. Overall, (LP)2 has the highest performance in All Slots with a score of 86.0%.

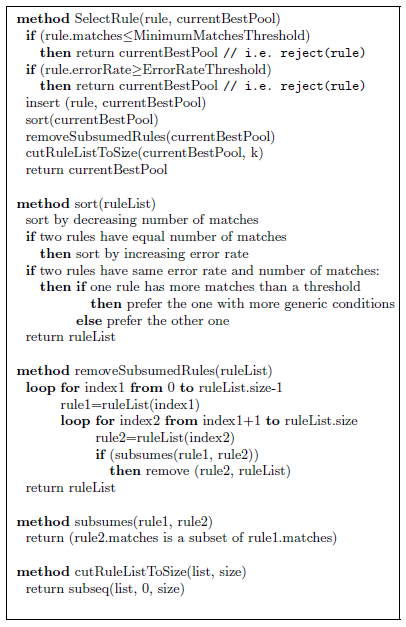


Figure ‑. Algorithm for Choosing the Best Rules

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. The same preprocessing as with the CMU Seminar announcements was done. Based on the results, (LP)2 outperformed Rapier in almost all the aspects. Rapier was able to outperform (LP)2 in salary, desired year, and desired degree. However, in the overall performance, (LP)2 has a higher performance in All Slots with a score of 84.1%.

#### 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 types of texts like unstructured, structured, and semi-structured texts. In their article, they presented the architecture they used in building the system. Aone’s IE system has six main modules in its architecture.

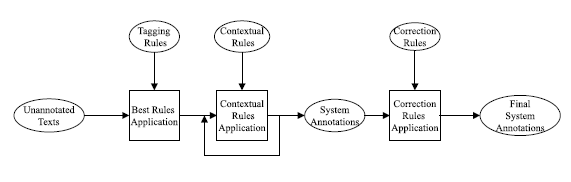


Figure ‑. Information Extraction Process of LearningPinocchio

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. 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 and 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 they work hand-in-hand. Module 3 or the PhraseTag module works by applying syntactico-semantic rules to 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. Module 5 or the Discourse Analysis Module is responsible for co-reference resolution or the merging of the previously extracted noun phrases. This module is implemented 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 co-reference. Strategy B or the Machine Learning-Based Strategy uses a decision tree that has been formed from learning a corpus tagged with co-references. 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 phrase co-references and the generation of the output in the desired format. Figure 3‑7 illustrates the architecture of the system proposed by Aone et al.

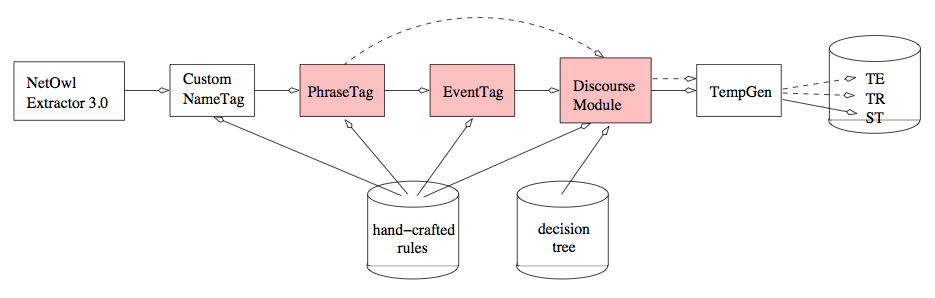


Figure ‑. Architecture of IE2 Adaptive Information Extraction System

#### SOMIDIA (Cheng et al., 2013)

SOMIDIA uses an adaptive information extraction system that extracts 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 use 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, Co-reference Resolution and English Extractor. For Filipino, they used Filipino NER and Filipino Extractor. The English information extraction process has POS Tagger, Chunker, English NER, Co-reference 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 is no existing gazetteer for Filipino. They used dictionary-based and rule-based approach in implementing the NER. Figure 3‑8 describes the architecture of SOMIDIA.



Figure ‑. SOMIDIA Architecture

For SOMIDIA to adapt to new instances, the rules must be adaptable. SOMIDIA has a pattern extractor module that is mainly responsible for extracting different patterns from a 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. 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. For each seed word 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 combinations of tokens and window setups to produce as much rules by considering 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 are able to extract more than its corresponding occurrence because these rules are too general and may have the tendency to extract irrelevant data.

## Ontology

Ontologies are sets of classes (concepts), attributes, and relationships that are used to represent domain knowledge. They are in a language (first-order logic) that can be abstracted from the data structures and implementations. Because ontologies are in the semantic level, they could be used to combine heterogeneous database, thus, making interoperability between systems possible (Gruber, 2009). Cimiano (2006) said that as the number of applications using ontologies is growing, and then every such application must now be clearly and formally defined into an ontology.

Cimiano (2006) formally defines ontology as

where,

.

In Vangelis et al. (2011), they presented four levels of classification on how an IE system exploited the ontology. The first level is the use of domain entities (including the variations), and word classes. For the first level, they can be represented by a gazetteer (flat) or ontologies (structured). By using ontologies, it can identify the text based on some constraints posed by the conceptual properties. An example system that uses the first level ontology is LearningPinocchio (Ciravegna & Lavelli, 2004). The second level uses concept hierarchies. In the second level, they focus more on taxonomic relations (consists of super/sub-ordination, “is-a” and “part-of” relationships). They could be used to generalize or specify extraction rules or check constraints. An example system is NAMIC (News Agencies Multilingual Information Categorisation) (Basili et al., 2003). The third level uses the concepts’ properties and relationships between concepts. These properties and relationships could then be used as guides for the information extraction process. An example system would be OBIE (Wang et al., 2005). The fourth level is the domain model. It combines the first three levels to be able to semantically interpret information. Domain models can merge with different structures, check consistency, make valid assumptions (for missing values), and discover implicit information. An example is BOEMIE (Bootstrapping Ontology Evolution with Multimedia Information Extraction) (Maedche, 2002). BOEMIE uses bootstrap or layered extraction process for its information extraction process. First, it extracts the entities, and then the relations. BOEMIE populates and enriches the ontology. It adds new individual entities and at the same time adds new concepts and relations.

### Ontology Design

In creating a domain-specific ontology, the following tasks must be done: selection of domain and scope, consideration of reusability, finding important terms, defining classes and class hierarchy, defining properties of classes and constraints and creation of instances of classes (Saloun & Klimanek, 2011).

There are different approaches to creating ontology: hand-making by expert, automatic, and semi-automatic. In hand-made by expert, the ontology is manually done by the experts. Its advantage is that the result will be in high quality. However, they are very expensive and time consuming. In an automatic approach, the creation of the model is done by a machine. It is fast and low cost, but the problem is that implementing it will be very difficult and will result to inaccurate models. In a semi-automatic approach, the concepts and relationship will be generated by a machine, and the expert will complete and validate it. It produces relatively good results at a short amount of time. The disadvantage is that the machine-generated concepts and relations might be inaccurate and might also cause an inconvenience (Saloun & Klimanek, 2011).

### Ontology Population

Ontology Population is the extraction and classification instances of classes and relationships of an ontology. There are three approaches for ontology population: manual, semi-automatic and automatic approaches. The manual population of ontology should be done by experts and a knowledgeable engineer. This could be costly and time consuming and the automatic approach might be inaccurate. For automatic and semi-automatic approaches, they have a common approach. They do entity name recognition, NLP techniques, and information extraction.

In Faria & Girardi (2011), the techniques they used are NLP and IE. The process has two phases: Extraction and Classification of Instances and Instance Representation. For the Extraction and Classification of Instances, all the possible relationships and class instances are generated. They consist of Corpus Analysis (Morpho-lexical analysis, Named-Entity Recognition and Co-reference), Specification of Extraction and Classification Rules, and Extraction and Classification of Instances. Then, they will manually generate a set of extraction rules based on the last task. After generating the rules, it will now use the extraction rules to look for text matching the patterns. This will now produce the instances. After the first phase, it will now go to Instance Representation. Instance Representation has two tasks: Refinement of Instances and Ontology Population. For Refinement of Instances, it will try first to see if the instance already exists in the ontology. If it does not, then it will go to . If it already exists in the ontology, it will look in to see if the instance needs to be updated. If it is, then it will be part of If not, it will be discarded. After refinement, the instance is now ready to fill the ontology. Given , it will now look in the ontology to find the class. Then if a class is found, the instance will now be instantiated. Figure 3‑9 shows the process of Faria & Girardi’s (2011) semi-automatic ontology population.

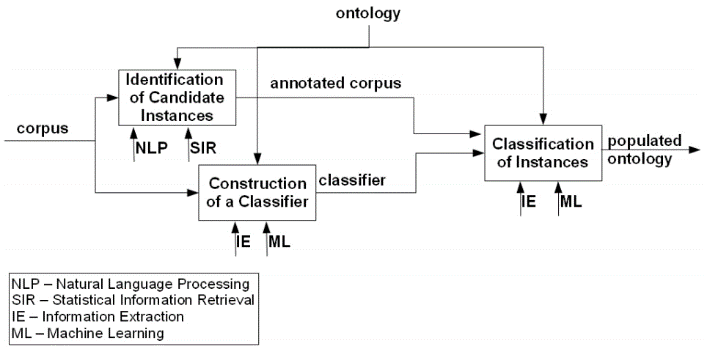


Figure ‑. Process of Semi-Automatic Ontology Population

## Twitter[[4]](#footnote-4)

Twitter is a microblogging social media platform wherein users may post messages of up to 140 characters long. Each of these posts is known as a "tweet". Mainly, a tweet is an expression of a moment or idea. Tweets 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. Thus, 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 have contributed to are 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), newscasting (Phelan et al., 2009), medicine (Hawn, 2009; Chew & Eysenbach, 2010), and business processes (Culnan et al., 2010).

### Twitter and Disasters

During disasters, Filipino Twitter users tend to retweet 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 users that use their native language when tweeting during disasters (Lee et al., 2013).

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 ‑. Examples of official government institution

|  |  |  |
| --- | --- | --- |
| **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 |

Table ‑. 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 Lagna 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) | * + magnitude * Davao Occidental |
| Earthquake | @manila\_bulletin:  BREAKING: 6.1 magnitude quake felt, east of Davao at 3:59PM. #EarthquakePH | * + 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 is correctly classified as positive or true in the gold standard.

Recall is the percentage of cases in the gold standard that is 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).

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 predicted correctly as negative while the False negative means a positive instance is predicted as negative (Davis and Goadrich, 2006). Figure 3‑3 shows its confusion matrix.

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

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

### Kappa Statistics

The common way of summarizing inter-rater agreement among observers is the kappa statistics. It allows measurement not only by chance and the observed agreement beyond chance is 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, respectively (Malpica et al., 2005).

## Tools

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

### 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. This will be used for the POS tagger and the JAPE.

#### Gazetteer

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

#### Sentence Splitter

The sentence splitter uses finite state transducers to split the text into sentences. It uses the gazetteer to check if punctuation is part of abbreviations or signals the end of the sentence. The sentence is 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 rule sets that have been trained in the Wall Street Journal corpus. However, the lexicon and rule sets 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 co-reference.

### 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 and it is also flexible as users can extend the API to customize the machine-learning algorithms (Weka 3, n.d.).

### JENA API (McBride, 2002)

JENA is a semantic web application that helps in building ontologies. It is a Java-based API that handles OWL (Web Ontology Language) and SPARQL. It also includes inference engines based on OWL and RDF. This will be used to create and manage the ontology.

Code Listing 3‑1 shows how to create an ontology.

Code Listing ‑. Ontology Model Creation

|  |
| --- |
| OntModel ontModel = ModelFactory.createOntologyModel(<model spec>); |

Code Listing 3‑2 shows how to create a class.

Code Listing ‑. Ontology Class Creation

|  |
| --- |
| Resource r = m.getResource(NS+”Paper”);  OntClass paper = r.as(OntClass.class); |

Code Listing 3‑3 shows how to create object properties.

Code Listing ‑. Ontology Object Property Creation

|  |
| --- |
| ObjectProperty hasProgramme = m.createObjectProperty( NS + "hasProgramme" );  hasProgramme.addDomain( orgEvent );  body.addRange( programme );  body.addLabel( "has programme", "en" ); |

Code Listing 3‑4 shows how to create instance/individuals.

Code Listing ‑. Ontology Instance Creation

|  |
| --- |
| OntClass c = m.createClass( NS + "SomeClass" );  Individual ind0 = m.createIndividual( NS + "ind0", c );  // second way: use a call on OntClass  Individual ind1 = c.createIndividual( NS + "ind1" ); |

### ArkNLP (Gimpel et al., 2011)

ArkNLP developed by Carnegie Mellon is a Java-based Tokenizer and POS tagger that was specifically made for Twitter. For the tokenizer, it now identifies the emoticon tokens. For the POS tagger, it can also tag slangs and emoticons. This will be used for tokenizing the tweets. Code Listing 3‑5 shows a sample code of how to use the tokenizer feature.

Code Listing ‑. Tweet Tokenization

|  |
| --- |
| List<String> tokens = Twokenize.tokenizeRawTweetText(text); |

### NormAPI (Nocon et al., 2014)

NormAPI is a text normalization API that is specifically built for the Filipino language. It currently has implementations for Dictionary Substitution Approach (DSA) and Statistical Machine Translation (SMT). The user can choose if the normalization will perform: (1) DSA only, (2) SMT only, (3) SMT after DSA, or (4) SMT before DSA. NormAPI accepts file or text as input. It also allows setting configuration files and training a new model. This will be used for the text normalization.

Code Listing ‑. Text Normalization with NormAPI

|  |
| --- |
| String normalizedText = NormAPI.normalize\_Text(shortcutText); |

# The FILIET System

This chapter presents the proposed system. It is divided into six sections. The first section discusses the system overview. The second section outlines the objectives of the system. 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 presents the resources that will be used in implementing the system.

## System Overview

Filipino Information Extraction for Twitter (FILIET) is a rule-based information extraction system for Filipino disaster related tweet. The FILIET system works with extracting information from tweets that are written in Filipino and English, along with their variations such as TXTSPK and code-switch. The system followed the methodology described below. The disaster-related tweets are loaded into the system. The system then classifies according to the following categories: (1) caution and advice, (2) casualties and damage, (3) donations, (4) call for help, and (5) others. The tweets proceed to the information extraction engine of the system wherein the system extracts relevant information from the tweets with regard to its given type of disaster. Extracted information from the given tweets vary based on the type of information the tweet contains.

## System Objectives

This section discusses the objectives of the system.

### General Objective

To develop an information extraction system that extracts relevant information from disaster-related tweets and considers the different available variations of the Filipino language.

### Specific Objectives

The following are the specific objectives of the system:

1. To preprocess the tweets;
2. To extract relevant features from the tweets;
3. To classify the tweets according to their content (i.e. caution and advice, casualties and damages, donations, and others);
4. To extract relevant information according to the type of tweet.

## System Scope and Limitations

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

The system was able to perform some preprocessing techniques onto the input tweet. These preprocessing tasks are limited to the following: (1) text normalization, which included support for input tweets that were written in the TXTSPK format; (2) text tokenization, which enabled word level analysis of the input tweet; (3) part-of-speech tagging, which enabled semantic level analysis of the input tweet; (4) named-entity recognition, which enabled proper identification of named-entities; and lastly, (5) disaster keyword tagging, which enabled proper recognition of disaster words in the input tweet. Lastly, by looking at the initial data and from the study of (Lee et al., 2013), it was observed that a high probability that Filipinos will post tweets in the Filipino language and that TXTSPK and code-switching were the variations being used.

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

Using the extracted features, the system was able to classify the input tweet based on the type of tweets. The tweet are classified into the following: caution and advice (CA), casualties and damage (CD), call for help (CH), donations (D), and others (O). This is important because each type of tweet have different extracted information. The categories are based on Extracting Relevant Information Nuggets from Disaster-Related Messages in Social Media by (Imran et al., 2013).

The data that used in the development of the system came from the Twitter Web Crawler developed by the De La Salle University - College of Computer Studies as well as from the crawler developed by the group. The system only processes data that are written in the Filipino language.

## Architectural Design

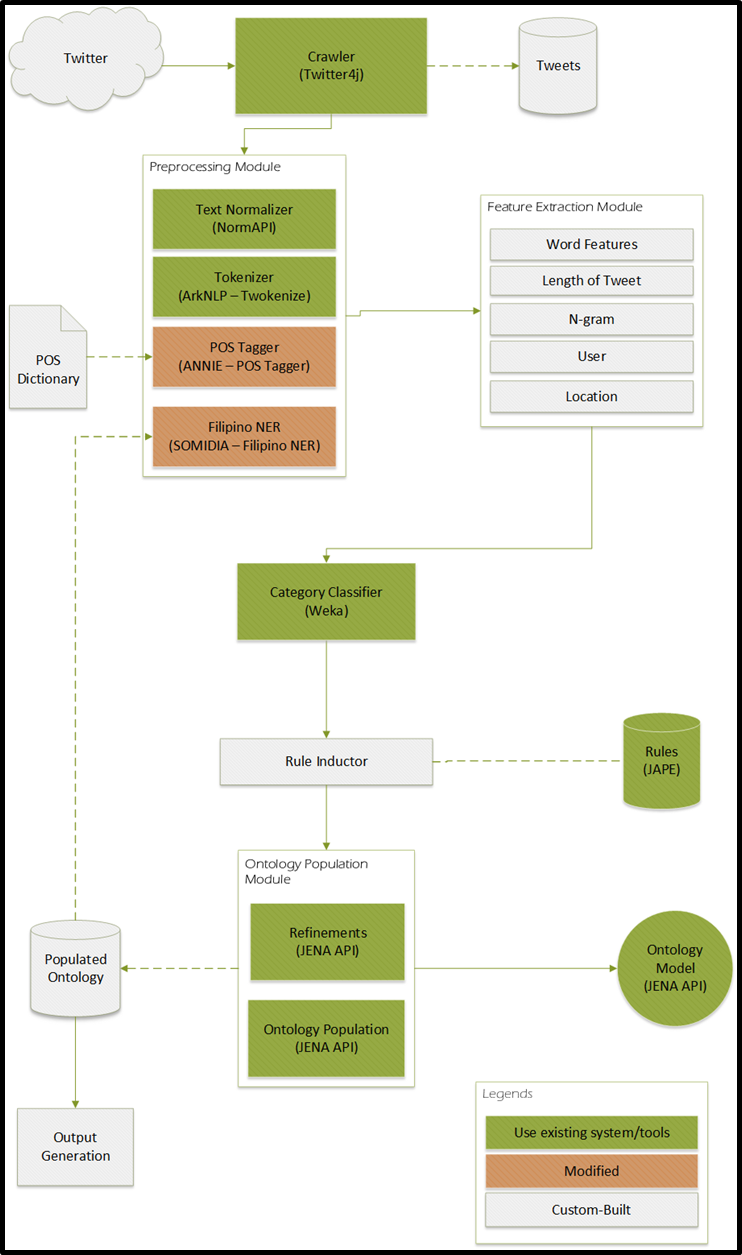


Figure ‑. FILIET Architectural Design

### Crawler Module

This module crawled Twitter to retrieve tweets. The system continuously collected the tweets using Twitter’s Stream API through the Twitter4j library.

### Preprocessing Module

This module is responsible for preprocessing the input tweets before they are passed on to the information extraction module. This module included the following text processing techniques: text normalizer, tokenizer, and POS Tagger. After going through this module, the preprocess tweets are then passed on to 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 (1) to convert the TXTSPK format of the tweets into full-word format so that the information when extracted will be consistent and (2) remove emoticons, links, and hashtags. The text normalizer accepts 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 used NormAPI (Nocon et al., 2014). Table 4‑1 shows a sample input and its corresponding output.

Table ‑. Sample Input/Output for Text Normalizer

|  |  |
| --- | --- |
| Input | Output |
| <tweet>  Dear Adnu sana po damit naman ang idonate natin para sa mga binagyo in case na may donation na ganapin. Plus canned goods na rin. Haha. :)  </tweet> | <tweet>  Dear Adnu sana po damit naman ang idonate natin para sa mga binagyo in case na may donation na ganapin. Plus canned goods na rin. Haha.  </tweet> |
| <tweet>  Kailangan na talaga ng military efforts sa most part of Leyte. Nagkakagulo na.  </tweet> | <tweet>  Kailangan na talaga ng military efforts sa most part of Leyte. Nagkakagulo na.  </tweet> |

#### Tokenizer

After normalizing the tweets, the tokenizer splits the input tweets into tokens like numbers, punctuations, words, abbreviations and other special characters like emoticons, hashtags, mentions, and the like. The tokenizer takes the normalized tweet as an input from the Text Normalizer. The tokenizer outputs an array containing the tokenized tweet in a form that is similar to this. Tokenized = {“@<username>”, “<punctuations>”, “#<hashtag>”...} or an array that would contain all the tokens in a given tweet. For this module, the researchers used ArkNLP’s Twokenize (Gimpel et al., 2011). Table 4‑2 shows a sample input and its corresponding output.

Table ‑. Sample Input/Output Tokenizer

|  |  |
| --- | --- |
| Input | Output |
| <tweet>  Dear Adnu sana po damit naman ang idonate natin para sa mga binagyo in case na may donation na ganapin. Plus canned goods na rin. Haha.  </tweet> | <tweet>  "Dear", "Adnu", "sana", "po", "damit", "naman", "ang", "idonate", "natin", "para", "sa", "mga", "binagyo", "in", "case", "na", "may", "donation", "na", "ganapin", ".", "Plus", "canned", "goods", "na", "rin", ".", "Haha", "."  </tweet> |
| <tweet>  Kailangan na talaga ng military efforts sa most part of Leyte. Nagkakagulo na.  </tweet> | <tweet>  "Kailangan", "na", "talaga", "ng", "military", "efforts", "sa", "most", "part", "of", "Leyte", ".", "Nagkakagulo", "na", "."  </tweet> |

#### POS Tagger

After tokenizing the tweets, the POS tagger accepts the tokenized Filipino tweet as an input and then, it tags each token with its corresponding part-of-speech. Each of the tokens are tagged as a noun, a verb, an adjective, an adverb or others. After tagging the tokens, the POS tagger outputs the tokens with their corresponding POS tag in the form of a text. For the module, the researchers used Filipino Tagger Dictionary (Oco & Borra, 2011). Table 4‑3 shows the sample input and output of POS tagger.

Table ‑. Sample Input/Output POS Tagger

|  |  |
| --- | --- |
| Input | Output |
| <tweet>  "Dear", "Adnu", "sana", "po", "damit", "naman", "ang", "idonate", "natin", "para", "sa", "mga", "binagyo", "in", "case", "na", "may", "donation", "na", "ganapin", ".", "Plus", "canned", "goods", "na", "rin", ".", "Haha", "."  </tweet> | <tweet>  "Dear\_UH", "Adnu", "sana\_VOTF", "po\_MAHM", "damit\_NCOM", "naman\_ENCL", "ang\_NA", "idonate", "natin\_PNGP", "para\_PRTA", "sa\_NCOM", "mga\_NA", "binagyo", "in\_IN", "case\_VBP", "na\_NA", "may\_MAEM", "donation\_NN:UN", "na\_NA", "ganapin", ".\_PSNS", "Plus\_JJ", "canned\_JJ", "goods\_NNS", "na\_NA", "rin\_ENCL", ".\_PSNS", "Haha\_NN", ".\_PSNS"  </tweet> |
| <tweet>  "Kailangan", "na", "talaga", "ng", "military", "efforts", "sa", "most", "part", "of", "Leyte", ".", "Nagkakagulo", "na", "."  </tweet> | <tweet>  "Kailangan\_VOTF", "na\_NA", "talaga\_IRIA", "ng\_NA", "military\_NCOM", "efforts\_NNS", "sa\_NCOM", "most\_JJS", "part\_JJ", "of\_IN", "Leyte\_NPRO", ".\_PSNS", "Nagkakagulo", "na\_NA", ".\_PSNS"  </tweet> |

#### Filipino NER

The Filipino NER identifies the proper nouns in the tweets. The module accepts the tweets that have passed through the preprocessing module. The output of the NER are tagged as proper nouns in the tweet. For the gazetteer, the SOMIDIA gazetteer was updated and used. Table 4‑4 shows a sample input and its corresponding output.

Table ‑. Sample Input/Output Gazetteer

|  |  |
| --- | --- |
| Input | Output |
| <tweet>  "Dear\_UH", "Adnu", "sana\_VOTF", "po\_MAHM", "damit\_NCOM", "naman\_ENCL", "ang\_NA", "idonate", "natin\_PNGP", "para\_PRTA", "sa\_NCOM", "mga\_NA", "binagyo", "in\_IN", "case\_VBP", "na\_NA", "may", "donation\_NN:UN", "na\_NA", "ganapin", ".\_PSNS", "Plus\_JJ", "canned\_JJ", "goods\_NNS", "na\_NA", "rin\_ENCL", ".\_PSNS", "Haha\_NN", ".\_PSNS"  </tweet> | <tweet>  "Dear\_UH", "Adnu", "sana\_VOTF", "po\_MAHM", "damit\_NCOM", "naman\_ENCL", "ang\_NA", "idonate", "natin\_PNGP", "para\_PRTA", "sa\_NCOM", "mga\_NA", "binagyo", "in\_IN", "case\_VBP", "na\_NA", "may", "donation\_NN:UN", "na\_NA", "ganapin", ".\_PSNS", "Plus\_JJ", "canned\_JJ", "goods\_NNS", "na\_NA", "rin\_ENCL", ".\_PSNS", "Haha\_NN", ".\_PSNS"  </tweet> |
| <tweet>  "Kailangan\_VOTF", "na\_NA", "talaga\_IRIA", "ng\_NA", "military\_NCOM", "efforts\_NNS", "sa\_NCOM", "most\_JJS", "part\_JJ", "of\_IN", "Leyte\_NPRO", ".\_PSNS", "Nagkakagulo", "na\_NA", ".\_PSNS"  </tweet> | <tweet>  "Kailangan\_VOTF", "na\_NA", "talaga\_IRIA", "ng\_NA", "military\_NCOM", "efforts\_NNS", "sa\_NCOM", "most\_JJS", "part\_JJ", "of\_IN", "<location: Leyte/>", ".\_PSNS", "Nagkakagulo", "na\_NA" ".\_PSNS"  </tweet> |

### Feature Extraction Module

This module is responsible for extracting the feature from the tweet. The module extracts 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; otherwise it is given “0”.

#### 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. To accomplish the n-gram generation/extraction tasks, the module makes use of the SRILM tool, which is specifically built for generating/extracting n-gram models.

#### User

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

### Category Classifier Module

Using the extracted features, the Category Classifier Module classifies the tweets into the following categories: (1) caution and advice (CA), (2) casualties and damage (CD), (3) donations (D), (4) call for help (CH), and (5) others (O). The module uses Weka (Weka, n.d.) and tried out different classifiers. Table 4‑5 shows a sample input/output of the Category Classifier Module.

Table ‑. Sample Input/Output Category Classifier Module

|  |  |
| --- | --- |
| Input | Output |
| <tweet>  "Dear\_UH", "Adnu", "sana\_VOTF", "po\_MAHM", "damit\_NCOM", "naman\_ENCL", "ang\_NA", "idonate", "natin\_PNGP", "para\_PRTA", "sa\_NCOM", "mga\_NA", "binagyo", "in\_IN", "case\_VBP", "na\_NA", "may", "donation\_NN:UN", "na\_NA", "ganapin", ".\_PSNS", "Plus\_JJ", "canned\_JJ", "goods\_NNS", "na\_NA", "rin\_ENCL", ".\_PSNS", "Haha\_NN", ".\_PSNS"  </tweet> | <tweet type=”D”>  "Dear\_UH", "Adnu", "sana\_VOTF", "po\_MAHM", "damit\_NCOM", "naman\_ENCL", "ang\_NA", "idonate", "natin\_PNGP", "para\_PRTA", "sa\_NCOM", "mga\_NA", "binagyo", "in\_IN", "case\_VBP", "na\_NA", "may", "donation\_NN:UN", "na\_NA", "ganapin", ".\_PSNS", "Plus\_JJ", "canned\_JJ", "goods\_NNS", "na\_NA", "rin\_ENCL", ".\_PSNS", "Haha\_NN", ".\_PSNS"  </tweet> |
| <tweet>  "Kailangan\_VOTF", "na\_NA", "talaga\_IRIA", "ng\_NA", "military\_NCOM", "efforts\_NNS", "sa\_NCOM", "most\_JJS", "part\_JJ", "of\_IN", "Leyte\_NPRO", ".\_PSNS", "Nagkakagulo", "na\_NA", ".\_PSNS"  </tweet> | <tweet type=”D”>  "Kailangan\_VOTF", "na\_NA", "talaga\_IRIA", "ng\_NA", "military\_NCOM", "efforts\_NNS", "sa\_NCOM", "most\_JJS", "part\_JJ", "of\_IN", "<location: Leyte/>", ".\_PSNS", "Nagkakagulo", "na\_NA" ".\_PSNS"  </tweet> |

### Rule Inductor Module

The Rule Inductor module accepts tokenized and tagged tweets. It applies the rules coming from the database. It looks for patterns in the text and apply the classification and generates the instances that will be used to populate the ontology. The rules try to match a pattern in the tweet, looking for part-of-speech, NER, and strings as its marker. In Table 4‑6 is an example of an extraction rule, the rules can composed of 1 or more tokens. In the example, the rule is looking for a string that has a word “lagpas” and a POS tag of NCOM (Common Noun). The Advice tells the extractor the type of information.

Table ‑. Example of an Extraction Rules

|  |
| --- |
| **Sample Rules** |
| <string:lagpas>[as]ADVICE <pos:NCOM>[as]ADVICE |

### Ontology Population Module

The ontology population module is responsible for filling up the ontology with extracted information from the previous module instances. It has two sub-modules: Ontology Population and Ontology Retrieval. Both submodules takes advantage of the OWL API to manipulate and modify the contents of the ontology. The structure of the ontology used for this module and this system is made using the Protégé ontology tool. Appendix I shows the representation of the ontology in OWL format.

#### Ontology Population

The Ontology Population module is responsible for storing the extracted information to the ontology by filling up the fields and asserting the relations that exists within the ontology. This sub-module includes pre-requisite functions to facilitate a seamless exchange of information between the system and the ontology like saving, loading, and etc. Due to the nature of ontologies, there is no need to check for duplicate instances within the ontology as the ontology does the checking and validating for this kind of scenario.

#### Ontology Retrieval

The Ontology Retrieval module is responsible for retrieving the information that was stored in the ontology by getting all the instances of ontological classes and assertions that exists within the ontology. This sub-module includes pre-requisite functions to facilitate a seamless retrieval of stored information between the system and the ontology like saving, loading, and etc. Due to the structure of the ontology designed for the system, the main point of contact for the retrieval process is the instances of the Tweet class of the ontology. Further retrieval of information is based on the relations that different instances from different classes have with the instances of the Tweet class.

### Data Sources

The data that will be collected will come from the filtered tweets. Some of these 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.

To be able to crawl the tweets that are strictly related to disaster relief operations, the researchers will make use of certain national official hashtags that are used by a number of relief organizations in the country. Examples of the unified hashtags are #ReliefPH, #RescuePH, #PHalert

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 4‑6 shows a sample of what can be seen in the CSV file.

Table ‑. Sample Entries of Tweets in CSV File

|  |  |
| --- | --- |
| # | Sample Output |
| 1 | 5280d16567833c59e17ebb66, SandyCervas, Dear Adnu sana po damit naman ang idonate natin para sa mga binagyo in case na may donation na ganapin. Plus canned goods na rin. Haha. :) , 11/11/2013 8:45, 13.7053384, 123.1980436 |
| 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 is used for the Filipino NER module. SOMIDIA’s gazetteer was updated and used. Table 4‑7 shows a sample gazetteer for the storm names in the Philippines.

Table ‑. Sample Gazetteer 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

Based on the tweets, the rules are handcrafted using JAPE. Then, the rules are stored in the database which will be used for extracting the information. Table 4‑8 shows a sample of the rules.

Table ‑. Sample Extracted Rules

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

#### Seed Words

The seed words are used for generating the rules. The list of seed words are stored in a text file. SOMIDIA’s seed word list is used and updated. Table 4‑9 shows the excerpts of the list of seed words.

Table ‑. Excerpts of the List of Seed Words

|  |  |  |  |
| --- | --- | --- | --- |
| tubig  kuryente  pagkain  tulong  donation  damit  gutom | rice  kanin  bigas  inumin  sardinas  sardines  canned goods | water  clothes  food  help  bahay  gamot  medicine | health kit  medical kit  relief goods  kasuotan  instant noodles  damit  pera |

#### POS Dictionary

The POS Dictionary is a dictionary that contains a list of words with its POS tag. This is used in the POS Lookup. The dictionary is stored in a file and contains a list of English and Filipino words. Table 4‑10 shows a sample of the excerpts of the POS dictionary

Table ‑. Excerpts of the POS Dictionary

|  |  |
| --- | --- |
| storms storm ENG NNS  storms storm ENG VBZ  storm storm ENG NN  storm storm ENG VB  bukid bukid TAG NCOM 2  bukirin bukirin TAG NCOM 2 | buko buko TAG NCOM 2  bula bula TAG NCOM 2  bulag bulag TAG NCOM 2  bulak bulak TAG NCOM 2  bulalas bulalas TAG NCOM 2 |

#### Ontology

For the ontology, this was created manually. The domain of the ontology is disaster, specifically for relief operations. The next step was the identification of the terms. After identifying the terms, the concept, properties, and constraints were defined. Class instantiations then follow. The format of the ontology is in OWL. Figure 4‑2 shows the ontology of the system.

## System Functions

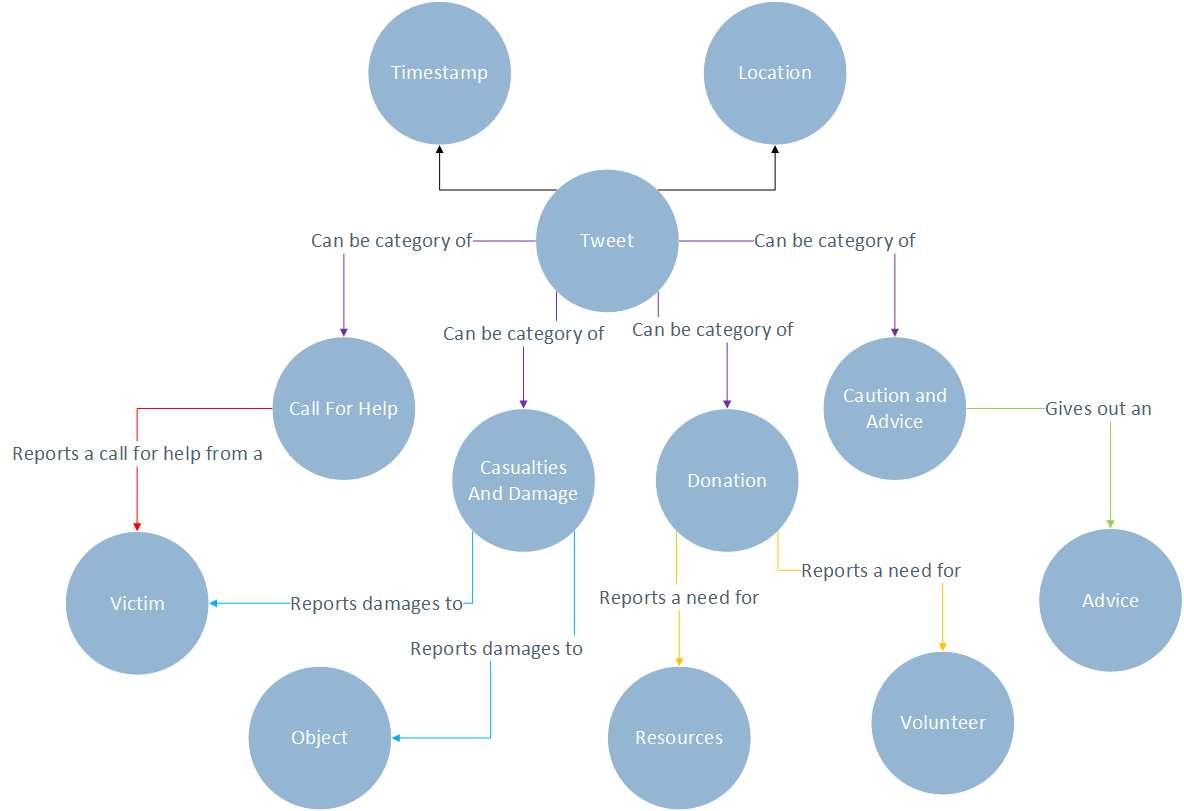


Figure ‑. FILIET Ontology

This section discusses the functions of the proposed systems.

### Tweet Retrieval

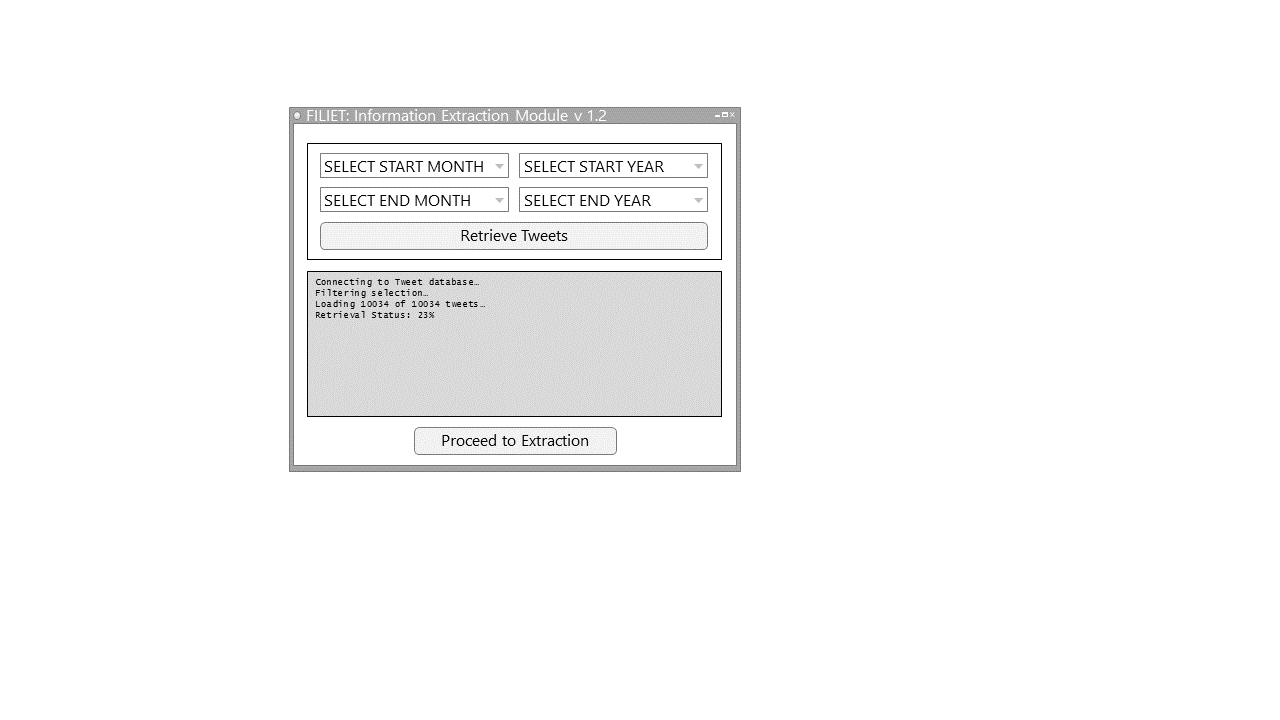


Figure ‑. Tweet Retrieval Screenshot

In this function, the system will access the tweets that were stored in the database by the Twitter crawler. The user can opt to filter the tweets for retrieval. Figure 4‑3 shows the screenshot of this function.

### Information Extraction

In this system function, the information extraction process starts with feature extraction which shall then be used for the classification of the tweets based on the categories defined in the system. After classification, the tweets shall then be examined for possible rules. The rules to be generated will then be applied to the tweets. Extracted information will be fed into the next function. Figure 4‑4 shows a screenshot of this function.

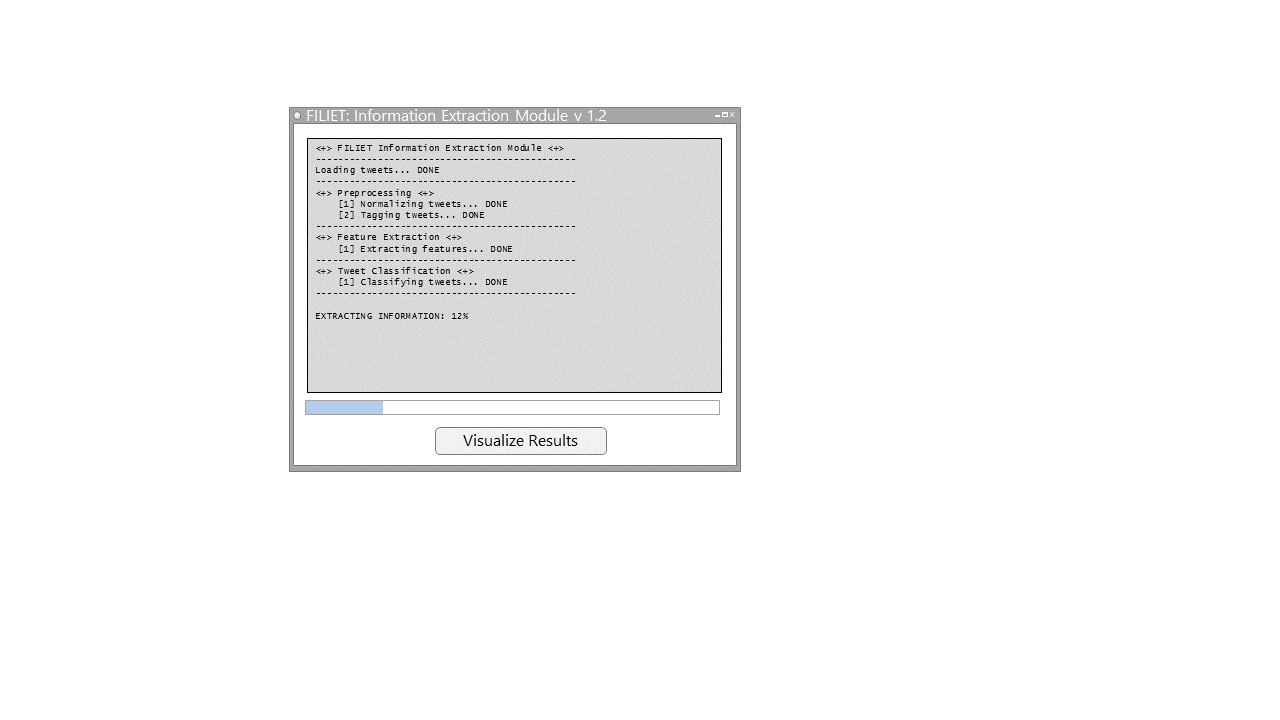


Figure ‑. Information Extraction Screenshot

### Ontology Population

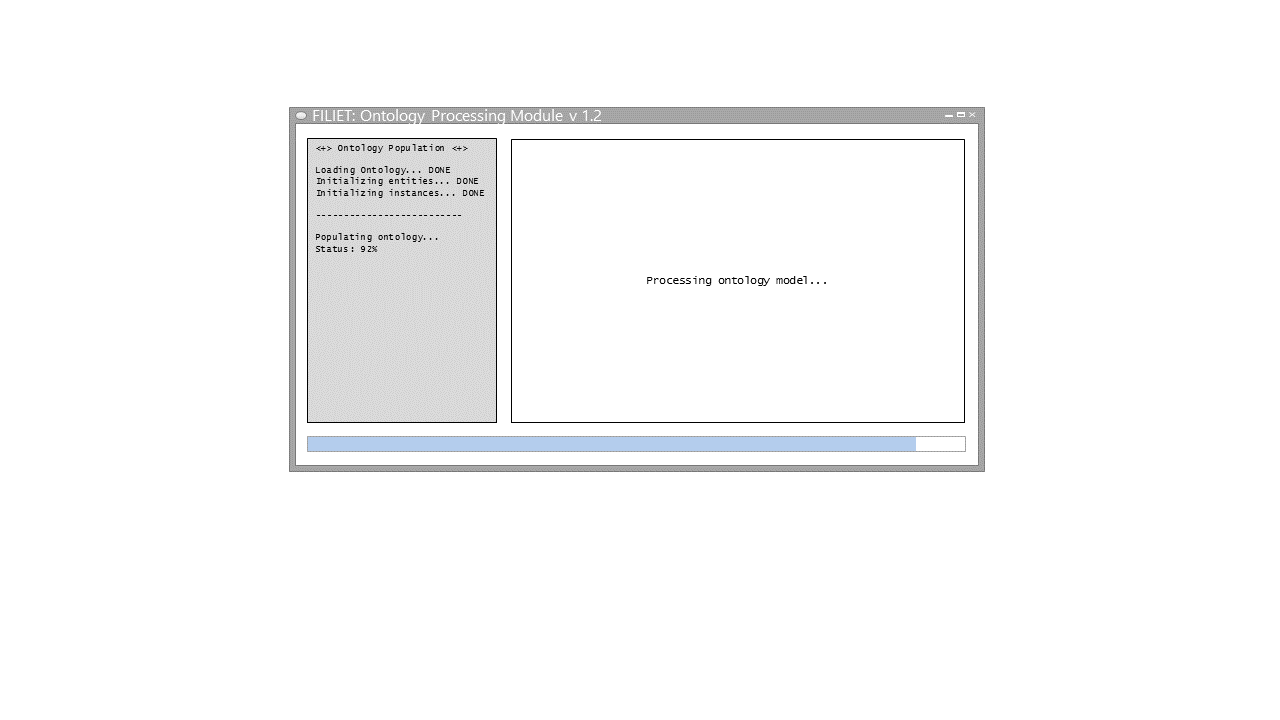


Figure ‑. Ontology Population Screenshot

In this system function, the extracted information will be initialized as entity instances for population of the ontology. The system, by default, automatically validates each of the extracted information before being introduced to the ontology. During validation, the system will check if the entity instances exist and if they do, the system will match the instances to their corresponding entity class/es. If they do not, the instances will immediately be discarded. Figure 4‑5 shows the screenshot of this function.

### Ontology Retrieval

In this system function, the populated ontology and the details of the instances per entity class can also be viewed. Relationships within entities can be seen or searched given a selected instance from the ontology model. Figure 4‑6 shows a screenshot of this function.

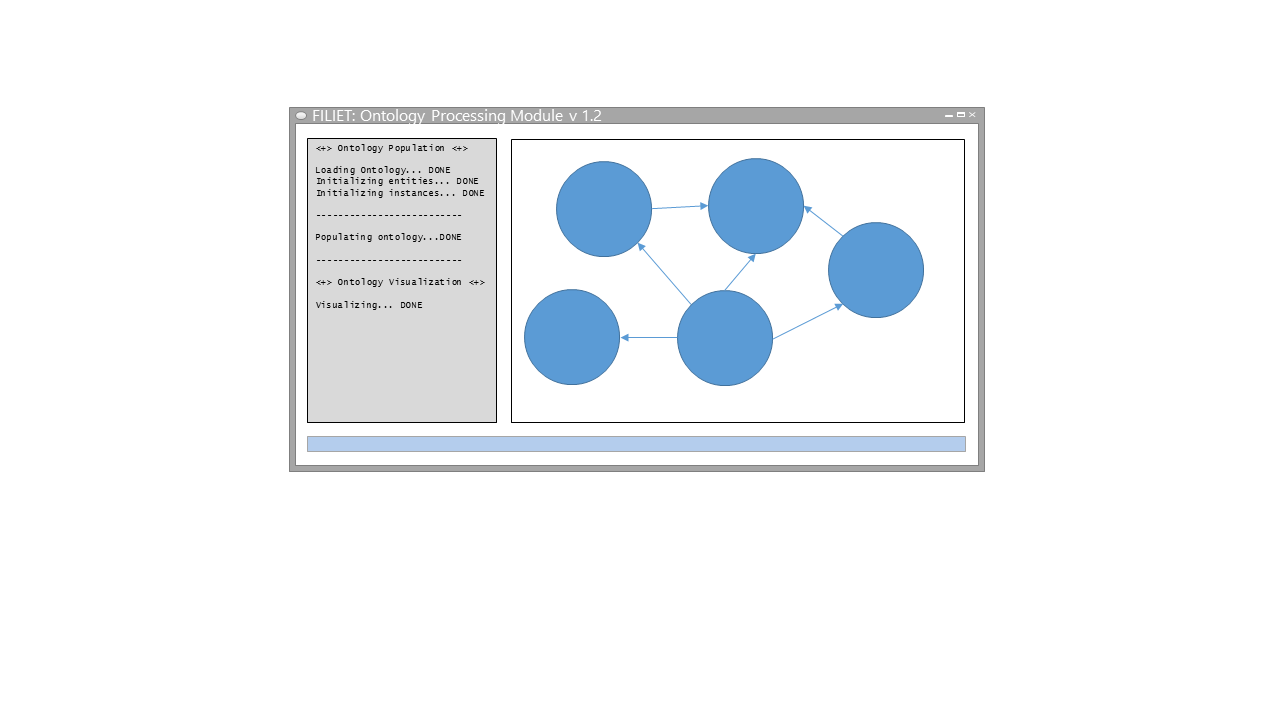


Figure ‑. Ontology Access Screenshot

## 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

# Design and Implementation Issues

This chapter discusses the design and implementation of the system as well as the issues encountered during its development.

## Resource Gathering

### Dataset Building

The Twitter crawler was deployed during the duration of Typhoons Mario (September 9, 2014) and Ruby (December 8, 2014). From the tweets collected, two datasets were formed. The first dataset is composed of Mario tweets whereas the second dataset is composed of Ruby tweets. Both datasets were manually annotated to which of the following categoried they belong to: Caution and Advise (CA), Casualty and Damage (CD), Call for Help (CH), Donation (D), and Others (O). The labels are manually annotated by the proponents. The annotation is done by votation. Each of the proponents will classify the the instance. Then, the majority was used as the label of the instance. Table 5‑1 details the contents of the two datasets. To measure the annotators’ agreement, kappa’s statistics was used as the metrics. The Mario dataset scored 0.884 kappa, and the Ruby dataset scored 0.979 kappa.

Table ‑. Contents of Each Dataset

|  |  |  |
| --- | --- | --- |
| **Category** | **Number of Instances** | |
| **Mario Dataset** | **Ruby Dataset** |
| CA | 651 (47.69 %) | 1279 (49.52 %) |
| CD | 99 (7.25 %) | 245 (9.49 %) |
| CH | 58 (4.25 %) | 9 (0.35 %) |
| D | 47 (3.44 %) | 37 (1.43 %) |
| O | 510 (37.36 %) | 1013 (39.22 %) |
| **Retweets** | 655 (47.99 %) | 1636 (63.34 %) |
| **Official** | 7 (0.5%) | 10 (0.4%) |
| **Kappa** | 0.884 | 0.979 |
| **Total** | 1365 | 2583 |

A large number of tweets were collected for both typhoons. However, a large percentage of these tweets were irrelevant, i.e. not disaster-related, irresponsible use of hastags. Aside from the large number of irrelevant tweets, there is also the issue of ambiguity. Some tweets can be annotated in more than one category. The current resolution done for the ambiguity issue is a consensus among the proponents as to which category the said tweet belongs to. The large number of tweets can be attributed to tweets being retweeted multiple times thus leading to only a few distinct tweet instances, with 47.99% of dataset for Mario dataset and 63.34% for Ruby dataset. Furthermore, the datasets created have an unbalanced distribution among the categories.

### Feature Words

Feature Words are collected from the Ruby dataset. These words will serve as part of the featureset in the classification. To get the list of feature words, the tweets are grouped into their respective categories. From each category, the distinct words were gathered except for the stopwords. The list of stopwords excluded from the collection of distinct words are listed in Appendix D. Each word is then weighted using the TFIDF weighting schemes. If a feature word appears in more than one category, it is discarded from the list. The top 10%, 20%, and 30% will serve as the feature words. Appendix F lists the feature words used.

## Crawler Module

The crawler module is responsible for the collection of disaster-related tweets. It uses Twitter4j API (Twitter4j, n.d.), an unofficial Java library that uses Twitter API, to crawl Twitter. This module uses an authenticated account that follows users. In order to get disaster-related tweets, the official hashtags used by the government were used to filter the tweets (please refer to Appendix C for the list of hashtags used). The module uses Twitter’s User Stream API to continuously listen for new tweets. Each tweet is then stored into the database.

Code Listing ‑. Crawl for tweets with the specified keywords

|  |
| --- |
| // Filters  FilterQuery fq = new FilterQuery();  String keywords[] = {"#reliefPH","#nopower", "#nowater", "#roadalert", "#tracingPH", "#rescuePH", "#floodPH", "#queenieph"};  fq.track(keywords);    TwitterStream tweetStream = twitterStreamFactory.getInstance();  tweetStream.addListener(listener);  tweetStream.filter(fq); |

Twitter4j retrieves the tweets the moment a tweet is updated in the account’s timeline. The listener is handled by the UserStreamListener, specifically the onStatus() method. The method will first receive a Status object. This contains all information regarding the tweet. Then, the following information is then processed: TweetID, User, Tweet, Latitude, Longitude, Language, IsUrl, IsHashtag, and IsRetweet to a Tweet object. After binding the Status object to the Tweet object, the Tweet object handles the storing of the tweet information to the database.

Code Listing ‑. Store the Twitter status to the database

|  |
| --- |
| Tweet tweet = new Tweet(status);  try {  tweet.StoreTweet();  } catch (SQLException e) {  e.printStackTrace();  } |

There were several issues encountered in the crawler module. In the first implementation of the crawler module, it was ran to collect tweets during the Typhoon Mario. The some of the tweets that were collected were in multi-line format. The collected tweets are then manually cleaned in order to remove the line breaks. Another problem encountered was the amount of irrelevant tweets that the crawler is collected. Even with using the official hashtags as a filter, the crawler is still getting numerous irrelevant tweets because people use the hashtags irresponsibly.

## Preprocessing Module

### Text Normalizer

The first task to do in the Preprocessing Module is to clean the tweets by normalizing the text the tweet contains. To accomplish this task, the NormAPI library (Nocon et al., 2014) was used. Using the API is easy to use because only the normalizeText(String) method is needed.

Code Listing ‑. Using NormAPI

|  |
| --- |
| String normalizedText = NormAPI.normalize\_Text(shortcutText); |

The normalization starts with the initialization of the Normalizer. The Normalizer needs a class that has implemented a NormalizerInterface.

Code Listing ‑. NormAPI Approach

|  |
| --- |
| normalizer = new Normalizer(new NormApiImpl()); |

To use the Normalizer, invoke the executeStrategy(String). The executeStrategy() will expect a String object that contains the actual tweet. The Normalizer will return a String object that is now normalized.

Code Listing ‑. FILIET Normalizer Execution

|  |
| --- |
| normalizedTweet = normalizer.executeStrategy(text); |

The only issue in using NormAPI to set up all the necessary pre-requisite tools and packages that NormAPI needs. There came a point in time where all the pre-requisites were installed and properly working but NormAPI still would not work. With the help of one of the API’s proponents, the cause of the error was found out to be that the copy of NormAPI.jar included in the installation files was the wrong one.

### Tokenizer

This is the second step in processing the incoming tweets that are coming from web crawler or a CSV file. The tokenizer takes a tweet and parses them into tokens. It returns a Sentence object that contains the array list of Tokens.

The Tokenizer module has two implementations: ArkNLP (Gimpel et al., 2011) and OpenNLP (Apache Software Foundation, 2010). However, other implementations could be easily added by implementing the TokenizerInterface. The TokenizerInterface returns a Sentence object that contains the tweets.

The tokenizer from ArkNLP is called using the static method tokenizeRawTweetText(String text) from the ArkNLP Library. It will then return the tokenized tweet through an array list of string.

Code Listing ‑. ArkNLP Tokenizer Approach

|  |
| --- |
| List<String> tokens = Twokenize.*tokenizeRawTweetText*(text); |

For the OpenNLP tokenizer, a TokenizerModel is needed to initialize the TokenizerME.

Code Listing ‑. OpenNLP Tokenizer Approach

|  |
| --- |
| List<String> tokens = Twokenize.*tokenizeRawTweetText*(text); |

To use the tokenizer in FILIET, a Tokenizer must be initialized first. The Tokenizer is accepting a class that implemented the TokenizerInterface, which is either the ArkNLP or OpenNLP approach. This is done so that tokenizer could be easily modified in the future.

Code Listing ‑. Tokenizer Implementations in FILIET

|  |
| --- |
| // ArkNLP implementation  Tokenizer tokenizer = new Tokenizer(new ArkNLPTokenizerImpl());  // OpenNLP implementation  Tokenizer tokenizer = new Tokenizer(new OpenNLPTokenizerImpl()); |

Both implementations implement the TokenizerInterface wherein they will only call the tokenize() method to tokenize the tweet. This returns an array of strings where each element is a token of the tweet.

Code Listing ‑. OpenNLP Tokenizer Approach

|  |
| --- |
| tokens = tokenizer.tokenize(text); |

To execute the tokenizer in FILIET, call the executeStrategy(String).

Code Listing ‑. FILIET Tokenizer Execution

|  |
| --- |
| tokens = tokenizer.executeStrategy(normalizedTweet); |

The issue of implementing the OpenNLP tokenizer is that it needs a model. There are currently no Filipino model that could be used for the tokenizer, so the group used an English model. Another issue in OpenNLP is tokenizing emoticons. It splits them into two tokens, when it should only be one. In comparison to ArkNLP tokenizer, ArkNLP is very simple to use as it only needs to call a static method from the library. It is much more adept to tweets, because it can handle emoticons. Also, ArkNLP performs significantly faster than OpenNLP. Therefore, ArkNLP tokenizer was preferred.

### POS Tagger

After tokenization, each of the tokens are tagged with its corresponding parts-of-speech tag. The POSTagger returns the Sentence object that now contains the POS Tags. The POSTagger uses a lookup to tag each token.

The POS Tagger starts with the initialization of the POSTagger. The POSTagger needs a class that has implemented a POSTaggerInterface.

Code Listing ‑. POS Dictionary Look-up Approach

|  |
| --- |
| POSTagger post = new POSTagger(new POSHashLookupImpl()); |

To use the POS Tagger, invoke the executeStrategy(Sentence). The executeStrategy() will expect a Sentence object that contains the array list of tokens. The POSTagger will return a Sentence object that is now tagged with POS.

Code Listing ‑. FILIET POS Tagger Execution

|  |
| --- |
| tokens = post.executeStrategy(tokens); |

The main issue in implementing the POS Tagger is that there are no available tools for the Filipino Language. The look-up approach implemented uses a dictionary that contains the tags for each word. However, the words found in the dictionary is not suitable for tweets since it was extracted from the novels *Noli Me Tangere* and *El Filibusterismo*.

### Filipino NER

Named Entity Recognition module accepts a Sentence object that contains the tokenized and POS tagged tweet. Using a dictionary, it tags the words that are locations. The NER module needs an object that implements the NER Interface. This object contains the actual implementation of the NER. The SomidiaNERImpl contains the implementation of a lookup NER. It uses the dictionary from SOMIDIA.

Code Listing ‑. SOMIDIA’s NER Approach

|  |
| --- |
| NamedEntityRecognizer ner = new NamedEntityRecognizer (new SomidiaNERImpl()); |

After the initialization, the NamedEntityRecognizer can now be used by calling the executeStrategy(Sentence). The method returns the Sentence that is now tagged with NER.

Code Listing ‑. FILIET Filipino NER Execution

|  |
| --- |
| tokens = ner.executeStrategy(tokens); |

The main issue in this module is that it can only tag one-word locations. It only looks at one token at a time. The problem now arises from locations that contains two or more words because the current approach does not support them. Another problem facing the NER module is that there are locations that are considered as adjectives and vice versa. In the NER gazetteer, locations like “Maginhawa” and “Salamat” could be considered as other type. So, when the NER encountered a tweet like “Maraming salamat po”. The word “Salamat” will be tagged as a named entity when it should not be.

### Preprocessor Manager

This module is responsible for initializing all of the sub-modules under the preprocessing module. The preprocessor manager accepts a String and then outputs a Sentence object that has been normalized, tokenized, POS tagged and NER tagged.

First, the preprocessor must be initialized. The preprocessor manager then initializes the sub-modules. For the normalizer, it initializes a normalizer using a NormApiImpl object. The tokenizer is initialized with ArkNLPTokenizer object. The POS Tagger is initialized with POSHashLookupImpl. Lastly, the NER Tagger is initialized with SomidiaNERImpl.

Code Listing ‑. FILIET Preprocessor Manager Initialization

|  |
| --- |
| PreprocessorManager preprocess = new PreprocessorManager();  Sentence preprocessed = preprocess.PreprocessText(String); |

## Feature Extraction Module

The feature extractor module is for extracting word features, presence features, and tweet length of a tweet. The n-gram feature is no longer considered because the initial assumption for the use of this feature is for the shortcut texts. However, this assumption was resolved with text normalization.

The module requires the files that contains the word features. The feature extractor counts how many times a word from the word features appeared in the tweet. For the presence of hashtags, it checks each token if it contains *“#”* sign. If it contains a hashtag, it will be labeled as 1, else 0. For the presence of mentions, it checks each token if it contains *“@”*. Then, it labels it as 1, else 0. Lastly, for the presence of URL, it checks each token will contain “http”. If it does, it labels it as 1, else 0.

The word features were extracted from the Mario dataset and the Ruby dataset. The tweets are then separated according to their annotated category. Then, the TFIDF scores are computed for each words. After getting the TFIDF scores, the top 10%, 20% and 30% highest scoring TFIDF is collected, excluding the stop words (Refer to Appendix D for the list of stop words). Then, the proponents checked the list of words and removed proper nouns and irrelevant words. The propoents used voting system to tell if the word is relevant or not. The process is done to the two datasets.

To use this, FeatureExtractor must first be initialized. It requires one parameters, the path to the the feature words files.

Code Listing ‑. FILIET Feature Extraction Initialization

|  |
| --- |
| String word = "./resources/model/word/ruby-word";  FeatureExtractor fe = new FeatureExtractor(word); |

After initializing, the FeatureExtractor can either do single processing or batch processing. For the single processing, the method extract(Sentence) is used. The method will be expecting a Sentence object that has been preprocessed. Then it will output a Sentence object that now contains extracted features. For the batch processing, the method extract(String, String) is used, where the method expects the path to a CSV file that contains the tweets, and the path where the output will be saved.

Code Listing ‑. FILIET Feature Extraction Execution

|  |
| --- |
| // Single Processing  fe.extract(sentence);  // Batch Processing  String tweets = "./resources/tweets/mario-datasets/original/ruby-dataset.csv”;  String saveModel = "./resources/tweets/test-extracted/mario-tfidf/ruby-extracted.csv  fe.extractFeatures(tweets,saveModel); |

## Category Classifier Module

The classifier module categorizes the tweet into one of the following categories: Caution and Advice (CA), Casualties and Damage (CD), Donation (D), Call for Help (CH), and Other (O). The classifier uses a model in order to classify the tweets. Models are trained using Weka (Weka, n.d.). The classifier module accepts a String object that has passed through the Preprocessing Module and Feature Extractor Module and returns the category of the tweet.

First, the classifier must be initialized. The classifier accepts a class that has implemented a ClassifierInterface. There are current two implementations of the classifier. The first is the single classifier. This classifier uses only one (1) classifier to classify all of the categories. The second is a multiple classifier. The multiple classifier is consists of five (5) classifiers, one for each category and multi-label classifier. Each of the classifier classifies the instance, if all of the instance classified it as O, then the category of that instance is O. However, if it is not a majority, the multi-label classifier checks the classification of the binary classifiers. If one the classifier matches the multi-label classifier, then that is the classification, else the classification is O. For the classification algorithm, all of the classifier uses Random Forest algorithm.

Code Listing ‑. FILIET Classifier Implementations

|  |
| --- |
| // Single Classifier  Classifier classifier = new Classifier(new ClassifierImpl());  // Multi Classifier  Classifier classifier = new Classifier(new MultiClassifierImpl()); |

The classifier initializes a ClassifierBuilder class. This class is responsible for binding the Sentence object to an Instance object that is used for the classification. The ClassifierBuilder has two constructors. The first constructor takes no parameters. This sets the word file to its default. The second constructor takes a parameter: path to the word file.

Code Listing ‑. Classifier Builder Initialization

|  |
| --- |
| // Default model  ClassifierBuilder builder = new ClassifierBuilder();  // Path to the model resource.  ClassifierBuilder builder = new ClassifierBuilder(wordPath); |

To run the classification, invoke the executeStrategy(Sentence) method.

Code Listing ‑. FILIET Classifier Execution

|  |
| --- |
| String category = classifier.executeStrategy(temp); |

## Rule Inductor Module

Information extractor is the module responsible for extracting the relevant information from the tweets. The module accepts preprocessed and classified tweets and outputs the Sentence object with the extracted information. It uses the hand-crafted rules to extract the information.

To initialize the Rule Inductor module, the constructor accepts a single string parameter. This is the file path to the rule file.

Code Listing ‑. Rule Inductor Initialization

|  |
| --- |
| RuleInductor ruleInductor = new RuleInductor(rulePath); |

For the rule file, the rules are categorized into four categories. They are separated by <Category>: [category]. One rule is listed per line. Then, the <end> tag is used to signify the end of list for that [category]. Refer to Appendix H for the complete list of extraction rules.

Code Listing ‑. Sample Extraction Rules for CA Category

|  |
| --- |
| <Category>: CA  <pos:JJ> <pos:NN> <pos:PSNS> <number:ANY>  <ner:LOCATION>[as]LOCATION  <pos:JJ> <string:#1>  <pos:JJ> <string:#2>  <pos:JJ> <string:#3>  <pos:VBZ> <string:classes> <pos:IN> <pos:JJ> <pos:VBZ>  <pos:VBP> <string:classes> <pos:IN> <pos:JJ> <pos:VBZ>  <string:#walangpasok> <pos:JJ> <pos:VBZ>  <string:signal> <pos:NN> <pos:PSNS> <number:ANY>  <string:#walangpasok> <pos:PSNS> <string:klase>  <string:#walangpasok> <string:sa> <pos:PIDP> <pos:NA> <string:antas>  <end> |

For the construction of extraction rules, each rule can consist of the following tags: string, number, pos, ner. The string tag is matched to the token’s word. The pos tag is matched to the token’s POS. The number tag is used to match numbers. Lastly, the ner is matched to the token’s NER. To use wildcards, the key “ANY” to match any values.

The Rule Inductor module uses match(Sentence) method to apply the rules.

Code Listing ‑. Rule Inductor Initialization

|  |
| --- |
| ruleInductor.setExtractedInformation( ruleInductor.match( extractedTweet ) ); |

The implementation of the FILIET architecture itself presents the problem of propagation of errors that may arise from the different modules that precede it. Another problem is the over application of rules. Even if the tweets are already categorized, thus only applying these specific set of rules, the tweet can still be matched to various rules making the system extract extraneous information.

## Ontology Population Module

After extracting the relevant information from the tweets based on their respective categories, they are now stored to an ontology that contains object relations between the different extracted information. The actual structure of the ontology was made using an external tool called Protegé that makes use of the OWL API. This module takes an instance or a list of instance of categorized tweet classes. The categorized tweet classes include the following: the CallForHelpTweet class for containing the information that were gathered under the Call For Help category; CasualtiesAndDamageTweet class for containing the information that were gathered under the Casualties and Damage category; CautionAndAdviceTweet class for containing the information that were gathered under the Caution and Advice category; lastly, DonationTweet class for containing the information that were gathered under the Casualties and Damage category. This module has two sub-parts that are both responsible for storing and accessing information in the ontology. The OntologyModule class is responsible for storing the extracted information to the ontology and the OntologyRetriever class is responsible for retrieving the information that was stored in the ontology.

The actual storing of information involves more sub modules or methods which helps streamlined the process. The process starts with the call to the respective method for storing the information extracted from its tagged category. Inside this method, there are certain variables that should be initialized so that they can be interfaced with the actual ontology. A sample code listing below shows the different variables to be initialized – take the addCautionAndAdviceReport for example.

Code Listing ‑. addCautionAndAdviceReport Initialization

|  |
| --- |
| // PREPARE THE TWEET INDIVIDUAL  OWLIndividual tweetIndividual = addTweetInformation(ca);    // PREPARE THE REASONER AND PREREQUISITE STUFF  OWLDataFactory dataFactory = manager.getOWLDataFactory();    // INSTANTIATE CLASSES NEEDED FOR THIS OPERATION  OWLClass cautionAdviceClass = dataFactory.getOWLClass(IRI.create(CAUTION\_ADVICE\_CLASS\_IRI));  OWLClass adviceClass = dataFactory.getOWLClass(IRI.create(ADVICE\_CLASS\_IRI));    // CREATE THE NEEDED INDIVIDUALS FOR THIS OPERATION  OWLIndividual caIndividual = dataFactory.getOWLNamedIndividual(IRI.create(BASE\_IRI + "\_CA-I\_" + ca.getTweetAdvice().replace(" ", "\_")));  OWLIndividual adviceIndividual = dataFactory.getOWLNamedIndividual(IRI.create(BASE\_IRI + "\_A-I\_" + ca.getTweetAdvice().replace(" ", "\_"))); |

After initializing the different needed variables, the next step is to prepare the Tweet individual that will serve as the root individual for storing all the extracted information into the ontology. To do so, call the addTweetInformation method with an instance of the CautionAndAdviceTweet class as a parameter for this method. This method returns an OWLIndividual that is used as a prerequisite for completing the storage process. Furthermore, this method prepares the Tweet individual to include other information that was extracted including the Tweet Location and the Tweet Timestamp. A sample code listing below shows how the addTweetInformation method works.

Code Listing ‑. Prerequisite Class Instantiations

|  |
| --- |
| // PREPARE THE REASONER AND PREREQUISITES  OWLDataFactory dataFactory = manager.getOWLDataFactory();    // INSTANTIATE CLASSES NEEDED FOR THIS OPERATION  OWLClass tweetClass = dataFactory.getOWLClass(IRI.create(TWEET\_CLASS\_IRI));  OWLClass locationClass = dataFactory.getOWLClass(IRI.create(LOCATION\_CLASS\_IRI));  OWLClass timestampClass = dataFactory.getOWLClass(IRI.create(TIMESTAMP\_CLASS\_IRI)); |

Code Listing ‑. Ontology Individual Creations

|  |
| --- |
| // CREATE THE NEEDED INDIVIDUALS FOR THIS OPERATION  OWLIndividual tweetIndividual = dataFactory.getOWLNamedIndividual(IRI.create(BASE\_IRI + "\_TI\_" + t.getTweetContent().replace(" ", "\_")));  OWLIndividual locationIndividual = dataFactory.getOWLNamedIndividual(IRI.create(BASE\_IRI + "\_LI\_" + t.getTweetContent().replace(" ", "\_")));  OWLIndividual timestampIndividual = dataFactory.getOWLNamedIndividual(IRI.create(BASE\_IRI + "\_TSI\_" + t.getTweetContent().replace(" ", "\_"))); |

Code Listing ‑. Class-Individual Assertions

|  |
| --- |
| Set<OWLClassAssertionAxiom> classAssertion = new HashSet<OWLClassAssertionAxiom>();    ....  OWLClassAssertionAxiom lAssertion = dataFactory.getOWLClassAssertionAxiom(locationClass, locationIndividual);  classAssertion.add(lAssertion);  ....    manager.addAxioms(filietOntology, classAssertion); |

Code Listing ‑. Data Property Assertion

|  |
| --- |
| Set<OWLDataPropertyAssertionAxiom> dataAssertion = new HashSet<OWLDataPropertyAssertionAxiom>();    // These are the data properties for the Tweet Individual  OWLDataProperty tweetHandle = dataFactory.getOWLDataProperty(IRI.create(BASE\_IRI + "tweetHandle"));  OWLDataPropertyAssertionAxiom tweetHandleDataPropertyAssertion = dataFactory.getOWLDataPropertyAssertionAxiom(tweetHandle, tweetIndividual, t.getTweetHandle());  dataAssertion.add(tweetHandleDataPropertyAssertion);  OWLDataProperty tweetContent = dataFactory.getOWLDataProperty(IRI.create(BASE\_IRI + "tweetContent"));  OWLDataPropertyAssertionAxiom tweetContentDataPropertyAssertion = dataFactory.getOWLDataPropertyAssertionAxiom(tweetContent, tweetIndividual, t.getTweetContent());  dataAssertion.add(tweetContentDataPropertyAssertion);  ....  manager.addAxioms(filietOntology, dataAssertion); |

Code Listing ‑. Object Property Assertion

|  |
| --- |
| Set<OWLObjectPropertyAssertionAxiom> objectAssertion = new HashSet<OWLObjectPropertyAssertionAxiom>();    OWLObjectProperty hasThisInfo = dataFactory.getOWLObjectProperty(IRI.create(BASE\_IRI + "has\_this\_information"));  OWLObjectPropertyAssertionAxiom hasLocationAssertion = dataFactory.getOWLObjectPropertyAssertionAxiom(hasThisInfo, tweetIndividual, locationIndividual);  objectAssertion.add(hasLocationAssertion);  OWLObjectPropertyAssertionAxiom hasTimestampAssertion = dataFactory.getOWLObjectPropertyAssertionAxiom(hasThisInfo, tweetIndividual, timestampIndividual);  objectAssertion.add(hasTimestampAssertion);    manager.addAxioms(filietOntology, objectAssertion); |

After preparing the Tweet individual, the next step is to assert or tell the ontology that the newly instantiated classes are instances of the instantiated Ontological classes. A sample code listing below shows how to assert individual instances as class instances.

Code Listing ‑. Asserting Individual Instances as Class Instances

|  |
| --- |
| Set<OWLClassAssertionAxiom> classAssertion = new HashSet<OWLClassAssertionAxiom>();    OWLClassAssertionAxiom caAssertion = dataFactory.getOWLClassAssertionAxiom(cautionAdviceClass, caIndividual);  classAssertion.add(caAssertion);  OWLClassAssertionAxiom adAssertion = dataFactory.getOWLClassAssertionAxiom(adviceClass, adviceIndividual);  classAssertion.add(adAssertion);    manager.addAxioms(filietOntology, classAssertion); |

After asserting that the newly instantiated classes are instances of the instantiated Ontological classes, the next step is to add the extracted information to the different individuals as their respective data properties. A sample code listing below shows how to add data properties to the individuals that were instantiated.

Code Listing ‑. Adding Data Properties to Instantiated Individuals

|  |
| --- |
| Set<OWLDataPropertyAssertionAxiom> dataAssertion = new HashSet<OWLDataPropertyAssertionAxiom>();    OWLDataProperty tweetAdvice = dataFactory.getOWLDataProperty(IRI.create(BASE\_IRI + "tweetAdvice"));  OWLDataPropertyAssertionAxiom adviceDataPropertyAssertion = dataFactory.getOWLDataPropertyAssertionAxiom(tweetAdvice, adviceIndividual, ca.getTweetAdvice());  dataAssertion.add(adviceDataPropertyAssertion);    manager.addAxioms(filietOntology, dataAssertion); |

After adding data properties to the individuals that were instantiated, the next step is to assert or tell the ontology the different relations that connect the different individuals that were instantiated. In ontological terms, this is called Object Properties. This is the part where connecting the different individuals are connected so that they would make sense when stored into the ontology. Examples of these connections include the “has\_this\_information” connection, the “gives\_out\_an” connection and the like. A sample code listing below shows how to add object properties between the individuals that were instantiated.

Code Listing ‑. OntologyModule Initialization

|  |
| --- |
| Set<OWLObjectPropertyAssertionAxiom> objectAssertion = new HashSet<OWLObjectPropertyAssertionAxiom>();    // Say that the Tweet Individual is of the category CA Individual  OWLObjectProperty canBeOfTheCategory = dataFactory.getOWLObjectProperty(IRI.create(BASE\_IRI + "can\_be\_of\_the\_category"));  OWLObjectPropertyAssertionAxiom canBeAssertion = dataFactory.getOWLObjectPropertyAssertionAxiom(canBeOfTheCategory, tweetIndividual, caIndividual);  objectAssertion.add(canBeAssertion);    OWLObjectProperty givesOutAn = dataFactory.getOWLObjectProperty(IRI.create(BASE\_IRI + "gives\_out\_an"));  OWLObjectPropertyAssertionAxiom givesAssertion = dataFactory.getOWLObjectPropertyAssertionAxiom(givesOutAn, caIndividual, adviceIndividual);  objectAssertion.add(givesAssertion);    manager.addAxioms(filietOntology, objectAssertion); |

After adding the object properties, the complete extracted information can now be finally stored to the ontology. Storing information to the ontology is simple done by calling the function saveFilietOntology. Figure 5‑1 shows a process diagram of how to store information to the ontology.

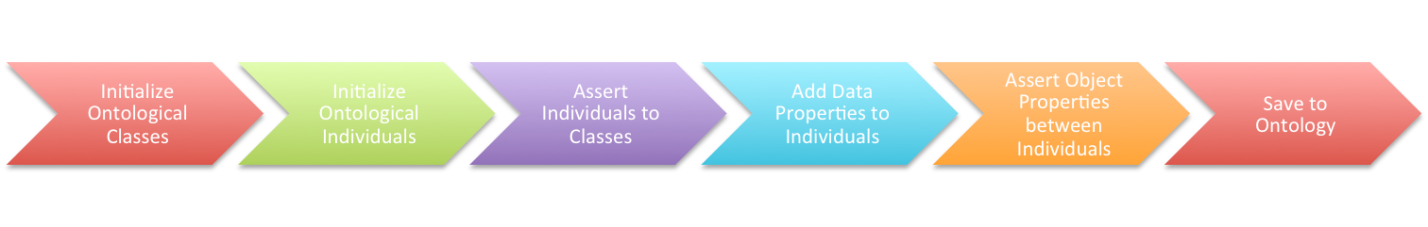


Figure ‑. Process Diagram for Storing Information to the Ontology

### OntologyModule

The OntologyModule class is the main class responsible for working around the storage and verification of the extracted information in the ontology that was designed for the use of the system. It has respective functions for the different pre-requisite steps that shall be taken before actually accessing and modifying the contents of the ontology. There are respective functions for loading, saving and removing the ontology from its manager. Also, general-purpose functions were included to streamline the process of verifying the information to the ontology. These functions include a categorized tweet information viewer and a data property value viewer.

To store information into the ontology, certain classes have to be initialized so that they can be manipulated within the module.

Code Listing ‑. OntologyModule Initialization

|  |
| --- |
| // Classes for containing the extracted information per category  CallForHelpTweet oCH = new CallForHelpTweet();  CasualtiesAndDamageTweet oCD = new CasualtiesAndDamageTweet();  CautionAndAdviceTweet oCA = new CautionAndAdviceTweet();  DonationTweet oD = new DonationTweet();  // Class for actually initializing the module  OntologyModule oModule = new OntologyModule(); |

After initializing the necessary classes, the extracted information is stored into the new initialized classes based on its given category. For instance, let’s take the information that was extracted from a tweet that was categorized to be a Caution and Advice tweet. A sample code listing below shows how to temporarily store the extracted information into its respective categorized tweet class.

Code Listing ‑. Sample Initialization of Categorized Tweet Information Instance

|  |
| --- |
| // SAMPLE INITIALIZATION FOR CAUTION AND ADVICE REPORTS  oCA.setTweetHandle("theonlykyleeeee");  oCA.setTweetContent(":(( RT WARNING! Baha sa Guadalupe!");  oCA.setTweetGeoLocation("10.00000121, 145.345300023");  oCA.setLocationInTweet("Guadalupe");  oCA.setTweetTimestamp("12/27/2014:00:13:67:40");  oCA.setTweetDate("December 27, 2014");  oCA.setTweetAdvice("WARNING! Baha sa Guadalupe!"); |

After temporarily storing the extracted information, actual storage of the information to the ontology now follows. Storing the extracted information is a fairly simple process; as it would only require calling one function and requires only one input. The catch, though, is that there are different methods for certain categories of tweet. Also, if you would want to verify if the storage process has been successful, you could do a rough view of the contents of the ontology just by calling a simple view method.

Code Listing ‑. Storing Information in the Ontology

|  |
| --- |
| try {  oModule.loadOntology();  // Permanently store information to the ontology  oModule.addCautionAndAdviceReport(oCA);  oModule.addCasualtiesAndDamageReport(oCD);  oModule.addDonationReport(oD);  oModule.addCallForHelpReport(oCH);  // View the contents of the ontology  oModule.displayStoredTweets();  oModule.removeOntologyFromManager();  } catch (OWLOntologyCreationException e) {  e.printStackTrace();  } |

In developing the OntologyModule class, there were some issues encountered with its actual implementation because of a number of reasons. First, the documentation that came with the API was not that comprehensive and streamlined in a way that there were actually no complete descriptions about the different methods that can be utilized; though, there were code samples that were confusing. With this type of documentation, there was difficulty in customizing the implementation of the different methods within the API to suit the needs and requirements of the system. Also, debugging/testing of the module was difficult because there was no complete reference for the actual functions of each method including its parameters and its outputs. Manipulation and modification of the actual ontology is a challenge because two different tools used. It is imperative to check if the changes are properly reflected in order for the module to function properly. If there are changes to the ontology, use Protegé first to fix the structure of the ontology and the different dependencies that might be affected upon performing the changes and then, modify the Java code so that there will be seamless interaction between the system and the ontology behind it. In addition, another issue that has benn found with the structure of the ontology is that the actual structure of the tweets, specifically the fields/data properties, is that they are not closely coupled with the actual structure of the tweets leading to some extracted information not being properly stored in the ontology.

### OntologyRetriever

The OntologyRetriever class is the main class responsible for working around the retrieval of the extracted information in the ontology that was designed for the use of the system. It has respective functions for the different pre-requisite steps that shall be taken before actually accessing and modifying the contents of the ontology. There are respective functions for loading and removing the ontology from its manager. Also, general-purpose functions were included to streamline the process of retrieving the information to the ontology. These functions include a categorized tweet information retriever and a data property value retriever. Also, category-specific functions were included to properly organize the information that was retrieved. These functions include a constructor method for retrieved Caution and Advice, Casualties and Damage, Call for Help, and Donation tweets, which just make each of the set of extracted information an instance of each of the classes representing the mentioned categories.

To retrieve the information from the ontology, one class has to be initialized so that other modules can manipulate it.

Code Listing ‑. OntologyRetriever Initialization

|  |
| --- |
| OntologyRetriever or = new OntologyRetriever(); |

After initializing the needed class, actual retrieval of the information from the ontology now follows. Retrieving the information stored in the ontology is a fairly simple process, as it would only require calling one function and requires no input. A variable of the type RetrievedTweetis needed to contain the information that will be returned by the retrieval function of the module. The variable should be of type RetrievedTweet because the retrieval function, logically, returns four ArrayLists for the different categories of the information.

Code Listing ‑. RetrievedTweet Class Structure

|  |
| --- |
| public class RetrievedTweet {  ArrayList<CallForHelpTweet> retrievedCFHTweets;  ArrayList<CasualtiesAndDamageTweet> retrievedCADTweets;  ArrayList<CautionAndAdviceTweet> retrievedCATweets;  ArrayList<DonationTweet> retrievedDTweets;    ......  } |

Code Listing ‑. OntologyRetriever Execution

|  |
| --- |
| try {  or.loadOntology();  RetrievedTweet rt = or.getStoredTweets();  or.removeOntologyFromManager();  } catch (Exception e) {  e.printStackTrace();  } |

In developing the OntologyRetriever class, almost the same issues as the ones mentioned in the OntologyModule were encountered. First, the documentation that came with the API was not that comprehensive and streamlined. With this type of documentation, there is difficulty in customizing the implementation of the different methods that are concerned with ontology retrieval within the API to suit the needs and requirements of the system’s retrieval method. Debugging/testing the module is hard because there is no complete reference for the actual output of each method including the effects of combining different methods to achieve a certain output from the ontology. Second, there have been some difficulties in structuring and implementing the actual retrieval method code because there is a need to look at the initial contents of the ontology and its actual relations and structures to be able translate the logical connections and relations between the instances stored in the ontology to a physical working code that will facilitate the exchange of information between the ontology and system. There should be a way to remember the specific the relational connections between the instances to and “simulate” them in order to reverse the process of adding/storing to the ontology; thus, essentially, enabling a retrieval process. Also, with this, essentially, the same difficulty of working with two separate tools was encountered.

# Results and Observation

This chapter presents the results of the various experiments conducted in the duration of the research. It details the observations as well as if the objectives were achieved.

## Evaluation of Word Feature Set

The objective of this experiment is to determine which feature set would render optimal results for classification as well as if they could be adapted to other disaster datasets.

### Methodology

The word features are extracted from the two datasets, the Mario and Ruby datasets. The Ruby dataset contains 2583 instances, which is composed of 1279 (CA), 245 (CD), 9 (CH), 37 (D), and 1013 (0); while the Mario dataset contains 1365 instances and is composed of 651 (CA), 99 (CD), 58 (CH), 47 (D) and 510 (0). To create the list of word features, the instances are grouped into their respective categories per dataset. From each category per dataset, the distinct words, excluding stop words (refer to Appendix D), accented characters (i.e. Ã), and links (i.e. http://t.cÃ?Â¢Ã¢â??Â¬Ã?Â¦) are collected. From the collected words, the TFIDF score is then computed per category and the top 10%, 20%, and 30% highest scoring TFIDF words will serve as the word features. To further clean the list of word features, the irrelevant words are removed (refer to Appendix E). This is done by having the proponents vote whether the word is deemed relevant or not. At this point, 24 word feature sets are created. Table 6‑1 lists these 24 word feature sets whereas Appendix F lists their contents. A combined feature set is also created by gathering the common words in all categories per dataset. Appendix G lists the content of the combined feature set.

Table ‑. Listing of Word Feature Sets

|  |  |  |  |
| --- | --- | --- | --- |
| Mario-CA-10 | Mario-CH-10 | Ruby-CA-10 | Ruby-CH-10 |
| Mario-CA-20 | Mario-CH-20 | Ruby-CA-20 | Ruby-CH-20 |
| Mario-CA-30 | Mario-CH-30 | Ruby-CA-30 | Ruby-CH-30 |
| Mario-CD-10 | Mario-D-10 | Ruby-CD-10 | Ruby-D-10 |
| Mario-CD-20 | Mario-D-20 | Ruby-CD-20 | Ruby-D-20 |
| Mario-CD-30 | Mario-D-30 | Ruby-CD-30 | Ruby-D-30 |

### Experiments

The word features are tested in various experiments in order to test which one is the most optimal. The algorithms used for testing are Naive Bayes, Bayesian Network, Random Forest, J48 Decision Tree, and k-Nearest Neighbor where k = {3, 5, 7}. They are tested with a 10 cross-fold validation and split testing (60% training – 40% testing) and the performance is measured with precision, recall, f-measure, and kappa statistics.

#### Whole Dataset using Combined Word Features

##### Mario Dataset

Table 6‑2 shows the results that were gathered from performing a classification test using 10 cross-fold validation on the Mario dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the said dataset.

Table ‑. Mario Dataset using top n% Mario-Combined Features (10-cross fold validation)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.79 | 0.777 | 0.776 | 0.6424 | 0.761 | 0.75 | 0.751 | 0.5976 | 0.817 | 0.804 | 0.804 | 0.6848 |
| **RF** | 0.816 | 0.813 | 0.813 | 0.7011 | 0.774 | 0.769 | 0.77 | 0.6281 | 0.857 | 0.846 | 0.847 | 0.7533 |
| **KNN3** | 0.756 | 0.749 | 0.75 | 0.6018 | 0.739 | 0.735 | 0.736 | 0.5714 | 0.798 | 0.778 | 0.778 | 0.6457 |
| **KNN5** | 0.771 | 0.758 | 0.758 | 0.6114 | 0.732 | 0.725 | 0.726 | 0.5515 | 0.813 | 0.78 | 0.78 | 0.6475 |
| **KNN7** | 0.781 | 0.761 | 0.761 | 0.6151 | 0.726 | 0.716 | 0.716 | 0.532 | 0.803 | 0.764 | 0.762 | 0.6203 |
| **NB** | 0.73 | 0.729 | 0.728 | 0.5616 | 0.742 | 0.728 | 0.727 | 0.5712 | 0.758 | 0.741 | 0.742 | 0.5923 |
| **BN** | 0.726 | 0.716 | 0.714 | 0.5308 | 0.735 | 0.729 | 0.728 | 0.5552 | 0.776 | 0.765 | 0.764 | 0.6179 |

Table 6‑3 shows the results that were gathered from performing a classification test using split testing on the Mario dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the said dataset

Table ‑. Mario Dataset using top n% Mario-Combined Features (Split Testing)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.809 | 0.797 | 0.794 | 0.6727 | 0.755 | 0.738 | 0.737 | 0.5749 | 0.817 | 0.804 | 0.804 | 0.6848 |
| **RF** | 0.812 | 0.806 | 0.805 | 0.6864 | 0.761 | 0.751 | 0.752 | 0.5932 | 0.857 | 0.846 | 0.847 | 0.7533 |
| **KNN3** | 0.781 | 0.769 | 0.769 | 0.63 | 0.708 | 0.705 | 0.704 | 0.5176 | 0.798 | 0.778 | 0.778 | 0.6457 |
| **KNN5** | 0.786 | 0.769 | 0.766 | 0.6253 | 0.721 | 0.705 | 0.706 | 0.51 | 0.813 | 0.78 | 0.78 | 0.6475 |
| **KNN7** | 0.784 | 0.767 | 0.765 | 0.6209 | 0.694 | 0.676 | 0.673 | 0.4574 | 0.803 | 0.764 | 0.762 | 0.6203 |
| **NB** | 0.719 | 0.72 | 0.719 | 0.5463 | 0.739 | 0.729 | 0.728 | 0.5718 | 0.758 | 0.741 | 0.742 | 0.5923 |
| **BN** | 0.71 | 0.689 | 0.678 | 0.4751 | 0.722 | 0.698 | 0.69 | 0.4878 | 0.776 | 0.765 | 0.764 | 0.6179 |

##### Ruby Dataset

Table 6‑4 shows the results that were gathered from performing a classification test using 10 cross-fold validation on the Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the said dataset.

Table ‑. Ruby Dataset using top n% Ruby-Combined Features (10 cross-fold validation)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.901 | 0.899 | 0.898 | 0.8279 | 0.914 | 0.914 | 0.913 | 0.8527 | 0.927 | 0.926 | 0.925 | 0.8746 |
| **RF** | 0.917 | 0.916 | 0.916 | 0.8579 | 0.933 | 0.933 | 0.933 | 0.8862 | 0.952 | 0.951 | 0.951 | 0.9165 |
| **KNN3** | 0.901 | 0.901 | 0.899 | 0.8301 | 0.918 | 0.917 | 0.916 | 0.8572 | 0.928 | 0.922 | 0.921 | 0.8666 |
| **KNN5** | 0.893 | 0.89 | 0.887 | 0.8113 | 0.917 | 0.916 | 0.915 | 0.8559 | 0.927 | 0.921 | 0.92 | 0.8652 |
| **KNN7** | 0.869 | 0.869 | 0.865 | 0.7749 | 0.917 | 0.916 | 0.915 | 0.8559 | 0.927 | 0.921 | 0.92 | 0.8652 |
| **NB** | 0.866 | 0.865 | 0.864 | 0.7687 | 0.876 | 0.868 | 0.87 | 0.7807 | 0.9 | 0.898 | 0.898 | 0.8286 |
| **BN** | 0.858 | 0.858 | 0.856 | 0.756 | 0.887 | 0.887 | 0.887 | 0.8082 | 0.903 | 0.903 | 0.902 | 0.8355 |

Table 6‑5 shows the results that were gathered from performing a classification test using spit testing on the Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the said dataset.

Table ‑. Ruby Dataset using top n% Ruby-Combined Features (Split Testing)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.889 | 0.885 | 0.883 | 0.8045 | 0.914 | 0.914 | 0.913 | 0.8527 | 0.912 | 0.916 | 0.913 | 0.8577 |
| **RF** | 0.909 | 0.911 | 0.909 | 0.8514 | 0.933 | 0.933 | 0.933 | 0.8862 | 0.936 | 0.939 | 0.936 | 0.8975 |
| **KNN3** | 0.879 | 0.88 | 0.878 | 0.7984 | 0.918 | 0.917 | 0.916 | 0.8572 | 0.91 | 0.906 | 0.902 | 0.8398 |
| **KNN5** | 0.862 | 0.864 | 0.861 | 0.7699 | 0.907 | 0.904 | 0.902 | 0.8343 | 0.898 | 0.889 | 0.885 | 0.8102 |
| **KNN7** | 0.86 | 0.862 | 0.857 | 0.7643 | 0.888 | 0.888 | 0.884 | 0.8057 | 0.864 | 0.853 | 0.844 | 0.7461 |
| **NB** | 0.86 | 0.859 | 0.857 | 0.7587 | 0.876 | 0.868 | 0.87 | 0.7807 | 0.91 | 0.908 | 0.908 | 0.8474 |
| **BN** | 0.846 | 0.85 | 0.846 | 0.7436 | 0.887 | 0.887 | 0.887 | 0.8082 | 0.879 | 0.884 | 0.881 | 0.8034 |

#### Dataset A using Combined Word Features of Dataset B (and vice-versa)

##### Mario Dataset

Table 6‑4 shows the results that were gathered from performing a classification test using 10 cross-fold validation on the Mario dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the Ruby dataset.

Table ‑. Mario Dataset using top n% Ruby-Combined features (10 cross-fold validation)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.711 | 0.718 | 0.71 | 0.5368 | 0.78 | 0.773 | 0.772 | 0.6345 | 0.803 | 0.793 | 0.794 | 0.6691 |
| **RF** | 0.748 | 0.753 | 0.749 | 0.601 | 0.804 | 0.8 | 0.8 | 0.68 | 0.841 | 0.837 | 0.838 | 0.738 |
| **KNN3** | 0.706 | 0.711 | 0.707 | 0.5288 | 0.766 | 0.763 | 0.763 | 0.62 | 0.771 | 0.768 | 0.768 | 0.6278 |
| **KNN5** | 0.682 | 0.684 | 0.681 | 0.4818 | 0.741 | 0.736 | 0.736 | 0.5747 | 0.746 | 0.741 | 0.74 | 0.5784 |
| **KNN7** | 0.666 | 0.666 | 0.662 | 0.4498 | 0.724 | 0.72 | 0.718 | 0.5448 | 0.732 | 0.722 | 0.722 | 0.5478 |
| **NB** | 0.662 | 0.678 | 0.669 | 0.4728 | 0.75 | 0.738 | 0.736 | 0.583 | 0.747 | 0.732 | 0.731 | 0.574 |
| **BN** | 0.666 | 0.687 | 0.667 | 0.4691 | 0.746 | 0.737 | 0.734 | 0.5741 | 0.754 | 0.742 | 0.741 | 0.5862 |

Table 6‑7 shows the results that were gathered from performing a classification test using split testing on the Mario dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the Ruby dataset.

Table ‑. Mario Dataset using top n% Ruby-Combined features (Split Testing)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.717 | 0.718 | 0.712 | 0.532 | 0.762 | 0.756 | 0.757 | 0.6072 | 0.795 | 0.786 | 0.785 | 0.6568 |
| **RF** | 0.735 | 0.74 | 0.735 | 0.5756 | 0.766 | 0.766 | 0.765 | 0.6232 | 0.819 | 0.813 | 0.813 | 0.6994 |
| **KNN3** | 0.7 | 0.694 | 0.695 | 0.5048 | 0.745 | 0.744 | 0.743 | 0.5864 | 0.781 | 0.78 | 0.78 | 0.645 |
| **KNN5** | 0.691 | 0.683 | 0.685 | 0.4852 | 0.739 | 0.734 | 0.735 | 0.5694 | 0.769 | 0.764 | 0.763 | 0.6124 |
| **KNN7** | 0.691 | 0.683 | 0.685 | 0.4852 | 0.746 | 0.734 | 0.735 | 0.5668 | 0.787 | 0.782 | 0.782 | 0.6437 |
| **NB** | 0.702 | 0.712 | 0.702 | 0.5267 | 0.769 | 0.756 | 0.756 | 0.6136 | 0.767 | 0.755 | 0.755 | 0.61 |
| **BN** | 0.629 | 0.668 | 0.641 | 0.4275 | 0.741 | 0.736 | 0.735 | 0.5704 | 0.757 | 0.749 | 0.748 | 0.5955 |

##### Ruby Dataset

Table 6‑8 shows the results that were gathered from performing a classification test using 10 cross-fold validation on the Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from Mario dataset.

Table ‑. Ruby Dataset using top n% Mario-Combined features (10 cross-fold validation)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.892 | 0.898 | 0.893 | 0.8255 | 0.872 | 0.874 | 0.872 | 0.7858 | 0.906 | 0.909 | 0.907 | 0.8455 |
| **RF** | 0.917 | 0.918 | 0.917 | 0.8607 | 0.895 | 0.895 | 0.895 | 0.8223 | 0.939 | 0.939 | 0.938 | 0.8968 |
| **KNN3** | 0.89 | 0.891 | 0.89 | 0.8154 | 0.881 | 0.882 | 0.88 | 0.7987 | 0.912 | 0.912 | 0.911 | 0.8496 |
| **KNN5** | 0.88 | 0.883 | 0.88 | 0.8003 | 0.871 | 0.872 | 0.869 | 0.7811 | 0.898 | 0.901 | 0.898 | 0.8306 |
| **KNN7** | 0.859 | 0.864 | 0.861 | 0.7694 | 0.85 | 0.855 | 0.851 | 0.751 | 0.876 | 0.88 | 0.877 | 0.794 |
| **NB** | 0.841 | 0.843 | 0.841 | 0.732 | 0.831 | 0.817 | 0.822 | 0.6973 | 0.877 | 0.866 | 0.87 | 0.7789 |
| **BN** | 0.845 | 0.847 | 0.844 | 0.738 | 0.826 | 0.82 | 0.821 | 0.699 | 0.875 | 0.867 | 0.87 | 0.7793 |

Table 6‑9 shows the results that were gathered from performing a classification test using 10 cross-fold validation on the Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from Mario dataset.

Table ‑. Ruby Dataset using top n% Mario-Combined features (Split Testing)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.868 | 0.882 | 0.875 | 0.8018 | 0.911 | 0.911 | 0.907 | 0.8491 | 0.893 | 0.898 | 0.895 | 0.8283 |
| **RF** | 0.887 | 0.899 | 0.893 | 0.8307 | 0.933 | 0.933 | 0.932 | 0.8874 | 0.922 | 0.923 | 0.922 | 0.87 |
| **KNN3** | 0.884 | 0.89 | 0.885 | 0.8139 | 0.901 | 0.905 | 0.902 | 0.8391 | 0.891 | 0.895 | 0.893 | 0.8244 |
| **KNN5** | 0.884 | 0.887 | 0.881 | 0.8088 | 0.884 | 0.887 | 0.882 | 0.8069 | 0.878 | 0.883 | 0.88 | 0.8023 |
| **KNN7** | 0.862 | 0.864 | 0.859 | 0.7716 | 0.866 | 0.864 | 0.856 | 0.7659 | 0.862 | 0.867 | 0.863 | 0.7755 |
| **NB** | 0.847 | 0.854 | 0.849 | 0.751 | 0.896 | 0.893 | 0.894 | 0.8225 | 0.879 | 0.876 | 0.877 | 0.7944 |
| **BN** | 0.829 | 0.832 | 0.822 | 0.7072 | 0.865 | 0.87 | 0.867 | 0.7801 | 0.856 | 0.856 | 0.855 | 0.7601 |

Based on the four tables presented in the two experiements (Table 6‑2; Table 6‑4; Table 6‑6; Table 6‑8), it can be seen that there has been a significant difference between the performances of the two feature sets when used to classify the instances from other datasets, that is, when the Mario feature sets were used in the Ruby dataset and when the Ruby feature sets were used in the Mario dataset. Also, it can be seen that the algorithm that produced the best results was the Random Forest classification algorithm. The latter experimental setup did not fare well as compared to the other experimental setup.  When the Ruby feature sets were used in the classification of the Mario dataset, it resulted to a significantly low performance rating of 0.749 F-measure, 0.8 F-measure and 0.838 F-measure on the top 10%, 20% and 30% features, respectively. On the other hand, when the Mario feature sets were used in the classification of the Ruby dataset, it resulted to a significantly high performance rating of 0.917 F-measure, 0.895 F-measure and 0.938 F-measure on the top 10%, 20% and 30% features, respectively. From these results, it can be said that the Ruby feature sets didn’t perform well on classifying the instances of the Mario dataset and that the Mario feature sets performed well on classifying the instances of the Ruby dataset. This phenomenon, specifically on the Ruby feature to Mario dataset, could be attributed to how the Ruby feature sets were extracted from the datasets. The Ruby dataset has greater number of instances than that of the Mario dataset but the large number of Ruby instances are attributed to the presence of retweets. A huge chunk of the Ruby dataset contained retweets. On the other hand, the Mario dataset had more unique instances as compared to the Ruby dataset. With this, more unique and diverse word features were extracted from the Mario dataset than those from the Ruby dataset. The feature sets from Ruby are not adaptable to other datasets by not being able to cover the other instances present in other datasets, which in this case, is the Mario dataset. The results of the split testing (Table 6‑3; Table 6‑5; Table 6‑7; Table 6‑9) shows only a slight decrease in performance.

#### Mario-Ruby (Combined) Dataset using Combined Word Features of One Dataset

##### Mario Dataset Features

Table 6‑6 shows the results that were gathered from performing a classification test on the Combined Mario-Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the Mario dataset.

Table ‑. Mario-Ruby Dataset using top n% Mario-Combined features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.818 | 0.81 | 0.81 | 0.6952 | 0.804 | 0.796 | 0.797 | 0.6718 | 0.828 | 0.823 | 0.822 | 0.7142 |
| **RF** | 0.842 | 0.836 | 0.836 | 0.737 | 0.846 | 0.844 | 0.844 | 0.7475 | 0.874 | 0.866 | 0.866 | 0.784 |
| **KNN3** | 0.779 | 0.774 | 0.774 | 0.6358 | 0.787 | 0.784 | 0.784 | 0.6489 | 0.828 | 0.805 | 0.806 | 0.6876 |
| **KNN5** | 0.789 | 0.774 | 0.773 | 0.6359 | 0.775 | 0.769 | 0.769 | 0.6218 | 0.819 | 0.786 | 0.786 | 0.6565 |
| **KNN7** | 0.794 | 0.766 | 0.765 | 0.6228 | 0.786 | 0.776 | 0.775 | 0.6311 | 0.813 | 0.763 | 0.762 | 0.6168 |
| **NB** | 0.746 | 0.745 | 0.745 | 0.5902 | 0.771 | 0.752 | 0.75 | 0.6104 | 0.774 | 0.753 | 0.753 | 0.613 |
| **BN** | 0.753 | 0.748 | 0.747 | 0.5888 | 0.78 | 0.77 | 0.769 | 0.6308 | 0.786 | 0.774 | 0.773 | 0.6393 |

Based on the results from the table shown above, there is an increase in performance as the percentage of top scoring word features extracted from the combined Mario datasets is increased for most of the classification algorithms. However, the KNN7 classification algorithm had a different result as the percentage increased from 20% to 30%. It had a significant decrease in performance from a 0.775 F-measure rating to a 0.762 F-measure rating. Note that there is a slight decline in performance for J48 from 10% to 20% but it increased as the percentage goes to 30%. Across all top percentage setups, the Random Forest classification algorithm garnered the best result based on all performance measures used.

##### Ruby Dataset Features

Table 6‑7 shows the results that were gathered from performing a classification test on the Combined Mario-Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the Ruby dataset.

Table ‑. Mario-Ruby Dataset using top n% Ruby-Combined features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.92 | 0.921 | 0.919 | 0.8652 | 0.92 | 0.919 | 0.918 | 0.8617 | 0.925 | 0.924 | 0.923 | 0.8709 |
| **RF** | 0.947 | 0.947 | 0.947 | 0.9107 | 0.944 | 0.943 | 0.943 | 0.9043 | 0.958 | 0.957 | 0.956 | 0.9271 |
| **KNN3** | 0.925 | 0.924 | 0.923 | 0.8695 | 0.924 | 0.923 | 0.922 | 0.8691 | 0.936 | 0.93 | 0.929 | 0.8806 |
| **KNN5** | 0.917 | 0.916 | 0.914 | 0.8561 | 0.903 | 0.904 | 0.901 | 0.8356 | 0.922 | 0.91 | 0.909 | 0.8457 |
| **KNN7** | 0.901 | 0.903 | 0.9 | 0.8342 | 0.891 | 0.892 | 0.887 | 0.8125 | 0.902 | 0.888 | 0.884 | 0.8072 |
| **NB** | 0.892 | 0.891 | 0.891 | 0.8148 | 0.889 | 0.883 | 0.885 | 0.8056 | 0.911 | 0.904 | 0.905 | 0.8402 |
| **BN** | 0.901 | 0.902 | 0.901 | 0.8325 | 0.891 | 0.891 | 0.89 | 0.8153 | 0.91 | 0.908 | 0.908 | 0.8452 |

Based on the results from the table shown above, there is an increase in performance as the percentage of top scoring word features extracted from the combined Mario datasets is increased for most of the classification algorithms. Note that there is a slight decline in performance in most of these algorithms from 10% to 20% but it increased as the percentage goes to 30%. However, the KNN7 classification algorithm had a different result as the percentage increased from 10% to 20% to 30%. It had a significant decrease in performance from a 0.9 F-measure rating to a 0.887 F-measure and to a 0.884 F-measure rating, respectively. Across all top percentage setups, the Random Forest classification algorithm garnered the best result based on all performance measures used.

#### Dataset A or B using Combined Word Features of Datasets A and B

##### Mario Dataset

Table 6‑8 shows the results that were gathered from performing a classification test on the Mario dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the Mario and Ruby datasets.

Table ‑. Mario Dataset using top n% Mario and Ruby-Combined features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.818 | 0.810 | 0.810 | 0.6952 | 0.804 | 0.796 | 0.797 | 0.6718 | 0.828 | 0.823 | 0.822 | 0.7142 |
| **RF** | 0.827 | 0.824 | 0.824 | 0.7165 | 0.838 | 0.837 | 0.836 | 0.7359 | 0.861 | 0.857 | 0.857 | 0.7699 |
| **KNN3** | 0.779 | 0.774 | 0.774 | 0.6358 | 0.787 | 0.784 | 0.784 | 0.6489 | 0.828 | 0.805 | 0.806 | 0.6876 |
| **KNN5** | 0.789 | 0.774 | 0.773 | 0.6359 | 0.775 | 0.769 | 0.769 | 0.6218 | 0.819 | 0.786 | 0.786 | 0.6565 |
| **KNN7** | 0.794 | 0.766 | 0.765 | 0.6228 | 0.786 | 0.776 | 0.775 | 0.6311 | 0.813 | 0.763 | 0.762 | 0.6168 |
| **NB** | 0.746 | 0.745 | 0.745 | 0.5902 | 0.771 | 0.752 | 0.751 | 0.6104 | 0.774 | 0.753 | 0.753 | 0.613 |
| **BN** | 0.753 | 0.748 | 0.747 | 0.5888 | 0.780 | 0.770 | 0.769 | 0.6308 | 0.786 | 0.774 | 0.773 | 0.6393 |

Based on the results from the three tables, there is an increase in performance as the percentage of top scoring word features extracted from the two datasets is increased for most of the classification algorithms. Note that there is a slight decline in performance for J48 and KNN5 from 10% to 20% but it increased. However, the performance of the top 30% word features for these two algorithms are significantly greater than that of top 10% word features. Across all top percentage setups, Random Forest garnered the best result based on all performance measures used.

##### Ruby Dataset

Table 6‑9 shows the results that were gathered from performing a classification test on the Ruby dataset using different classification algorithms with the combined top 10%, 20%, and 30% of the top word features extracted from the Mario and Ruby datasets.

Table ‑. Ruby Dataset using top n% Mario and Ruby-Combined features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.920 | 0.921 | 0.919 | 0.8652 | 0.920 | 0.919 | 0.918 | 0.8617 | 0.925 | 0.924 | 0.923 | 0.8709 |
| **RF** | 0.944 | 0.944 | 0.944 | 0.9054 | 0.938 | 0.938 | 0.938 | 0.8948 | 0.954 | 0.954 | 0.953 | 0.9211 |
| **KNN3** | 0.925 | 0.924 | 0.923 | 0.8695 | 0.924 | 0.923 | 0.922 | 0.8691 | 0.936 | 0.930 | 0.929 | 0.8806 |
| **KNN5** | 0.917 | 0.916 | 0.914 | 0.8561 | 0.903 | 0.904 | 0.901 | 0.8356 | 0.922 | 0.910 | 0.909 | 0.8457 |
| **KNN7** | 0.901 | 0.903 | 0.900 | 0.8342 | 0.891 | 0.892 | 0.887 | 0.8125 | 0.902 | 0.888 | 0.884 | 0.8072 |
| **NB** | 0.892 | 0.891 | 0.891 | 0.8148 | 0.889 | 0.883 | 0.885 | 0.8056 | 0.911 | 0.904 | 0.905 | 0.8402 |
| **BN** | 0.901 | 0.902 | 0.901 | 0.8325 | 0.891 | 0.891 | 0.890 | 0.8153 | 0.910 | 0.908 | 0.908 | 0.8452 |

Based on the tables presented above, it can be seen that as the percentage of top feature words increases, the performance of the classification algorithms also increases. However, in this particular dataset, as the top percentages were increased from 10% to 20%, there is a slight decrease in the performance of the different classification algorithms. In addition, as seen in the tables, it can be said that the classification algorithm that performed the best is the Random Forest classification algorithm with F-measure performance ratings above 0.930 across all top percentage setups; 10%, 20%, and 30% with 0.944, 0.938, and 0.953, respectively.

Table 6‑8 and Table 6‑9 show an increase performance from Mario or Ruby feature sets alone, Table 6‑5 and Table 6‑4, respectively. This is because the cases that the Mario features could not classify correctly is now being detected by the Ruby features. The Ruby dataset is very different from Mario dataset. In the Ruby dataset, most of the tweets is about typhoon signal, status, and suspension of classes. However in the Mario dataset, most of the tweets are about road status. The extracted features are very different from each other. This helps because the Ruby feature set supplemented the Mario feature set. This is shown through the increase in performance for both of the dataset.

#### Balance Dataset – Balance Feature Set

##### Mario

Table ‑. Balance Dataset - Mario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Mario Dataset** | | | |
| **P** | **R** | **FM** | **K** |
| **J48** | 0.642 | 0.626 | 0.631 | 0.5319 |
| **RF** | 0.714 | 0.706 | 0.709 | 0.633 |
| **KNN3** | 0.685 | 0.634 | 0.65 | 0.5426 |
| **KNN5** | 0.676 | 0.621 | 0.637 | 0.5266 |
| **KNN7** | 0.663 | 0.6 | 0.618 | 0.5 |
| **NB** | 0.698 | 0.651 | 0.669 | 0.5638 |
| **BN** | 0.587 | 0.54 | 0.551 | 0.4255 |

Table 6‑14 above presents the results that were gathered when an experiment was done on a balanced Mario Dataset (all of the categories have the same number of instances to be categorized) using a balanced Mario feature set (all of the categories have the same number of features extracted from them). This dataset has a total of 235 instances because 47 instances were taken from each of the categories (47 was least number among the categories and it came from the D category - the least among categories) and a total of 111 features because 27 features were extracted from each of the categories (27 was the least number of instances and it came from the CD category - the least among the categories). Moving on, as seen in the table presented above, it can be said that given a limited number of instances and features for the classification yielded relatively low scores as compared to the previous experiments (those with large number of features and instances). This phenomenon can be attributed to the fact that the limited number of features is not enough to cover the diversities in the content of the tweets even though it is of a limited number. The classification heavily relies on the features; thus, if it only has a limited “reference” for the classification, it would definitely lead to a significant decrease in the performance of the classifier, in general. However, it can be seen also, that the result produced by this experiment is satisfactory (it is greater than 50%). This can be attributed to the fact that there were only few instances to categorize; thus, there are not much instances for the classifier to fail in the classification.

##### Ruby

Table ‑. Balance Dataset - Ruby

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Ruby Dataset** | | | |
| **P** | **R** | **FM** | **K** |
| **J48** | 0.693 | 0.662 | 0.671 | 0.5509 |
| **RF** | 0.729 | 0.69 | 0.701 | 0.5864 |
| **KNN3** | 0.722 | 0.614 | 0.636 | 0.4879 |
| **KNN5** | 0.774 | 0.641 | 0.652 | 0.5271 |
| **KNN7** | 0.729 | 0.607 | 0.622 | 0.4817 |
| **NB** | 0.779 | 0.71 | 0,723 | 0.6165 |
| **BN** | 0.55 | 0.497 | 0.467 | 0.3196 |

As seen in Table 6‑15, it can be said that given a limited number of instances and features for the classification yielded relatively low scores as compared to the previous experiments (those with large number of features and instances). This phenomenon can be attributed to the fact that the limited number of features is not enough to cover the diversities in the content of the tweets even though it is of a limited number. The classification heavily relies on the features; thus, if it only has a limited “reference” for the classification, it would definitely lead to a significant decrease in the performance of the classifier, in general. However, it can be seen also, that the result produced by this experiment is satisfactory (it is greater than 55%) as compared to the experiment with the balanced Mario features and dataset. This can be attributed to the fact that there were more instances to categorize and more features to be used; thus, enabling the classifier to support more variations in the content of the tweets in the dataset.

## Evaluation of Classifiers

The objective of the experiment is to determine which classifier will yield the best result: a single classifier or a cascading classifier. A single classifier is a classifier in which it will classify the instance into all of the category. A cascading classifier is a classifier which is made up of a series of binary classifiers. Each of the classifier will only categorize the assigned category and others (O).

### Methodology

The Mario and Ruby dataset was used to test the classifier. The Mario dataset contains 1365 instances and is composed of 651 (CA), 99 (CD), 58 (CH), 47 (D), and 510 (O), while the Ruby dataset contains 2583 instances and is composed of 1279 (CA), 245 (CD), 9 (CH), 37 (D), and 1013 (O). Both of the classifier will use random forest algorithm.

The single classifier uses the top 30% Mario feature set as its features. Then, the classifier is tested on both Mario and Ruby dataset.

The cascading classifier is consist 4 binary classifier: CA classifier, CD classifier, CH classifier, and D classifier. Each classifier has its own set of features. For example, the CA classifier has only CA features, CD classifier has CD features, CH classifier has CH features and D classifier has D features. Then, the classifier are tested on datasets that contains only their category and O. Table 6‑10 shows the number of instances in the datasets used in this experiment.

Table ‑. Summary of Cascading Classifier Dataset Contents

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Mario Dataset** | | | | **Ruby Dataset** | | | |
| CA | 651 | CH | 99 | CA | 1279 | CH | 245 |
| O | 651 | O | 99 | O | 1013 | O | 245 |
| Total | 1302 | Total | 198 | Total | 2292 | Total | 490 |
| CD | 58 | D | 47 | CD | 9 | D | 37 |
| O | 58 | O | 47 | O | 9 | O | 37 |
| Total | 116 | Total | 94 | Total | 18 | Total | 74 |

### Multiple Binary Classifier

#### Mario Dataset

##### CA Category

Table 6‑11 shows the results that were gathered from performing a classification test on the Mario CA dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted from the said dataset.

Table ‑. Mario Dataset (CA) using top n% Mario-CA features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.818 | 0.801 | 0.801 | 0.6057 | 0.776 | 0.759 | 0.759 | 0.5224 | 0.837 | 0.821 | 0.821 | 0.6448 |
| **RF** | 0.841 | 0.836 | 0.837 | 0.6714 | 0.769 | 0.763 | 0.764 | 0.5241 | 0.874 | 0.864 | 0.864 | 0.7258 |
| **KNN3** | 0.81 | 0.802 | 0.803 | 0.6041 | 0.73 | 0.736 | 0.736 | 0.4688 | 0.816 | 0.792 | 0.791 | 0.5894 |
| **KNN5** | 0.789 | 0.775 | 0.776 | 0.5536 | 0.738 | 0.732 | 0.733 | 0.463 | 0.826 | 0.788 | 0.787 | 0.5859 |
| **KNN7** | 0.814 | 0.792 | 0.793 | 0.5902 | 0.738 | 0.731 | 0.732 | 0.4618 | 0.837 | 0.79 | 0.788 | 0.5911 |
| **NB** | 0.774 | 0.769 | 0.77 | 0.5367 | 0.774 | 0.769 | 0.77 | 0.5367 | 0.784 | 0.775 | 0.776 | 0.5514 |
| **BN** | 0.771 | 0.761 | 0.761 | 0.5227 | 0.732 | 0.731 | 0.731 | 0.4554 | 0.792 | 0.773 | 0.774 | 0.552 |

##### CD Category

Table 6‑12 shows the results that were gathered from performing a classification test on the Mario CD dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted from the said dataset.

Table ‑. Mario Dataset (CD) using top n% Mario-CD features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.898 | 0.897 | 0.896 | 0.7931 | 0.898 | 0.897 | 0.896 | 0.7931 | 0.926 | 0.922 | 0.922 | 0.8448 |
| **RF** | 0.933 | 0.931 | 0.931 | 0.8621 | 0.946 | 0.940 | 0.939 | 0.8793 | 0.975 | 0.974 | 0.974 | 0.9483 |
| **KNN3** | 0.880 | 0.871 | 0.870 | 0.7414 | 0.908 | 0.888 | 0.887 | 0.7759 | 0.903 | 0.879 | 0.878 | 0.7586 |
| **KNN5** | 0.896 | 0.879 | 0.878 | 0.7586 | 0.897 | 0.871 | 0.868 | 0.7414 | 0.877 | 0.836 | 0.832 | 0.6724 |
| **KNN7** | 0.885 | 0.871 | 0.870 | 0.7414 | 0.872 | 0.853 | 0.852 | 0.7069 | 0.849 | 0.819 | 0.815 | 0.6379 |
| **NB** | 0.922 | 0.914 | 0.913 | 0.8276 | 0.882 | 0.845 | 0.841 | 0.6897 | 0.903 | 0.879 | 0.878 | 0.7586 |
| **BN** | 0.908 | 0.888 | 0.887 | 0.7759 | 0.882 | 0.845 | 0.841 | 0.6897 | 0.892 | 0.862 | 0.859 | 0.7241 |

##### CH Category

Table 6‑13 shows the results that were gathered from performing a classification test on the Mario CH dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted from the said dataset.

Table ‑. Mario Dataset (CH) using top n% Mario-CH features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.825 | 0.813 | 0.811 | 0.6263 | 0.879 | 0.869 | 0.898 | 0.7374 | 0.889 | 0.879 | 0.878 | 0.7576 |
| **RF** | 0.847 | 0.813 | 0.808 | 0.6263 | 0.899 | 0.889 | 0.888 | 0.7778 | 0.903 | 0.894 | 0.893 | 0.7879 |
| **KNN3** | 0.825 | 0.823 | 0.823 | 0.6465 | 0.869 | 0.869 | 0.869 | 0.7374 | 0.915 | 0.909 | 0.909 | 0.8182 |
| **KNN5** | 0.788 | 0.783 | 0.782 | 0.5657 | 0.839 | 0.838 | 0.838 | 0.6768 | 0.893 | 0.884 | 0.883 | 0.7677 |
| **KNN7** | 0.827 | 0.818 | 0.817 | 0.6364 | 0.839 | 0.838 | 0.838 | 0.6768 | 0.899 | 0.884 | 0.883 | 0.7677 |
| **NB** | 0.859 | 0.859 | 0.859 | 0.7172 | 0.903 | 0.899 | 0.899 | 0.798 | 0.907 | 0.904 | 0.904 | 0.8081 |
| **BN** | 0.864 | 0.864 | 0.864 | 0.7273 | 0.89 | 0.889 | 0.889 | 0.7778 | 0.925 | 0.924 | 0.924 | 0.8485 |

##### D Category

Table 6‑14 shows the results that were gathered from performing a classification test on the Mario D dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted for the said dataset.

Table ‑. Mario Dataset (D) using top n% Mario-D features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.892 | 0.862 | 0.859 | 0.7234 | 0.892 | 0.862 | 0.859 | 0.7234 | 0.854 | 0.851 | 0.851 | 0.7021 |
| **RF** | 0.894 | 0.894 | 0.894 | 0.7872 | 0.904 | 0.904 | 0.904 | 0.8085 | 0.947 | 0.947 | 0.947 | 0.8936 |
| **KNN3** | 0.809 | 0.809 | 0.809 | 0.617 | 0.789 | 0.787 | 0.787 | 0.5745 | 0.758 | 0.755 | 0.755 | 0.5106 |
| **KNN5** | 0.872 | 0.872 | 0.872 | 0.7447 | 0.866 | 0.862 | 0.861 | 0.7234 | 0.885 | 0.851 | 0.849 | 0.7021 |
| **KNN7** | 0.814 | 0.809 | 0.808 | 0.617 | 0.778 | 0.777 | 0.776 | 0.5532 | 0.892 | 0.862 | 0.859 | 0.7234 |
| **NB** | 0.905 | 0.894 | 0.893 | 0.7872 | 0.892 | 0.883 | 0.882 | 0.766 | 0.896 | 0.894 | 0.892 | 0.7872 |
| **BN** | 0.704 | 0.702 | 0.702 | 0.4043 | 0.789 | 0.787 | 0.787 | 0.5745 | 0.801 | 0.798 | 0.797 | 0.5957 |

#### Ruby Dataset

##### CA Category

Table 6‑15 shows the results that were gathered from performing a classification test on the Ruby CA dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted for the said dataset.

Table ‑. Ruby Dataset (CA) using top n% Ruby-CA Features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.902 | 0.898 | 0.899 | 0.7957 | 0.897 | 0.897 | 0.897 | 0.7914 | 0.941 | 0.939 | 0.939 | 0.8777 |
| **RF** | 0.931 | 0.931 | 0.931 | 0.8606 | 0.929 | 0.928 | 0.928 | 0.8551 | 0.959 | 0.958 | 0.958 | 0.9146 |
| **KNN3** | 0.928 | 0.928 | 0.928 | 0.8535 | 0.912 | 0.912 | 0.912 | 0.8216 | 0.945 | 0.942 | 0.942 | 0.8826 |
| **KNN5** | 0.925 | 0.924 | 0.924 | 0.8467 | 0.91 | 0.909 | 0.909 | 0.8156 | 0.939 | 0.934 | 0.934 | 0.8671 |
| **KNN7** | 0.914 | 0.913 | 0913 | 0.8249 | 0.900 | 0.900 | 0.900 | 0.7973 | 0.932 | 0.926 | 0.926 | 0.8516 |
| **NB** | 0.884 | 0.884 | 0.884 | 0.7649 | 0.883 | 0.883 | 0.883 | 0.762 | 0.831 | 0.93 | 0.93 | 0.8591 |
| **BN** | 0.885 | 0.884 | 0.884 | 0.7658 | 0.883 | 0.882 | 0.882 | 0.7616 | 0.916 | 0.914 | 0.915 | 0.8276 |

##### CD Category

Table 6‑16 shows the results that were gathered from performing a classification test on the Ruby CD dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted for the said dataset.

Table ‑. Ruby Dataset (CD) using top n% Ruby-CD Features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.898 | 0.892 | 0.891 | 0.7837 | 0.875 | 0.873 | 0.873 | 0.7469 | 0.931 | 0.931 | 0.931 | 0.8612 |
| **RF** | 0.935 | 0.935 | 0.935 | 0.8694 | 0.936 | 0.935 | 0.935 | 0.8694 | 0.958 | 0.957 | 0.957 | 0.9143 |
| **KNN3** | 0.903 | 0.902 | 0.902 | 0.8041 | 0.899 | 0.896 | 0.896 | 0.7918 | 0.918 | 0.902 | 0.901 | 0.8041 |
| **KNN5** | 0.893 | 0.886 | 0.885 | 0.7714 | 0.871 | 0.867 | 0.867 | 0.7347 | 0.879 | 0.841 | 0.837 | 0.6816 |
| **KNN7** | 0.892 | 0.882 | 0.881 | 0.7633 | 0.866 | 0.861 | 0.861 | 0.7224 | 0.868 | 0.82 | 0.814 | 0.6408 |
| **NB** | 0.888 | 0.882 | 0.881 | 0.7633 | 0.895 | 0.894 | 0.894 | 0.7878 | 0.928 | 0.927 | 0.926 | 0.8531 |
| **BN** | 0.815 | 0.814 | 0.814 | 0.6286 | 0.865 | 0.865 | 0.865 | 0.7306 | 0.864 | 0.863 | 0.863 | 0.7265 |

##### CH Category

Table 6‑17 shows the results that were gathered from performing a classification test on the Ruby CH dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted for the said dataset.

Table ‑. Ruby Dataset (CH) using top n% Ruby-CH Features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.889 | 0.889 | 0.889 | 0.7778 | 0.95 | 0.944 | 0.944 | 0.8889 | 0.95 | 0.944 | 0.944 | 0.8889 |
| **RF** | 0.8889 | 0.889 | 0.889 | 0.7778 | 0.889 | 0.889 | 0.889 | 0.7778 | 0.837 | 0.833 | 0.833 | 0.6667 |
| **KNN3** | 0.95 | 0.944 | 0.944 | 0.8889 | 0.95 | 0.944 | 0.944 | 0.8889 | 0.821 | 0.722 | 0.699 | 0.4444 |
| **KNN5** | 0.95 | 0.944 | 0.944 | 0.8889 | 0.792 | 0.778 | 0.775 | 0.5556 | 0.821 | 0.722 | 0.699 | 0.4444 |
| **KNN7** | 0.889 | 0.889 | 0.889 | 0.7778 | 0.781 | 0.611 | 0.542 | 0.2222 | 0.765 | 0.556 | 0.446 | 0.1111 |
| **NB** | 0.846 | 0.778 | 0.766 | 0.5556 | 0.846 | 0.778 | 0.766 | 0.5556 | 0.846 | 0.778 | 0.766 | 0.5556 |
| **BN** | 0.444 | 0.444 | 0.444 | -0.111 | 0.444 | 0.444 | 0.444 | -0.111 | 0.444 | 0.444 | 0.444 | -0.111 |

##### D Category

Table 6‑18 shows the results that were gathered from performing a classification test on the Ruby D dataset using different classification algorithms with top 10%, 20%, and 30% of the top word features extracted for the said dataset.

Table ‑. Ruby Dataset (D) using top n% Ruby-D features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **10%** | | | | **20%** | | | | **30%** | | | |
| **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** | **P** | **R** | **FM** | **K** |
| **J48** | 0.908 | 0.905 | 0.905 | 0.8108 | 0.908 | 0.905 | 0.905 | 0.8108 | 0.908 | 0.905 | 0.905 | 0.8108 |
| **RF** | 0.974 | 0.973 | 0.973 | 0.9459 | 0.951 | 0.946 | 0.946 | 0.8919 | 0.92 | 0.905 | 0.905 | 0.8108 |
| **KNN3** | 0.924 | 0.919 | 0.919 | 0.8378 | 0.878 | 0.838 | 0.833 | 0.6757 | 0.87 | 0.824 | 0.819 | 0.6486 |
| **KNN5** | 0.924 | 0.919 | 0.919 | 0.8378 | 0.885 | 0.851 | 0.848 | 0.7027 | 0.856 | 0.797 | 0.789 | 0.5946 |
| **KNN7** | 0.866 | 0.865 | 0.865 | 0.7297 | 0.878 | 0.838 | 0.833 | 0.6757 | 0.856 | 0.797 | 0.789 | 0.5946 |
| **NB** | 0.902 | 0.878 | 0.877 | 0.7568 | 0.902 | 0.878 | 0.877 | 0.7568 | 0.902 | 0.878 | 0.877 | 0.7568 |
| **BN** | 0.83 | 0.743 | 0.725 | 0.4865 | 0.83 | 0.743 | 0.725 | 0.4865 | 0.83 | 0.743 | 0.725 | 0.4865 |

### Single Classifier

#### Mario Dataset

Table 6‑19 shows the results that were gathered from performing a classification test on the Mario dataset using the multiple binary classifier with the best feature set that was tested in the previous section, that is, the combined top 30% word features extracted from the Mario dataset.

Table ‑. Mario Dataset using top 30% Mario-Combined features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F-measure** | **Kappa** |
| **J48** | 0.817 | 0.804 | 0.804 | 0.6848 |
| **Random Forest** | 0.857 | 0.846 | 0.847 | 0.7533 |
| **KNN-3** | 0.798 | 0.778 | 0.778 | 0.6457 |
| **KNN-5** | 0.813 | 0.78 | 0.78 | 0.6475 |
| **KNN-7** | 0.803 | 0.764 | 0.762 | 0.6203 |
| **Naive Bayes** | 0.758 | 0.741 | 0.742 | 0.5923 |
| **Bayesian Network** | 0.776 | 0.765 | 0.764 | 0.6179 |

#### Ruby Dataset

Table 6‑20 shows the results that were gathered from performing a classification test on the Ruby dataset using the multiple binary classifier with the best feature set that was tested in the previous section, that is, the combined top 30% word features extracted from the Mario dataset.

Table ‑. Ruby Dataset using top 30% Mario-Combined features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F-measure** | **Kappa** |
| **J48** | 0.906 | 0.909 | 0.907 | 0.8455 |
| **Random Forest** | 0.939 | 0.939 | 0.938 | 0.8968 |
| **KNN-3** | 0.912 | 0.912 | 0.911 | 0.8496 |
| **KNN-5** | 0.898 | 0.901 | 0.898 | 0.8306 |
| **KNN-7** | 0.876 | 0.88 | 0.877 | 0.794 |
| **Naive Bayes** | 0.877 | 0.866 | 0.87 | 0.7789 |
| **Bayesian Network** | 0.875 | 0.867 | 0.87 | 0.7793 |

#### Combined Mario-Ruby Dataset

Table 6‑21 shows the results that were gathered from performing a classification test on the Combined Mario-Ruby dataset using the multiple binary classifier with the best feature set that was tested in the previous section, that is, the combined top 30% word features extracted from the Mario dataset.

Table ‑. Mario-Ruby Dataset using top 30% Mario-Combined features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F-measure** | **Kappa** |
| **J48** | 0.828 | 0.823 | 0.822 | 0.7142 |
| **Random Forest** | 0.874 | 0.866 | 0.866 | 0.784 |
| **KNN-3** | 0.828 | 0.805 | 0.806 | 0.6876 |
| **KNN-5** | 0.819 | 0.786 | 0.786 | 0.6565 |
| **KNN-7** | 0.813 | 0.763 | 0.762 | 0.6168 |
| **Naive Bayes** | 0.774 | 0.753 | 0.753 | 0.613 |
| **Bayesian Network** | 0.786 | 0.774 | 0.773 | 0.6393 |

## Evaluation of Information Extraction

The objective of the information extraction is to extract the relevant information on the tweet based on the classified category. For the CA category the following information is extracted: location mentioned in the tweet and the advice/caution. For the CD category, location in tweet, the object that is destroyed, the details of the object, and the victim’s name. For the CH category, the location in tweet and the victim’s name. For the D category, the location in tweet, the items to be donated and the details of the item. The aim of this evaluation is to test the performance of the hand-crafted rules.

### Methodology

The information extraction was tested on the Mario and Ruby datasets. The Mario dataset contained 1365 instances consisting of 651 (CA), 58 (CD), 99 (CH), 47 (D), 510 (O). The Ruby dataset contained 3224 instances consisting 1280 (CA), 246 (CD), 10 (CH), 38 (D), 1650 (O),. The extracted information was then compared to the manual annotation. The system is then measured using precision, recall and f-measure. To do the evaluation, the scoring method indicated in (Maynard et al., 2006) was used.

Precision measures the correctness of the extraction based on the number of items identified. The formula for precision is:

Whereas Recall is the measure of number of correctly indentified items over the total number of correct items. The formula for recall is:

Where,

Correct is the number of correctly extracted information

Partial is the number of incomplete extraction

Missing is the number of information that were not extracted

Spurious is the number of incorrect extraction

F-measure is the measure of the relationship between precision and recall. The formula for f-measure is:

### Caution and Advice Category

This section presents the results that were gathered from the extraction experiment on the Ruby dataset and Mario dataset for the Caution and Advice category. This section also presents the performance of the extraction based on the different information that were to be extracted given the said category.

#### Location

Table ‑. Ruby Dataset Location Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 0.8216 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.4454 |
| 2987 | 330 | 814 | 3819 | **F-measure** | 0.5777 |

Table ‑. Ruby Dataset Location Extraction Statistics (Using NER)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 0.4829 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.7777 |
| 2914 | 1492 | 3174 | 303 | **F-measure** | 0.5957 |

Table ‑. Mario Dataset Location Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 0.6762 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.3352 |
| 357 | 119 | 199 | 806 | **F-measure** | 0.4482 |

Table ‑4. Mario Dataset Location Extraction Statistics (Using NER)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 0.4602 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.2931 |
| 203 | 414 | 274 | 782 | **F-measure** | 0.3581 |

Referring to Table 6‑22 and Table 6‑23, for the location that has been extracted from the the tweets, both Ruby and Mario dataset performed a fairly high score in the precision with 0.8216 and 0.6762, respectively. This is because the dataset mostly contained single word locations. However, the rules are extracting words that are not really locations, thus increasing the spurious (incorrect) extractions. This is caused by the NER detecting location words but not used as a location. For example, the Bulkang Mayon is the name of the volcano, however, Mayon is also a street name. This causes the NER to tag it as a location. In the Mario data, the over application of rules causes the spurious extraction, thus decreasing precision. However, the recall of the extraction performed poorly in both Ruby and Mario dataset. The low recall is because the rules could not extract multi-words location. For example, the rules could not extract Calamian Group of Islands because there are currently no rules handling that location. Another reason is because the NER could not detect it because it is not in the dictionary. This causes the extraction to miss, thus decreasing recall. Table 6‑24 shows the extracted location information in some sample tweets.

In comparison with

Table ‑. Sample Extraction of Location in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| RT @meralco: Paalala: Bago pasukin ng baha ang bahay, patayin ang circuit breaker upang makaiwas sa aksidente. #RubyPH http://t.co/i5iBuBFnÃ¢â‚¬Â¦ | <none> | Bago |
| RT @DZMMTeleRadyo: Palibot ng SM Manila at Manila City Hall, may gutter deep na baha #MarioPH http://t.co/G4CK8xEsuW... http://t.co/LwM3cniÃ¢â‚¬Â¦ | SM Manila at Manila City Hall | may Manila Manila City |
| PHOTO: Maulap ngayon sa Legaspi City Albay; Bulkang Mayon, halos di matanaw http://t.co/Ed40WbLeH9 via @zhandercayabyab #RubyPH | Legaspi City Albay | Albay Mayon |
| RT @IMReadyPH: #RubyPH SIGNAL NO. 1: Calamian Group of Islands, Cuyo and Metro Manila | via @dost\_pagasa | Calamian Group of Islands, Cuyo, and Metro Manila | Cuyo Manila |
| RT @DZMMTeleRadyo: Baha sa Monumento, Caloocan, lagpas gutter na partikular sa tapat ng MCU | ulat ni RP35 J. IbaÃƒÂ±ez #MarioPH | Monumento, Caloocan | Caloocan |

#### Caution/Advice

Table ‑. Ruby Dataset Advice Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ADVICE EXTRACTION STATISTICS** | | | | **Precision** | 0.6332 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.3010 |
| 553 | 14 | 1457 | 2248 | **F-measure** | 0.4080 |

Table ‑. Mario Dataset Advice Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ADVICE EXTRACTION STATISTICS** | | | | **Precision** | 0.5593 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.3388 |
| 413 | 305 | 193 | 898 | **F-measure** | 0.4219 |

Referring to Table 6‑25 and Table 6‑26, for the extraction of advices, both Ruby and Mario dataset performed below average with less than 0.5 f-measure for both. The Ruby dataset mostly contained information about typhoon signals, while the Mario dataset contained information about flood levels. For the Ruby dataset, the extraction of the signal strength is fairly simple, however, there are still words that are not extracted like the word “No.”. The extraction only extracts “Signal 1” This causes the extraction to score only as “partial”. For the Mario dataset, the tweets are mostly concerned about flood levels. The extraction could extract the details about the flood like “gutter deep”, “abot leeg”. However, the problem now when the words are hyphenated like “abot-dibdib”. The rules cannot extract that because they are considered as one token. Table 6‑27 shows the extracted caution/advice information in some sample tweets.

Table ‑. Sample Extraction of Caution/Advice in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| RT @IMReadyPH: #RubyPH SIGNAL NO. 1: Calamian Group of Islands, Cuyo and Metro Manila | via @dost\_pagasa | Signal No. 1 | Signal 1 |
| Zhander Cayabyab on Twitter: "PHOTO: Abot-dibdib na baha sa Araneta cor. E.Rod QC (padala ni... http://t.co/QNshINlg4Q | Abot-dibdib na baha | baha |

### Casualties and Damage

This section presents the results that were gathered from the extraction experiment on the Ruby dataset and Mario dataset for the Casualties and Damage category. This section presents the performance of the extraction based on the different information that were to be extracted given the said category.

#### Location

Table ‑. Ruby Dataset Location Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 0.6274 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.5142 |
| 196 | 115 | 7 | 185 | **F-measure** | 0.5652 |

Table ‑. Mario Dataset Location Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 0.4700 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.0803 |
| 5 | 7 | 21 | 167 | **F-measure** | 0.1372 |

Referring to Table 6‑28 and Table 6‑29, for the Location information that has been extracted from the Ruby dataset, it can be seen that the extraction was able to correctly extract 196 location words as referenced from the manually annotated extraction results. However, there have been 115 incorrectly extracted words. This could be attributed to a number of reasons: (1) sometimes the location words repeat; (2) the words that were extracted were not deemed to be actual location words; and (3) there are instances in the dataset that contains multiple location words. Some of the rules that were used by the extractor seemed to be over applied to the dataset resulting to the duplication of the extracted words and that some of the words that were tagged as location words are not actually location words (e.g. Extracted = Ruby). In addition, the extractor has also extracted 7 partially correct location words - these location words are partially correct because they missed some words that are part of the location (e.g. Actual = Tacloban Airport; Extracted = Tacloban). This could be attributed to the fact that some of locations in the Philippines are composed of more than a word and since, in the current implementation, the location words are recognized in a per-word basis, it would consequently result to partially correct extraction and the rules could not handle them. Furthermore, there have been 185 missing location words because the extractor failed to extract some location words that are present in the tweet. This could be attributed to a number of reasons: (1) there is great diversity in how locations are named in the country. Some other location names can be mistakenly tagged as another part of speech in the tweet instance (e.g. Talisay as a noun instead of a place); and (2) the locations mentioned in the tweet instance are not actual words (e.g. Region 6, 7, 8). However, the rules performed poorly on the Mario dataset, with only 0.1372 f-measure. The low precision of the extraction is because (1) most of the locations are addresses. Currently, there are no rules for extracting addresses (2) use of text speak. The text normalization still could not handle it. Table 6‑30 shows the extracted location information in some sample tweets.

Table ‑. Sample Extraction of Location in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| TVPatrol #RubyPH nanalasa sa Tacloban, ilang bahay at tent winalis ng bagyo. Ulat ni @rongagalac mula sa Tacloban http://t.co/muFIoUHaL1 | Tacloban | Tacloban Tacloban Tacloban |
| #TyphoonRuby #24Oras anu na po lates update dito sa province of cavite wala na pu kasing kuryente ang ibang lugar dito #RubyPH | Cavite | dito |
| #RubyPH PHOTO: 683 pamilya sa Quirangay, Camalig, Albay, nanatili sa evacuation center dahil sa pabugso bugsong... http://t.co/LnlHB0m0JS | Quirangay Camalig Albay | Camalig |
| "@DZMMTeleRadyo: Bubong at kisame ng Tacloban Airport, sinira ni #RubyPH http://t.co/FZHADakW9X http://t.co/0u0ziXbw9e" grabe | Tacloban Airport | Tacloban |
| district 2,estero narcisa bgy. 259 tinatayang 150 at 200 n pmilya nman s estero de magdalena ang apektado s bagyong mario.#MarioPH #FloodPH | district 2 estero narcisa bgy. 259 | <none> |
| RT @DZMMTeleRadyo: Tulay sa CamNorte, nawasak ng baha; Quirino Hiway ang daan pa-Daet (Photo from @PRO5React) http://t.co/l9AopM1Zmy httpÅ â€‿\_ | CamNorte | <none> |

#### Object Name

Table ‑. Ruby Dataset Object Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **OBJECT NAME EXTRACTION STATISTICS** | | | | **Precision** | 0.5693 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.3790 |
| 131 | 99 | 1 | 215 | **F-measure** | 0.4550 |

Table ‑. Mario Dataset Object Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **OBJECT NAME EXTRACTION STATISTICS** | | | | **Precision** | 0.4737 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.1125 |
| 0 | 1 | 18 | 62 | **F-measure** | 0.1818 |

Referring to Table 6‑31 and Table 6‑32, for the Object Name information that has been extracted from the Ruby dataset, it can be seen that the extraction was able to correctly extract 131 location words as referenced from the manually annotated extraction results. However, there have been 99 incorrectly extracted words. This could be attributed to a number of reasons: (1) sometimes the words extracted already includes the words that should be extracted for object details (e.g. Actual = bahay; Extracted = 10 bahay); (2) the extractor also extracts words that are not really needed to be extracted (e.g. Actual = <none>; Extracted = walang casualty); and (3) there are instances in the dataset that contains multiple object names; thus, confusing the extractor unto which object name should it really extract (e.g. ...bahay nawasak dahil sa naglalakihang alon at lakas na hangin...). In addition, the extractor has also extracted 1 partially correct object name - these location words are partially correct because they missed some words that are part of the object name (e.g. Actual = bahay sa dalampasigan; Extracted = bahay). This could be attributed to the fact that some object names are composed of more than a word and since, in the current implementation, the extraction works in a per-word basis, it would consequently result to partially correct extraction. Furthermore, there have been 215 missing object names because the extractor failed to extract some object names that are present in the tweet. This could be attributed to a number of reasons: (1) there is great diversity in how words are used in the Filipino language. Some other object names can be mistakenly tagged/extracted as another part of speech in the tweet instance (e.g. barangay as a part of a location name instead of a noun); and (2) the rules that were used by the extractor does not support extracting object names that are placed at the start of the tweet or right after the tweet handle (e.g. @DZMMTeleRadyo: Poste ng kuryente, bumagsak sa ilang...) and those that have adjectives before them (e.g. malalaking kainan). The extraction in the Mario dataset also performed poorly. The same reason as the Ruby dataset for the low precision and recall. Table 6‑33 shows the extracted object name information in some sample tweets.

Table ‑. Sample Extraction of Object Name in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| RT  @News5AKSYON: #AksyonSaHagupit: 10 bahay sa dalampasigan ng Albay, tinangay ng malakas na alon #RubyPH | Ulat ni @edlingao | bahay | 10 bahay |
| "@gmanews: #RubyPH update mula sa press briefing ng DILG: Walang naitalang casualty sa Leyte." Good to hear. Sana sa ibang lugar din :) | <none> | Walang casualty |
| RT @DZMMTeleRadyo: PHOTO: Mga establisimyento sa Tacloban, pinapakuan ng plywood kontra baha at alon #RubyPH http://t.co/kAsVyyCCjD | via @Ã¢â‚¬Â¦ | establisimyento | baha alon |

#### Object Details

Table ‑. Ruby Dataset Object Details Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **OBJECT DETAILS EXTRACTION STATISTICS** | | | | **Precision** | 0.7531 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.1317 |
| 51 | 10 | 20 | 392 | **F-measure** | 0.2247 |

Table ‑. Mario Dataset Object Details Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **OBJECT DETAILS EXTRACTION STATISTICS** | | | | **Precision** | 0 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0 |
| 0 | 0 | 0 | 155 | **F-measure** | 0 |

Referring to Table 6‑34 and Table 6‑35, for the Object Details information that has been extracted from the Ruby dataset, it can be seen that the extraction was able to correctly extract 51 location words as referenced from the manually annotated extraction results. This can be attributed to the fact that some of the object details that have to be extracted are just composed of exactly one word. However, there have been 10 incorrectly extracted words. This could be attributed to a number of reasons: (1) sometimes the words extracted include words that should not be extracted for the object details (e.g. Actual = sinira; Extracted = sinira #RubyPH); and (2) since the extractor already extracted the details and placed them under the Object Name information, the extractor had nothing to extract for the actual object details (e.g. From the tweet: ...walang casualty…; Actual = walang, Extracted = <none>). In addition, the extractor has also extracted 1 partially correct object name - these location words are partially correct because they missed some words that are part of the object name (e.g. Actual = bahay sa dalampasigan; Extracted = bahay). This could be attributed to the fact that some object details are composed of more than a word and since, in the current implementation, the extraction works in a per-word basis, it would consequently result to partially correct extraction (e.g. Actual = lubog sa baha; Extracted = lubog baha). Furthermore, there have been 392 missing object details because the extractor failed to extract most of the object details that are present in the tweet. This could be attributed to a number of reasons: (1) since there were no extracted object names, the extractor failed to extract their corresponding object details; and (2) since there can be multiple object names that can be extracted from the tweet and their corresponding details do not necessarily follow as they were written, the extractor fails to completely extract their corresponding details. Table 6‑36 shows the extracted object details information in some sample tweets.

Table ‑. Sample Extraction of Object Details in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| RT @News5AKSYON: ROMBLON: As of 2:30PM, umabot na sa 9,584 pamilya o 30,689 katao na ang inilikas sa mga evacuation center. #RubyPH | via @Ã¢â‚¬Â¦ | 9584; 30689; inilikas | <none> |
| #AksyonSaHagupit: 10 bahay sa dalampasigan ng Albay, tinangay ng malakas na alon #RubyPH | Ulat ni @edlingao | bahay sa dalampasigan | bahay |

#### Victim Name

Table ‑. Ruby Dataset Victim Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VICTIM NAME EXTRACTION STATISTICS** | | | | **Precision** | 1 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.9825 |
| 392 | 0 | 0 | 7 | **F-measure** | 0.9912 |

Table ‑. Mario Dataset Victim Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VICTIM NAME EXTRACTION STATISTICS** | | | | **Precision** | 1 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 1 |
| 58 | 0 | 0 | 0 | **F-measure** | 1 |

Referring to Table 6‑37 and Table 6‑38, for the Victim Name information that has been extracted from the Ruby and Mario dataset, it can be seen that the extraction was able to correctly extract victim names as referenced from the manually annotated extraction results. This is due to the fact that most of the instances in the datasets actually have no victim names. However, there have been 7 missing victim names in the Ruby dataset. This is because there have been very few instances that had victim names stated within them. Also, it could be due to the fact that there is no dictionary that contains common names of people; thus, giving the extractor a hard time tagging/extracting names of people. Table 6‑39 shows the extracted victim name information in some sample tweets.

Table ‑. Sample Extraction of Victim Names in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| RT @untvradio1350: 18 taong gulang na lalake sa Iriga City na si Justine C.CaÃ±eso namatay dahil sa sobrang lamig ng panahon #RubyPH | Justine C.CaÃ±eso | <none> |

### Donation Category

This section presents the results that were gathered from the extraction experiment on the Ruby dataset and Mario dataset for the Donation category. This section presents the performance of the extraction based on the different information that were to be extracted given the said category.

#### Location

Table ‑. Ruby Dataset Location Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.3030 |
| 30 | 0 | 0 | 69 | **F-measure** | 0.4651 |

Table ‑. Mario Dataset Location Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| Correct | Spurious | Partial | *Missing* | **Recall** | 0.2602 |
| 19 | 0 | 0 | 54 | **F-measure** | 0.4130 |

Referring to Table 6‑40 and Table 6‑41, the location extracts the location that is mentioned in the tweets. The extraction scored a 0.465 f-measure. The low f-measure is because of the the low recall. The low recall is because (1) there are locations that could not be detected by the NER. (2) The extractor could not extract the complete address of the location because there are no rules that could handle addresses. (3) The system could not handle multi-words location. Table 6‑42 shows the extracted location information in some sample tweets.

Table ‑. Sample Extraction of Location in Tweets

|  |  |  |
| --- | --- | --- |
| **Tweet** | **Expected** | **Extracted** |
| RT @govph: PHOTO: @PN\_Speak personnel of PS36 in Palawan transported relief goods to Cuyo, Agutaya and Magsaysay towns. #RubyPH http://t.coÃ¢â‚¬Â¦ | Palawan Cuyo Agutaya and Magsaysay | Palawan |
| #TulongKabataan Manila Drop-off point: @anakbayan\_ph @LFSphilippines Offices 444 M.F. Jhocson St. Sampaloc Manila City. #ReliefPH #RubyPH | 44 M.F. Jhocson St. Sampaloc Manila City | <no extraction> |
| (3/3) Maaari po nating dalhin ang ating tulong simula December 6 (Sabado) 1 p.m. sa PBB House Quezon City. Salamat po. #RubyPH | PBB House Quezon City | <no extraction> |

#### Resource Name

Table ‑. Ruby Dataset Resource Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| RESOURCE NAME EXTRACTION STATISTICS | | | | **Precision** | 0.9688 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 0.8267 |
| 60 | 0 | 4 | 11 | **F-measure** | 0.8921 |

Table ‑. Mario Dataset Resource Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **RESOURCE NAME EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 1.0000 |
| 39 | 0 | 0 | 0 | **F-measure** | 1.0000 |

Referring to Table 6‑43 and Table 6‑44, the resource name extracts information about the item being donated. The extraction scored a 0.892 f-measure. The high precision is because the extractor could detect the simple items like relief goods, food packs. However, the precision will decrease once new items are found in the tweets. For the Mario dataset, no relevant information was found in the instances, so the score is perfect.

Table ‑. Ruby Dataset Resource Details Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **RESOURCE DETAILS EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 1.0000 |
| 37 | 0 | 0 | 0 | **F-measure** | 1.0000 |

Table ‑. Mario Dataset Resource Details Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **RESOURCE DETAILS EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 1.0000 |
| 50 | 0 | 0 | 0 | **F-measure** | 1.0000 |

Referring to Table 6‑45 and Table 6‑46, the resource details extract informations about the details about the item. This usually in numeric form. The resource detail got a 1 in f-measure because the extractor only needed to identify the number inside the tweet. However, the extractor will not be able to identify if the number is spelled out.

Table ‑. Ruby Dataset Victim Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VICTIM NAME EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 1.0000 |
| 31 | 0 | 0 | 0 | **F-measure** | 1.0000 |

Table ‑. Mario Dataset Victim Name Extraction Statistics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VICTIM NAME EXTRACTION STATISTICS** | | | | **Precision** | 1.0000 |
| *Correct* | *Spurious* | *Partial* | *Missing* | **Recall** | 1.0000 |
| 46 | 0 | 0 | 0 | **F-measure** | 1.0000 |

Referring to Table 6‑47 and Table 6‑48, the extraction for the victim name for both Ruby dataset and Mario dataset both scored a 1 in f-measure. This is because both of the dataset do not contained instances of victim names. Thus, garnering a perfect score. However, the extractor will not be able to handle extracting victim names.

### Call for Help Category

Because of the low number of instances for this category, the proponents could not create rules for the extraction. When the proponent examined the datasets, there are no relevant information that could be extracted from the instances.

# Conclusion and Recommendation

This chapter is divided into two sections. The first section presents the conclusion of the study and whether objectives of the research was achieved. The second section details the recommendations for possible areas of improvement per module.

## Conclusion

Overall, all of the research and system objectives were met. FILIET was able to extract relevant information from disaster related tweets, using a rule-based information extraction. The data were collected from Twitter using an external twitter mining tool called Twitter4j. After collecting the tweets, they were preprocessed through a number of techniques, specifically, normalization (this is done through the help of an external tool called NormAPI), tokenization, part-of-speech tagging and named entity recognition. After which, features were extracted from the tweet so that they may be used for the classification. In this research, the features that were used were the combination of features that were extracted from the Ruby and Mario dataset since they yielded the better results after numerous testing procedures. Moving on, the tweets were classified to different categories, which include the following: Caution and Advice (for tweets that contains helpful warnings and advisories), Casualties and Damages (for tweets that contain information about casualties and damages that arose during a typhoon/calamity), Donation (for tweets that contain information about people or organization that are willing to donate resources to those who are in need), Call for Help (for tweets that contain information about people or organization who are in need of resources) and Others (for tweets that contain information that are not relevant to the research). Then, based on the categories, specific rules are applied to extract information that was needed. Finally, the extracted information was stored to an ontology that was specifically made for this system. After going through all this, the system was then evaluated to check and see the effectiveness of the entire information extraction system.

However, there were numerous problems that were encountered when the system was developed. First, the lack of preprocessing tools that were built for the Filipino language posed problems for the extraction. The POS dictionary was still incomplete as there are words in the different tweet instances that could not be tagged by the system’s POS tagger. Also, the NER tagger was not able to fully and correctly tag the named entities in the tweet instances because of the nature of the Filipino language, wherein, named entities can be locations and vis-à-vis. Second, the ambiguity of the tweets posed problems for the classifier as some of the tweets could fall into more than two categories. Third, the lack of disaster-related tweets posed a problem to the research as the research is dependent on the availability of these kinds of tweets to meet the objectives of the research. The crawler, during the entire course of the research, was only able to collect two datasets, that is, datasets for the Typhoon Mario and the Typhoon Ruby. In addition, the lack of experts in the domain of this research forced the researchers to do a manual annotation of data to facilitate an accurate evaluation of the data and the actual system’s performance.

* Reviewed different information extraction systems through review of different papers that discussed various implementations of information extraction systems (smachine learning-based, ontology-based, and adaptive and etc.).
* Categorized the tweets into Caution and Advice, Casualty and Damages, Donation, Call for Help and Others and identified the information to be extracted for each categories.
* Reviewed different NLP techniques that were used for the preprocessing module like text normalization, part-of-speech tagging, named entity recognition, and tokenization.
* Reviewed different tools that can be used for the system like ANNIE, GATE, JAPE, ArkNLP, and Protégé.
* Analyzed different information extraction approaches like machine learning-based approach, ontology-based approach, adaptive approach, and rule-based approach.
* Implemented an information extraction system that extracts relevant information from Twitter using the different techniques, approaches and tools that were reviewed during the research.

Reviewed different metrics for the evaluation of information extraction system like precision, recall, kappa statistic, and f-measure.

## Recommendation

Listed below are some areas where improvements can be made for the system.

### Preprocessing Module

* Inclusion of a lemmatizer sub-module so as to facilitate a cleaner dataset for subsequent modules. With Filipino being a morphologically rich language, words such as “bumabaha”, “binabaha”, and “binaha” all pertain to their root word which is “baha”. With a lemmatizer, certain words will be tagged to their corresponding root word which will then simplify the job of the other modules.
* The current implementation of the POS Tagger makes use of a look-up dictionary extracted from a novels. Possible recommendations for this is to either improve upon the POS dictionary itself so that it can accommodate Twitter corpora or to create a stable POS Tagger for the Filipino language.
* Multi-worded named entities are not recognized because the system only processes on a one-token basis. Inclusion of a chunker sub-module will group related words which can help the next module identify multi-worded named entities (i.e. St. Ana, Tacloban City)
* Improve the Filipino NER dictionary in order to recognize the names of people and to be able to detect the named entities in hashtags (i.e. #cebucity, #taclobancity). By doing so, this may contribute to more accurate extraction of information.

### Category Classifier Module

* Improve categorization of tweets because there are ambiguous tweets due to ambiguity of the Filipino language, there are tweets that fall under two or more categories. There are also tweets that are uncategorized that does not fall under the existing categories.

### Ontology Population Module

* Inclusion of a mechanism to be able to store multiple instances of classes that are found within the structures of the actual ontology. Ontological classes that could include multiple instances per instance of the root Tweet class are the Location class, Victim Class, and Object classes. As encountered in the different tweet instances that were processed by the system, there are actually tweets that had multiple locations, objects and victims related to the said instances. For instance, “Laguna, Cavite & Quezon” can be found in the location that can be extracted from the given tweet instance.
* Inclusion of more class specific fields for the ontology since, currently, the information that is stored for instances of different ontological classes seemed to be lacking. Like for example, the information that is stored for the Victim class is only limited to just the victim name. Further details can be added to this class like victim details and the like.
* Inclusion of a more powerful and streamlined visualization for the ontology. With the current implementation of the visualization of the ontology, the system does not take advantage of the powerful features of using ontology to store information. The current view is limited to just viewing the information stored in the ontology in table form and there is no concrete step for the users to search and manipulate its results within the ontology.
* Inclusion of a more generalized ontological population approach to facilitate a more open and non-strict way of storing information into the ontology. With the current implementation, the system provides a very linear way of storing information by starting with the root Tweet class instance until the related instances from other classes are all linked before actually storing them into the ontology. With a generalized ontological population approach, the system can be able to manipulate the information stored in a more customizable way by being able to separately store instances for different classes.

1. 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.

Basili, R., Moschitti, A., Pazienza, M. T., & Zanzotto, F. M. (2003). Personalizing web publishing via information extraction. IEEE Intelligent Systems, 18(1), 62-70.

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.

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

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.

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 Conference 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.

Faria, Carla, and Rosario Girardi. "An Information Extraction Process for Semi-automatic Ontology Population." Soft Computing Models in Industrial and Environmental Applications, 6th International Conference SOCO 2011. Springer Berlin Heidelberg, 2011.

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. In Database 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

Geertzen, J. (2012). Inter-Rater Agreement with multiple raters and variables. Retrieved April 17, 2015, from https://mlnl.net/jg/software/ira/

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>

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., & Smith, N. A. (2011, June). Part-of-speech tagging for twitter: Annotation, features, and experiments. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2 (pp. 42-47). Association for Computational Linguistics.

Grishman, R. (1997). Information extraction: Techniques and challenges. In Information 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.

Gruber, T. (2009). Ontology. Encyclopedia of database systems, 1963-1965.

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.

Horridge, M., & Bechhofer, S. (2011). The owl api: A java api for owl ontologies. Semantic Web, 2(1), 11-21.

Howard, B. (2013) . Scanning social media to improve typhoon Haiyan relief efforts. National Geographic Daily News. Retrieved from http://news.nationalgeographic.com/news/2013/11/131108-typhoon-haiyan-philippines-crisis-mapping/

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. In Advances 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.

King, D. (2005, April). Humanitarian knowledge management. In Proceedings of the Second International ISCRAM Conference (Vol. 1, pp. 1-6). Brussels.

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/

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.

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.

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

Maedche, A., Neumann, G., & Staab, S. (2003). Bootstrapping an ontology-based information extraction system. In Intelligent exploration of the web (pp. 345-359). Physica-Verlag HD.

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.

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.

McBride, B. (2002). Jena: A semantic web toolkit. IEEE Internet Computing, 6(6), 55-59.

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/

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.

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

Neubig, G., Matsubayashi, Y., Hagiwara, M., & Murakami, K. (2011, November). Safety Information Mining-What can NLP do in a disaster-. In IJCNLP (pp. 965-973).

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

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.

Saloun, P., Velart, Z., & Klimanek, P. (2011, December). Semiautomatic domain model building from text-data. In Semantic Media Adaptation and Personalization (SMAP), 2011 Sixth International Workshop on (pp. 15-20). IEEE.

Sammut, C. (2011). TF-IDF. In Encyclopedia of machine learning. New York: Springer.

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>

Vieweg, S., Hughes, A. L., Starbird, K., & Palen, L. (2010, April). Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 1079-1088). ACM.

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

Wang, T., Bontcheva, K., Li, Y., & Cunningham, H. (2005). D2. 1.2/Ontology-Based Information Extraction (OBIE) v. 2. EU-IST Project IST-2003-506826 SEKT SEKT: Semantically Enabled Knowledge Technologies.

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

Zsu, M. (2009). TF\*IDF. In Encyclopedia of database systems. New York: Springer.

1. Examples of Filipino Morphemes

|  |  |  |  |
| --- | --- | --- | --- |
| Morpheme Element | Root Word | Prefix / 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 | - | matamis-tamis |

1. List of Unified Hashtags Used in the Crawler Module

|  |
| --- |
| #reliefph |
| #nopower |
| #nowater |
| #roadalert |
| #tracingPH |
| #rescuePH |
| #floodph |
| #marioph |
| #rubyph |

1. List of Filipino Stop Words

|  |  |  |  |
| --- | --- | --- | --- |
| sa  ng  ang  mga  kay  ni  si  kina  nina  sina  at  kaya  pati | saka  maging  man  ni  o  kapag  kung  sakali  datapwat  kahit  ngunit  samantala  subalit | dahil  sapagkat  para  upang  nang  ilalim  kinaroroonan  likod  taas  nasa  mula  hanggang  para | alinsunod  ayon  hinggil  patungkol  tanong  tungkol  ukol  wika  payag  sang-ayon  labag  laban  tulong  pamamagitan |

1. List of Irrelevant Words Removed from Top Scoring Word Features

**Mario Dataset**

| **Removed CA Features** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_@thecarlomagtaas | 6.751101 | | W\_ramon | 6.751101 | | W\_@skywaysomco | 6.751101 | | W\_@ifyuseekay | 6.751101 | | W\_antonio | 6.751101 | | W\_@moveph | 6.751101 | | W\_vista | 6.751101 | | W\_naykupo | 6.751101 | | W\_langkiwa | 6.751101 | | W\_b | 6.751101 | | W\_j | 6.751101 | | W\_aquino | 6.751101 | | W\_q | 6.751101 | | W\_abs | 6.751101 | | W\_huhuhuhu | 6.751101 | | W\_villageeastcaintarizal | 6.751101 | | W\_samson | 6.751101 | | W\_cksc | 6.751101 | | W\_@jacquemanabat | 6.751101 | | W\_@abscbnkapamilya | 6.751101 | | W\_eli | 6.751101 | | W\_@unetaelrhey | 6.751101 | | W\_pba | 6.751101 | | W\_palico | 6.751101 | | W\_kayoooo | 6.751101 | | W\_@bamalegre | 6.751101 | | W\_@dianajcnt | 6.751101 | | W\_diyos | 6.751101 | | W\_darren | 6.751101 | | W\_magsaysay | 6.751101 | | W\_@realtalkbes | 6.751101 | | W\_villageeastcainta | 6.751101 | | W\_@chipabaya | 6.751101 | | W\_@lexzxc | 6.751101 | | W\_antipolo | 6.751101 | | W\_@heyalvinfhorest | 6.751101 | | W\_@tomcatust | 6.751101 | | W\_guraabe | 6.751101 | | W\_blumenttrit | 6.751101 | | W\_@glennbrus | 6.751101 | | W\_mariano | 6.751101 | | W\_quirino | 6.751101 | | W\_silver | 6.751101 | | W\_@itsjayssayobab | 6.751101 | | W\_@akosikoyote | 6.751101 | | W\_shaw | 6.751101 | | W\_manalo | 6.751101 | | W\_imma | 6.751101 | | W\_read | 6.751101 | | W\_hahaah | 6.751101 | | W\_@aheginainq | 6.751101 | | W\_amp | 6.751101 | | |  |  | | --- | --- | | W\_nga | 6.751101 | | W\_verde | 6.751101 | | W\_centris | 6.751101 | | W\_cuenco | 6.751101 | | W\_lester | 6.751101 | | W\_@eenuhxarvlo | 6.751101 | | W\_jusko | 6.751101 | | W\_v | 9.570496 | | W\_guys | 9.570496 | | W\_pedro | 9.570496 | | W\_paquita | 10.25503 | | W\_borja | 10.25503 | | W\_tskk | 10.25503 | | W\_@itsmeacenith | 10.25503 | | W\_c | 10.25503 | | W\_thank | 10.25503 | | W\_god | 10.25503 | | W\_@polytechunivph | 10.25503 | | W\_@vargasmannysen | 10.25503 | | W\_@hadjirieta | 10.25503 | | W\_washington | 10.25503 | | W\_idul | 10.25503 | | W\_olympia | 10.25503 | | W\_@inquirerdotnet | 10.25503 | | W\_frisco | 10.25503 | | W\_ako | 10.25503 | | W\_shet | 10.25503 | | W\_ortigas | 10.40036 | | W\_@dzmmteleradyo | 10.90716 | | W\_s | 11.69612 | | W\_a | 11.82606 | | W\_p | 11.86238 | | W\_nino | 11.86238 | | W\_sana | 11.86238 | | W\_isidro | 11.86238 | | W\_rosa | 11.86238 | | W\_miguel | 11.86238 | | W\_ntvl | 11.86238 | | W\_silang | 11.86238 | | W\_joseph | 12.79362 | | W\_tsk | 12.79362 | | W\_morayta | 12.79362 | | W\_is | 12.79362 | | W\_makati | 12.79362 | | W\_cavite | 12.79362 | | W\_@youscoop | 12.91454 | | W\_@stateofdnation | 12.93192 | | W\_pasay | 12.93192 | | W\_mesa | 12.93192 | | W\_marilao | 13.23227 | | W\_marikina | 13.26459 | | W\_manila | 13.35563 | | |  |  | | --- | --- | | W\_@dzbb | 13.41685 | | W\_smcenter | 13.41685 | | W\_@marcmytweet | 13.41685 | | W\_laurente | 13.41685 | | W\_tondo | 13.41685 | | W\_tamo | 13.41685 | | W\_pasong | 13.41685 | | W\_marc | 13.41685 | | W\_@nimfaravelo | 13.41685 | | W\_@imreadyph | 13.47718 | | W\_@abscbnnews | 13.72558 | | W\_youscooper | 13.78627 | | W\_@mmda | 13.80115 | | W\_the | 13.83548 | | W\_@zhandercayabyab | 13.83548 | | W\_bacoor | 13.83548 | | W\_bulacan | 13.83548 | | W\_mateo | 13.83548 | | W\_maynila | 14.01906 | | W\_araneta | 14.11924 | | W\_lng | 14.15221 | | W\_clara | 14.15221 | | W\_ma | 14.15221 | | W\_mandaluyong | 14.15221 | | W\_metro | 14.23608 | | W\_sm | 14.32601 | | W\_pala | 14.36079 | | W\_@clingyboypren | 14.36079 | | W\_rizal | 14.36079 | | W\_cainta | 14.36097 | | W\_rin | 14.47714 | | W\_qc | 14.48414 | | W\_quezon | 14.54092 | | W\_n | 14.55977 | | W\_marcos | 14.55977 | | W\_lang | 14.55977 | | W\_@oscaroida | 14.55977 | | W\_infographic | 14.67223 | | W\_vicencio | 14.67223 | | W\_ariel | 14.67223 | | W\_e | 14.74301 | | W\_photo | 14.74301 | | W\_burgos | 14.75744 | | W\_rod | 14.75844 | | W\_padre | 14.85444 | | W\_bonifacio | 14.85444 | | W\_pasig | 14.85504 | | W\_buendia | 14.91976 | | W\_monumento | 14.93293 | | W\_@officialdohgov | 14.95971 | | W\_@rapplerdotcom | 14.99184 | | W\_@govph | 15.01515 | |

|  |  |  |
| --- | --- | --- |
| **Removed CD Features** | | |
| |  |  | | --- | --- | | W\_tomorrow | 6.751101 | | W\_court | 6.751101 | | W\_domingo | 8.390945 | | W\_baseco | 9.570496 | | W\_cuyab | 10.25503 | | |  |  | | --- | --- | | W\_tunasan | 10.25503 | | W\_muntinlupa | 10.25503 | | W\_roxas | 11.12834 | | W\_santiago | 11.86238 | | W\_d | 11.86238 | | |  |  | | --- | --- | | W\_s | 12.22628 | | W\_kalaw | 12.79362 | | W\_mario | 12.85478 | | W\_camnorte | 14.55977 | |

|  |  |  |
| --- | --- | --- |
| **Removed CH Features** | | |
| |  |  | | --- | --- | | W\_der | 6.751101 | | W\_@ancalerts | 6.751101 | | W\_rd | 6.751101 | | W\_cayetano | 6.751101 | | W\_nicolo | 6.751101 | | W\_marikina | 7.234898 | | W\_@tvpatrol | 9.570496 | | W\_domingo | 11.82606 | | |  |  | | --- | --- | | W\_camacho | 11.86238 | | W\_@maphnie | 11.86238 | | W\_& | 12.26952 | | W\_leodegracio | 13.41685 | | W\_silva | 13.41685 | | W\_magrt | 13.41685 | | W\_@jierectioner | 13.41685 | | W\_aquilino | 13.41685 | | |  |  | | --- | --- | | W\_victorino | 13.41685 | | W\_@themayorlim | 13.41685 | | W\_@philredcross | 13.83548 | | W\_@yelpalisoc | 14.36079 | | W\_@rescueph | 14.61099 | | W\_bra | 14.75185 | | W\_hi | 14.75185 | | W\_@ruffaguchenez | 14.85444 | |

|  |  |  |
| --- | --- | --- |
| **Removed D Features** | | |
| |  |  | | --- | --- | | W\_youscooper | 8.76042 | | W\_@smsupermalls | 10.25503 | | W\_ortigas | 10.40036 | | |  |  | | --- | --- | | W\_@youscoop | 10.77199 | | W\_carlo | 11.86238 | | W\_for | 11.9822 | | |  |  | | --- | --- | | W\_@tweetupmnl | 12.79362 | | W\_@laomamba | 13.37649 | | W\_@akbayanyouth | 13.83548 | |

**Ruby Dataset**

| **Removed CA Features** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_abangan | 11.28598359 | | W\_ahensya | 16.4946584 | | W\_aksidente | 17.33298031 | | W\_alalahanin | 17.23830091 | | W\_alerto | 7.358830898 | | W\_all | 7.358830898 | | W\_aming | 16.77411243 | | W\_area | 17.10733609 | | W\_bago | 17.40824207 | | W\_bagong | 7.358830898 | | W\_bahagi | 7.358830898 | | W\_bahagya | 11.28598359 | | W\_bahagyang | 15.00268403 | | W\_balita | 14.24918846 | | W\_balitang | 7.358830898 | | W\_bandang | 16.83938133 | | W\_banta | 17.02753041 | | W\_barangay | 8.735039934 | | W\_baterya | 7.358830898 | | W\_bilang | 16.82120952 | | W\_bilis | 15.53480254 | | W\_billboard | 14.24918846 | | W\_binabayo | 7.358830898 | | W\_biyernes | 17.34585995 | | W\_boat | 7.358830898 | | W\_breaker | 17.33298031 | | W\_bugso | 15.53480254 | | W\_bukas | 16.50360892 | | W\_bumaba | 7.358830898 | | W\_bumagal | 15.94222303 | | W\_buong | 14.30213027 | | W\_calamity | 15.53480254 | | W\_cellphones | 7.358830898 | | W\_center | 15.35700521 | | W\_centers | 7.358830898 | | W\_circuit | 17.33298031 | | W\_city | 13.64660321 | | W\_coast | 11.28598359 | | W\_dagat | 16.25364554 | | W\_daluyong | 17.22163461 | | W\_dapat | 17.14602451 | | W\_delata | 16.83938133 | | W\_delikado | 16.83938133 | | W\_dpwh | 7.358830898 | | W\_eastern | 13.94122087 | | W\_emergency | 15.94222303 | | W\_evacuation | 16.31910308 | | W\_everyone | 10.11022515 | | W\_forecast | 14.24918846 | | W\_gabi | 16.96290317 | | W\_galaw | 15.53480254 | | W\_ginawa | 7.358830898 | | W\_grabe | 9.734568524 | | W\_group | 16.3192836 | | W\_guard | 11.28598359 | | W\_gumagabi | 13.13643458 | | W\_gym | 15.94222303 | | W\_habang | 16.69868202 | | W\_hagupit | 17.25625022 | | W\_halos | 11.28598359 | | W\_handa | 17.33298031 | | W\_hanggang | 16.68396181 | | W\_harap | 7.358830898 | | W\_helpdesk | 15.94222303 | | W\_humupa | 7.358830898 | | W\_hydromet | 11.28598359 | | W\_i | 11.28598359 | | W\_iba | 16.69868202 | | W\_ilang | 14.74731956 | | W\_imbak | 16.83938133 | | W\_inasuyan | 7.358830898 | | W\_including | 15.00268403 | | W\_iniulat | 11.28598359 | | W\_ipinagagamit | 16.83938133 | | W\_isinailalim | 16.4946584 | | W\_isinasagawa | 15.00268403 | | W\_isla | 7.358830898 | | W\_island | 14.97092052 | | W\_islands | 15.66246385 | | W\_itinaas | 17.44974971 | | W\_just | 16.25364554 | | W\_kahit | 7.358830898 | | W\_kalupaan | 14.24918846 | | |  |  | | --- | --- | | W\_kalye | 7.358830898 | | W\_kanina | 15.00268403 | | W\_kapilya | 16.83938133 | | W\_kasalukuyang | 11.28598359 | | W\_kasunod | 11.28598359 | | W\_katamtamang | 7.358830898 | | W\_kaugnay | 7.358830898 | | W\_kayang | 7.358830898 | | W\_kaysa | 14.24918846 | | W\_klase | 17.23830091 | | W\_kong | 7.358830898 | | W\_kph | 17.43411441 | | W\_kung | 16.1337199 | | W\_kuryente | 16.91226476 | | W\_lahat | 14.38137672 | | W\_lalawigan | 17.37984163 | | W\_latest | 13.13643458 | | W\_les | 7.358830898 | | W\_light | 11.28598359 | | W\_lilikas | 7.358830898 | | W\_list | 11.67777091 | | W\_loob | 17.42107339 | | W\_lubang | 13.32088281 | | W\_lugar | 16.81388066 | | W\_lumabas | 16.83938133 | | W\_lumakas | 16.25364554 | | W\_lumihis | 15.94222303 | | W\_lunes | 16.86782415 | | W\_maaaring | 16.36704056 | | W\_maayos | 7.358830898 | | W\_mag | 16.83938133 | | W\_magingat | 7.358830898 | | W\_magkumpuni | 13.13643458 | | W\_maglandfall | 14.24918846 | | W\_magsilikas | 7.358830898 | | W\_magsisimulang | 7.358830898 | | W\_makaiwas | 17.33298031 | | W\_makatumba | 7.358830898 | | W\_malapit | 16.4946584 | | W\_mamayang | 16.83938133 | | W\_maramdaman | 7.358830898 | | W\_mas | 15.53480254 | | W\_mataas | 11.28598359 | | W\_materials | 7.358830898 | | W\_maulap | 11.28598359 | | W\_may | 16.80653174 | | W\_mula | 14.53034144 | | W\_muling | 15.53480254 | | W\_muna | 7.358830898 | | W\_nagbago | 13.13643458 | | W\_nagkansela | 17.39020351 | | W\_nagpapatupad | 11.28598359 | | W\_nagsimula | 13.13643458 | | W\_nagsisiksikang | 11.28598359 | | W\_naibalik | 11.28598359 | | W\_naka | 7.358830898 | | W\_nakapasok | 16.25364554 | | W\_nakataas | 13.13643458 | | W\_nakatira | 16.4946584 | | W\_nakatutok | 15.00268403 | | W\_naman | 15.19811337 | | W\_namataan | 15.94222303 | | W\_name | 14.24918846 | | W\_nananatili | 7.358830898 | | W\_nang | 17.13050486 | | W\_naririnig | 7.358830898 | | W\_nasa | 16.36049878 | | W\_natitirang | 11.28598359 | | W\_near | 7.358830898 | | W\_news | 17.18477212 | | W\_ninyo | 17.10733609 | | W\_noah | 16.25364554 | | W\_norte | 15.45638079 | | W\_northeast | 11.28598359 | | W\_northern | 16.74549957 | | W\_northwest | 15.53480254 | | W\_now | 13.13643458 | | W\_number | 16.83938133 | | W\_numero | 17.10733609 | | W\_official | 7.358830898 | | W\_overnight | 7.358830898 | | W\_paalala | 17.37404282 | | |  |  | | --- | --- | | W\_pagbaha | 17.43880434 | | W\_pagkain | 16.83938133 | | W\_pagpasok | 13.13643458 | | W\_paparating | 7.358830898 | | W\_papuntang | 7.358830898 | | W\_par | 16.69868202 | | W\_para | 15.79585525 | | W\_pasok | 11.28598359 | | W\_patayin | 17.33298031 | | W\_patuloy | 13.13643458 | | W\_paunan | 7.358830898 | | W\_peligroso | 7.358830898 | | W\_photo | 15.40065961 | | W\_pinalilikas | 16.25364554 | | W\_port | 11.28598359 | | W\_posibilidad | 7.358830898 | | W\_posibleng | 16.4946584 | | W\_preemptive | 15.00268403 | | W\_preso | 15.53480254 | | W\_private | 14.99547006 | | W\_probinsya | 11.82606216 | | W\_project | 16.25364554 | | W\_punongkahoy | 7.358830898 | | W\_question | 16.4946584 | | W\_rainfall | 15.88362636 | | W\_ready | 15.53480254 | | W\_regional | 14.24918846 | | W\_reinforce | 11.28598359 | | W\_requests | 16.4946584 | | W\_rescueteam | 11.28598359 | | W\_residenteng | 7.358830898 | | W\_responsibility | 17.22201042 | | W\_sakaling | 16.54310717 | | W\_sana | 11.67777091 | | W\_sapat | 16.83938133 | | W\_says | 7.358830898 | | W\_serbisyo | 17.10733609 | | W\_sibuyan | 16.4946584 | | W\_silangan | 15.00268403 | | W\_simbahan | 16.83938133 | | W\_sinarado | 7.358830898 | | W\_situation | 7.358830898 | | W\_southern | 15.15186544 | | W\_state | 15.53480254 | | W\_storm | 16.95375724 | | W\_subalit | 7.358830898 | | W\_subd | 7.358830898 | | W\_sumusunod | 17.10733609 | | W\_super | 11.28598359 | | W\_supermalls | 7.358830898 | | W\_suplay | 15.53480254 | | W\_surge | 16.95002486 | | W\_tandaan | 16.88229623 | | W\_tatawaging | 11.28598359 | | W\_teleradyo | 14.24918846 | | W\_teritoryo | 13.13643458 | | W\_tinupi | 13.13643458 | | W\_today | 16.48741185 | | W\_transistor | 11.28598359 | | W\_tsansang | 15.53480254 | | W\_tulad | 16.83938133 | | W\_tumama | 13.13643458 | | W\_tumawag | 16.4946584 | | W\_tumutok | 17.38203517 | | W\_tuwing | 17.22163461 | | W\_typhoon | 17.11877859 | | W\_ulan | 17.22163461 | | W\_ulat | 11.44171858 | | W\_units | 7.358830898 | | W\_update | 15.18667228 | | W\_updated | 11.28598359 | | W\_updates | 11.28598359 | | W\_utos | 13.13643458 | | W\_via | 7.537344055 | | W\_village | 15.94222303 | | W\_wala | 16.6698066 | | W\_walang | 12.11924969 | | W\_waves | 7.358830898 | | W\_weather | 17.1708879 | | W\_with | 11.28598359 | | W\_year | 11.28598359 | | W\_yellow | 15.88362636 | |

| **Removed CD Features** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_airport | 11.28598 | | W\_alon | 16.25365 | | W\_ayon | 14.56667 | | W\_bagsak | 15.94222 | | W\_bagyo | 13.5139 | | W\_baha | 13.72129 | | W\_barangay | 15.19811 | | W\_bayan | 15.51214 | | W\_baylon | 13.13643 | | W\_brgy | 12.06575 | | W\_brownout | 7.358831 | | W\_bubong | 16.83938 | | W\_bugsong | 13.13643 | | W\_bunkhouses | 17.06717 | | W\_buong | 13.68196 | | W\_casualty | 16.9629 | | W\_city | 13.81308 | | W\_correspondent | 11.28598 | | W\_count | 7.358831 | | W\_cross | 13.71974 | | W\_dahil | 11.44452 | | W\_dalampasigan | 14.24919 | | W\_dalawa | 17.2972 | | W\_damages | 16.69868 | | W\_dead | 15.00268 | | W\_dilg | 14.24919 | | W\_downed | 15.00268 | | W\_dulot | 7.358831 | | W\_dzmm | 12.06569 | | W\_eastern | 15.11842 | | W\_electrical | 15.00268 | | W\_establisimyento | 13.13643 | | W\_evacuees | 16.25365 | | W\_front | 11.28598 | | W\_good | 11.28598 | | W\_gov | 13.27858 | | W\_highway | 16.49466 | | W\_hypothermia | 7.358831 | | W\_ilang | 13.7455 | | W\_imprastraktura | 15.94222 | | |  |  | | --- | --- | | W\_inilikas | 16.68496 | | W\_iniwan | 15.94222 | | W\_isinara | 7.358831 | | W\_issue | 11.28598 | | W\_istruktura | 17.4388 | | W\_itinumba | 11.28598 | | W\_kahoy | 16.25365 | | W\_kasagsagan | 7.358831 | | W\_kasing | 7.358831 | | W\_katao | 15.00268 | | W\_komunikasyon | 15.54885 | | W\_kuryente | 7.893301 | | W\_lalake | 11.28598 | | W\_lates | 7.358831 | | W\_lines | 16.69868 | | W\_linya | 15.54885 | | W\_list | 11.67777 | | W\_lubog | 16.69868 | | W\_lumikas | 15.36456 | | W\_mahigit | 15.56327 | | W\_malalaking | 7.358831 | | W\_malawak | 17.22163 | | W\_mangingisda | 11.28598 | | W\_mayor | 14.73523 | | W\_mula | 13.17291 | | W\_nagdulot | 15.47884 | | W\_nagsitumbahan | 11.28598 | | W\_nailikas | 7.358831 | | W\_naitalang | 17.44081 | | W\_namatay | 13.13643 | | W\_nanatili | 13.13643 | | W\_nasawi | 17.25628 | | W\_nasira | 17.4388 | | W\_natuklap | 13.13643 | | W\_natuklapan | 15.00268 | | W\_nawasak | 7.358831 | | W\_nddrmc | 7.358831 | | W\_ngcp | 16.69868 | | W\_northeastern | 15.00268 | | W\_pabugso | 13.13643 | | |  |  | | --- | --- | | W\_pamilya | 17.38204 | | W\_pananalasa | 16.80653 | | W\_passable | 7.358831 | | W\_patay | 17.33298 | | W\_persons | 7.358831 | | W\_photo | 11.27457 | | W\_pinadapa | 7.358831 | | W\_pinsala | 17.4388 | | W\_plywood | 14.24919 | | W\_pm | 13.78627 | | W\_police | 7.358831 | | W\_post | 16.69868 | | W\_poste | 11.28598 | | W\_power | 16.69868 | | W\_press | 14.24919 | | W\_probinsya | 14.59942 | | W\_pswd | 15.94222 | | W\_red | 13.71974 | | W\_residente | 16.18527 | | W\_sagkahan | 15.88701 | | W\_san | 13.7525 | | W\_sana | 9.421552 | | W\_sinira | 15.94222 | | W\_sobrang | 11.28598 | | W\_strong | 14.52027 | | W\_tala | 11.28598 | | W\_tan | 14.24919 | | W\_tindahang | 16.25365 | | W\_topples | 15.00268 | | W\_total | 15.94222 | | W\_ulat | 15.95314 | | W\_umabot | 13.13643 | | W\_umakyat | 17.22163 | | W\_video | 16.00385 | | W\_winalis | 7.358831 | | W\_winasak | 16.69868 | | W\_yari | 16.25365 | | W\_years | 13.13643 | | W\_yero | 16.25365 | | W\_youscooper | 17.44975 | |

|  |  |  |
| --- | --- | --- |
| **Removed CH Features** | | |
| |  |  | | --- | --- | | W\_namin | 7.358831 | | W\_mauubusan | 7.358831 | | W\_ipriority | 7.358831 | | W\_kailangan | 7.358831 | | W\_phase | 7.358831 | | W\_pamimigay | 7.358831 | | W\_indigo | 7.358831 | | W\_maaapektuhan | 7.358831 | | |  |  | | --- | --- | | W\_purok | 7.358831 | | W\_paglilista | 7.358831 | | W\_local | 7.358831 | | W\_nasa | 8.742336 | | W\_talaga | 11.28598 | | W\_hindi | 12.91378 | | W\_water | 13.71974 | | |  |  | | --- | --- | | W\_than | 14.24919 | | W\_need | 14.24919 | | W\_nila | 14.24919 | | W\_church | 14.24919 | | W\_more | 14.24919 | | W\_iaccommodate | 14.24919 | | W\_yung | 15.00268 | |

|  |  |  |
| --- | --- | --- |
| **Removed D Features** | | |
| |  |  | | --- | --- | | W\_tumatanggap | 7.358831 | | W\_bangus | 7.358831 | | W\_victims | 7.358831 | | W\_countries | 7.358831 | | W\_globe | 7.358831 | | W\_ulit | 7.358831 | | W\_handang | 7.358831 | | W\_sagip | 7.358831 | | W\_providing | 7.358831 | | W\_donasyon | 7.358831 | | W\_simula | 7.358831 | | W\_nating | 7.358831 | | W\_hinihikayat | 7.358831 | | W\_maaari | 7.358831 | | W\_calls | 7.358831 | | |  |  | | --- | --- | | W\_grocery | 7.358831 | | W\_ngiti | 7.358831 | | W\_offers | 7.358831 | | W\_biktima | 7.358831 | | W\_bigas | 7.358831 | | W\_magbigay | 7.358831 | | W\_offices | 10.11023 | | W\_evacuee | 10.11023 | | W\_volunteers | 10.59839 | | W\_nakahanda | 10.59839 | | W\_free | 10.59839 | | W\_photo | 10.85922 | | W\_ipamigay | 11.28598 | | W\_point | 11.28598 | | |  |  | | --- | --- | | W\_maapektuhan | 11.61537 | | W\_inihahanda | 12.06569 | | W\_dswd | 13.13643 | | W\_salamat | 13.27858 | | W\_food | 15.0861 | | W\_family | 15.94222 | | W\_navy | 15.94222 | | W\_municipality | 15.94222 | | W\_towns | 16.25365 | | W\_packs | 16.49466 | | W\_relief | 16.76702 | | W\_goods | 17.05885 | | W\_transported | 17.22163 | | W\_personnel | 17.22163 | |

1. Top 30% TFIDF Word Features

**Mario Dataset**

| **Caution and Advice** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_floods | 6.751101 | | W\_pahirap | 6.751101 | | W\_araw | 6.751101 | | W\_sasakyang | 6.751101 | | W\_lipa | 6.751101 | | W\_hala | 6.751101 | | W\_tska | 6.751101 | | W\_davao | 6.751101 | | W\_raw | 6.751101 | | W\_binti | 6.751101 | | W\_handa | 6.751101 | | W\_humupa | 6.751101 | | W\_flight | 6.751101 | | W\_kanto | 6.751101 | | W\_swimming | 6.751101 | | W\_makita | 6.751101 | | W\_mahabang | 6.751101 | | W\_ngaun | 6.751101 | | W\_bandang | 6.751101 | | W\_kahit | 6.751101 | | W\_fifth | 6.751101 | | W\_scout | 6.751101 | | W\_sapa | 6.751101 | | W\_red | 6.751101 | | W\_post | 6.751101 | | W\_generated | 6.751101 | | W\_maliliit | 6.751101 | | W\_sang | 6.751101 | | W\_papuntang | 6.751101 | | W\_evac | 6.751101 | | W\_nakakaiyak | 6.751101 | | W\_sinarado | 6.751101 | | W\_tweet | 6.751101 | | W\_with | 6.751101 | | W\_slex | 6.751101 | | W\_family | 6.751101 | | W\_kaunti | 6.751101 | | W\_kasiglahan | 6.751101 | | W\_tong | 6.751101 | | W\_pinapansin | 6.751101 | | W\_muna | 6.751101 | | W\_harap | 6.751101 | | W\_quad | 6.751101 | | W\_magkabilang | 6.751101 | | W\_hway | 6.751101 | | W\_lubog | 6.751101 | | W\_boundary | 6.751101 | | W\_facilitate | 6.751101 | | W\_rising | 6.751101 | | W\_rob | 6.751101 | | W\_meters | 6.751101 | | W\_wala | 6.993005 | | W\_apektado | 7.988709 | | W\_tumana | 8.802752 | | W\_bubong | 9.077457 | | W\_habang | 9.570496 | | W\_bukas | 9.570496 | | W\_condo | 10.25503 | | W\_isla | 10.25503 | | W\_stop | 10.25503 | | W\_gma | 10.25503 | | W\_yellow | 10.25503 | | W\_severe | 10.25503 | | W\_agos | 10.25503 | | W\_puddle | 10.25503 | | W\_calamity | 10.25503 | | |  |  | | --- | --- | | W\_nag | 10.25503 | | W\_reinforce | 10.25503 | | W\_block | 10.25503 | | W\_kayang | 10.25503 | | W\_system | 10.25503 | | W\_sulong | 10.25503 | | W\_kawawa | 10.25503 | | W\_tapat | 10.25503 | | W\_below | 10.25503 | | W\_save | 10.79036 | | W\_share | 10.79036 | | W\_barangay | 11.25127 | | W\_building | 11.86238 | | W\_marami | 11.86238 | | W\_blvd | 11.86238 | | W\_padala | 11.86238 | | W\_motorista | 11.86238 | | W\_mataas | 11.86238 | | W\_ngyon | 11.86238 | | W\_everyone | 11.86238 | | W\_around | 11.86238 | | W\_buti | 11.86238 | | W\_grabi | 11.86238 | | W\_homes | 11.86238 | | W\_town | 11.86238 | | W\_news | 11.86238 | | W\_bus | 11.86238 | | W\_out | 11.86238 | | W\_sainyo | 11.86238 | | W\_ilang | 12.12608 | | W\_matinding | 12.26952 | | W\_now | 12.26952 | | W\_of | 12.75078 | | W\_labas | 12.79362 | | W\_abo | 12.79362 | | W\_inyong | 12.79362 | | W\_area | 12.79362 | | W\_tahanan | 12.79362 | | W\_namin | 12.79362 | | W\_declares | 12.79362 | | W\_bewang | 12.79362 | | W\_floor | 12.79362 | | W\_avenue | 12.79362 | | W\_wag | 12.79362 | | W\_uhod | 12.79362 | | W\_bago | 12.79362 | | W\_pumasok | 12.79362 | | W\_sasakyan | 12.79362 | | W\_kanina | 12.79362 | | W\_question | 12.79646 | | W\_may | 12.90756 | | W\_ave | 12.93192 | | W\_weather | 13.12667 | | W\_flood | 13.3548 | | W\_road | 13.37649 | | W\_bagong | 13.41685 | | W\_lalim | 13.41685 | | W\_point | 13.41685 | | W\_along | 13.41685 | | W\_magdulot | 13.41685 | | W\_limay | 13.41685 | | W\_umiwas | 13.41685 | | W\_pwedeng | 13.41685 | | W\_paglusong | 13.41685 | | W\_as | 13.6089 | | W\_and | 13.6125 | | |  |  | | --- | --- | | W\_lagpas | 13.67392 | | W\_city | 13.68057 | | W\_ito | 13.73807 | | W\_subd | 13.83548 | | W\_pero | 13.83548 | | W\_video | 13.83548 | | W\_ulan | 13.88812 | | W\_village | 14.07786 | | W\_malapit | 14.15221 | | W\_samin | 14.15221 | | W\_subided | 14.15221 | | W\_lugar | 14.15221 | | W\_tao | 14.15221 | | W\_patungong | 14.19832 | | W\_kuha | 14.27654 | | W\_dahil | 14.32601 | | W\_cor | 14.34579 | | W\_bahagi | 14.36079 | | W\_bahang | 14.36079 | | W\_nananatiling | 14.36079 | | W\_kahabaan | 14.36079 | | W\_our | 14.53482 | | W\_dito | 14.5498 | | W\_hall | 14.55977 | | W\_dapat | 14.55977 | | W\_palibot | 14.55977 | | W\_river | 14.55977 | | W\_warning | 14.55977 | | W\_highway | 14.55977 | | W\_gawin | 14.55977 | | W\_babala | 14.55977 | | W\_karangalan | 14.55977 | | W\_safe | 14.55977 | | W\_passable | 14.56979 | | W\_gutter | 14.6137 | | W\_magingat | 14.67223 | | W\_deep | 14.67223 | | W\_hotlines | 14.67223 | | W\_ukol | 14.67223 | | W\_via | 14.7123 | | W\_san | 14.74301 | | W\_hanggang | 14.75185 | | W\_live | 14.75845 | | W\_naman | 14.75845 | | W\_high | 14.75845 | | W\_ngayong | 14.7598 | | W\_amin | 14.7598 | | W\_liwasang | 14.85444 | | W\_paalala | 14.85444 | | W\_kung | 14.85504 | | W\_sitwasyon | 14.88148 | | W\_follow | 14.88148 | | W\_grabe | 14.88152 | | W\_not | 14.91976 | | W\_tuhod | 14.93293 | | W\_going | 14.95971 | | W\_blog | 14.95971 | | W\_tayo | 14.97828 | | W\_vehicles | 14.97828 | | W\_all | 14.99184 | | W\_dibdib | 14.99184 | | W\_subsided | 14.99184 | | W\_tubig | 14.99184 | | W\_dry | 14.99184 | | W\_ingat | 14.99184 | | W\_types | 14.99184 | |

| **Casualty and Damage** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_quiteria | 6.751101 | | W\_ilikas | 6.751101 | | W\_volunteers | 6.751101 | | W\_limandaang | 6.751101 | | W\_in | 6.960314 | | W\_ilang | 6.993005 | | W\_pagbaha | 7.354702 | | W\_sto | 7.710384 | | W\_church | 8.390945 | | |  |  | | --- | --- | | W\_patuloy | 8.705593 | | W\_inilikas | 8.705593 | | W\_kabuluan | 10.25503 | | W\_cupang | 10.25503 | | W\_compound | 11.82606 | | W\_brgy | 12.24199 | | W\_ay | 12.35598 | | W\_mahigit | 12.79362 | | W\_bagyo | 13.41685 | | |  |  | | --- | --- | | W\_dulot | 13.41685 | | W\_estero | 13.41685 | | W\_apektado | 14.1153 | | W\_district | 14.32601 | | W\_bagyong | 14.36079 | | W\_dist | 14.55977 | | W\_tulay | 14.7598 | | W\_tinatayang | 14.88148 | | W\_pamilya | 14.97851 | |

| **Call For Help** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_complex | 6.751101 | | W\_wla | 6.751101 | | W\_sub | 6.751101 | | W\_roofs | 6.751101 | | W\_people | 6.751101 | | W\_call | 6.751101 | | W\_open | 7.217344 | | W\_bubong | 9.077457 | | W\_st | 9.077457 | | W\_pls | 9.786751 | | W\_luloy | 10.25503 | | W\_isang | 10.25503 | | W\_natrapped | 10.25503 | | W\_roga | 10.25503 | | W\_po | 10.30097 | | W\_sto | 10.86689 | | |  |  | | --- | --- | | W\_rescue | 11.12834 | | W\_school | 11.65389 | | W\_lahat | 11.82606 | | W\_aabot | 11.86238 | | W\_kabayani | 11.86238 | | W\_elem | 12.16895 | | W\_friends | 12.26952 | | W\_nasa | 12.40281 | | W\_evacuees | 12.53579 | | W\_nangka | 12.79362 | | W\_nila | 12.79362 | | W\_dear | 12.79362 | | W\_needs | 12.79362 | | W\_blogger | 12.79362 | | W\_iaccommodate | 12.79362 | | |  |  | | --- | --- | | W\_than | 12.79362 | | W\_families | 12.93192 | | W\_pahelp | 13.41685 | | W\_sir | 13.41685 | | W\_food | 13.68088 | | W\_help | 13.79993 | | W\_water | 14.0234 | | W\_tumana | 14.14026 | | W\_wala | 14.31461 | | W\_kasi | 14.46535 | | W\_kami | 14.46535 | | W\_good | 14.55977 | | W\_kong | 14.61099 | | W\_need | 14.6137 | | W\_i | 14.75185 | | W\_nabaha | 14.75185 | |

| **Donation** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_metro | 8.20982 | | W\_lahat | 8.390945 | | W\_you | 9.077457 | | W\_ating | 9.570496 | | W\_extension | 10.08178 | | W\_weather | 10.17201 | | W\_requests | 10.17201 | | W\_parking | 10.25503 | | W\_ginawa | 10.25503 | | W\_advisory | 10.25503 | | |  |  | | --- | --- | | W\_kaba | 10.25503 | | W\_malls | 10.25503 | | W\_service | 10.25503 | | W\_volunteer | 10.40036 | | W\_affected | 11.25127 | | W\_pong | 11.25127 | | W\_ilang | 11.61242 | | W\_gusto | 11.86238 | | W\_nagpabuhat | 11.86238 | | |  |  | | --- | --- | | W\_magvolunteer | 11.86238 | | W\_mabasa | 11.86238 | | W\_more | 12.69391 | | W\_ngayon | 13.34645 | | W\_open | 13.62874 | | W\_contact | 13.62874 | | W\_just | 13.62874 | | W\_masilungan | 13.78627 | | W\_rescuer | 14.55977 | |

**Ruby Dataset**

| **Caution and Advice** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_abangan | 11.28598359 | | W\_ahensya | 16.4946584 | | W\_aksidente | 17.33298031 | | W\_alalahanin | 17.23830091 | | W\_alerto | 7.358830898 | | W\_all | 7.358830898 | | W\_aming | 16.77411243 | | W\_area | 17.10733609 | | W\_bago | 17.40824207 | | W\_bagong | 7.358830898 | | W\_bahagi | 7.358830898 | | W\_bahagya | 11.28598359 | | W\_bahagyang | 15.00268403 | | W\_balita | 14.24918846 | | W\_balitang | 7.358830898 | | W\_bandang | 16.83938133 | | W\_banta | 17.02753041 | | W\_barangay | 8.735039934 | | W\_baterya | 7.358830898 | | W\_bilang | 16.82120952 | | W\_bilis | 15.53480254 | | W\_billboard | 14.24918846 | | W\_binabayo | 7.358830898 | | W\_biyernes | 17.34585995 | | W\_boat | 7.358830898 | | W\_breaker | 17.33298031 | | W\_bugso | 15.53480254 | | W\_bukas | 16.50360892 | | W\_bumaba | 7.358830898 | | W\_bumagal | 15.94222303 | | W\_buong | 14.30213027 | | W\_calamity | 15.53480254 | | W\_cellphones | 7.358830898 | | W\_center | 15.35700521 | | W\_centers | 7.358830898 | | W\_circuit | 17.33298031 | | W\_city | 13.64660321 | | W\_coast | 11.28598359 | | W\_dagat | 16.25364554 | | W\_daluyong | 17.22163461 | | W\_dapat | 17.14602451 | | W\_delata | 16.83938133 | | W\_delikado | 16.83938133 | | W\_dpwh | 7.358830898 | | W\_eastern | 13.94122087 | | W\_emergency | 15.94222303 | | W\_evacuation | 16.31910308 | | W\_everyone | 10.11022515 | | W\_forecast | 14.24918846 | | W\_gabi | 16.96290317 | | W\_galaw | 15.53480254 | | W\_ginawa | 7.358830898 | | W\_grabe | 9.734568524 | | W\_group | 16.3192836 | | W\_guard | 11.28598359 | | W\_gumagabi | 13.13643458 | | W\_gym | 15.94222303 | | W\_habang | 16.69868202 | | W\_hagupit | 17.25625022 | | W\_halos | 11.28598359 | | W\_handa | 17.33298031 | | W\_hanggang | 16.68396181 | | W\_harap | 7.358830898 | | W\_helpdesk | 15.94222303 | | W\_humupa | 7.358830898 | | W\_hydromet | 11.28598359 | | W\_i | 11.28598359 | | W\_iba | 16.69868202 | | W\_ilang | 14.74731956 | | W\_imbak | 16.83938133 | | W\_inasuyan | 7.358830898 | | W\_including | 15.00268403 | | W\_iniulat | 11.28598359 | | W\_ipinagagamit | 16.83938133 | | W\_isinailalim | 16.4946584 | | W\_isinasagawa | 15.00268403 | | W\_isla | 7.358830898 | | W\_island | 14.97092052 | | W\_islands | 15.66246385 | | W\_itinaas | 17.44974971 | | W\_just | 16.25364554 | | W\_kahit | 7.358830898 | | W\_kalupaan | 14.24918846 | | |  |  | | --- | --- | | W\_kalye | 7.358830898 | | W\_kanina | 15.00268403 | | W\_kapilya | 16.83938133 | | W\_kasalukuyang | 11.28598359 | | W\_kasunod | 11.28598359 | | W\_katamtamang | 7.358830898 | | W\_kaugnay | 7.358830898 | | W\_kayang | 7.358830898 | | W\_kaysa | 14.24918846 | | W\_klase | 17.23830091 | | W\_kong | 7.358830898 | | W\_kph | 17.43411441 | | W\_kung | 16.1337199 | | W\_kuryente | 16.91226476 | | W\_lahat | 14.38137672 | | W\_lalawigan | 17.37984163 | | W\_latest | 13.13643458 | | W\_les | 7.358830898 | | W\_light | 11.28598359 | | W\_lilikas | 7.358830898 | | W\_list | 11.67777091 | | W\_loob | 17.42107339 | | W\_lubang | 13.32088281 | | W\_lugar | 16.81388066 | | W\_lumabas | 16.83938133 | | W\_lumakas | 16.25364554 | | W\_lumihis | 15.94222303 | | W\_lunes | 16.86782415 | | W\_maaaring | 16.36704056 | | W\_maayos | 7.358830898 | | W\_mag | 16.83938133 | | W\_magingat | 7.358830898 | | W\_magkumpuni | 13.13643458 | | W\_maglandfall | 14.24918846 | | W\_magsilikas | 7.358830898 | | W\_magsisimulang | 7.358830898 | | W\_makaiwas | 17.33298031 | | W\_makatumba | 7.358830898 | | W\_malapit | 16.4946584 | | W\_mamayang | 16.83938133 | | W\_maramdaman | 7.358830898 | | W\_mas | 15.53480254 | | W\_mataas | 11.28598359 | | W\_materials | 7.358830898 | | W\_maulap | 11.28598359 | | W\_may | 16.80653174 | | W\_mula | 14.53034144 | | W\_muling | 15.53480254 | | W\_muna | 7.358830898 | | W\_nagbago | 13.13643458 | | W\_nagkansela | 17.39020351 | | W\_nagpapatupad | 11.28598359 | | W\_nagsimula | 13.13643458 | | W\_nagsisiksikang | 11.28598359 | | W\_naibalik | 11.28598359 | | W\_naka | 7.358830898 | | W\_nakapasok | 16.25364554 | | W\_nakataas | 13.13643458 | | W\_nakatira | 16.4946584 | | W\_nakatutok | 15.00268403 | | W\_naman | 15.19811337 | | W\_namataan | 15.94222303 | | W\_name | 14.24918846 | | W\_nananatili | 7.358830898 | | W\_nang | 17.13050486 | | W\_naririnig | 7.358830898 | | W\_nasa | 16.36049878 | | W\_natitirang | 11.28598359 | | W\_near | 7.358830898 | | W\_news | 17.18477212 | | W\_ninyo | 17.10733609 | | W\_noah | 16.25364554 | | W\_norte | 15.45638079 | | W\_northeast | 11.28598359 | | W\_northern | 16.74549957 | | W\_northwest | 15.53480254 | | W\_now | 13.13643458 | | W\_number | 16.83938133 | | W\_numero | 17.10733609 | | W\_official | 7.358830898 | | W\_overnight | 7.358830898 | | W\_paalala | 17.37404282 | | |  |  | | --- | --- | | W\_pagbaha | 17.43880434 | | W\_pagkain | 16.83938133 | | W\_pagpasok | 13.13643458 | | W\_paparating | 7.358830898 | | W\_papuntang | 7.358830898 | | W\_par | 16.69868202 | | W\_para | 15.79585525 | | W\_pasok | 11.28598359 | | W\_patayin | 17.33298031 | | W\_patuloy | 13.13643458 | | W\_paunan | 7.358830898 | | W\_peligroso | 7.358830898 | | W\_photo | 15.40065961 | | W\_pinalilikas | 16.25364554 | | W\_port | 11.28598359 | | W\_posibilidad | 7.358830898 | | W\_posibleng | 16.4946584 | | W\_preemptive | 15.00268403 | | W\_preso | 15.53480254 | | W\_private | 14.99547006 | | W\_probinsya | 11.82606216 | | W\_project | 16.25364554 | | W\_punongkahoy | 7.358830898 | | W\_question | 16.4946584 | | W\_rainfall | 15.88362636 | | W\_ready | 15.53480254 | | W\_regional | 14.24918846 | | W\_reinforce | 11.28598359 | | W\_requests | 16.4946584 | | W\_rescueteam | 11.28598359 | | W\_residenteng | 7.358830898 | | W\_responsibility | 17.22201042 | | W\_sakaling | 16.54310717 | | W\_sana | 11.67777091 | | W\_sapat | 16.83938133 | | W\_says | 7.358830898 | | W\_serbisyo | 17.10733609 | | W\_sibuyan | 16.4946584 | | W\_silangan | 15.00268403 | | W\_simbahan | 16.83938133 | | W\_sinarado | 7.358830898 | | W\_situation | 7.358830898 | | W\_southern | 15.15186544 | | W\_state | 15.53480254 | | W\_storm | 16.95375724 | | W\_subalit | 7.358830898 | | W\_subd | 7.358830898 | | W\_sumusunod | 17.10733609 | | W\_super | 11.28598359 | | W\_supermalls | 7.358830898 | | W\_suplay | 15.53480254 | | W\_surge | 16.95002486 | | W\_tandaan | 16.88229623 | | W\_tatawaging | 11.28598359 | | W\_teleradyo | 14.24918846 | | W\_teritoryo | 13.13643458 | | W\_tinupi | 13.13643458 | | W\_today | 16.48741185 | | W\_transistor | 11.28598359 | | W\_tsansang | 15.53480254 | | W\_tulad | 16.83938133 | | W\_tumama | 13.13643458 | | W\_tumawag | 16.4946584 | | W\_tumutok | 17.38203517 | | W\_tuwing | 17.22163461 | | W\_typhoon | 17.11877859 | | W\_ulan | 17.22163461 | | W\_ulat | 11.44171858 | | W\_units | 7.358830898 | | W\_update | 15.18667228 | | W\_updated | 11.28598359 | | W\_updates | 11.28598359 | | W\_utos | 13.13643458 | | W\_via | 7.537344055 | | W\_village | 15.94222303 | | W\_wala | 16.6698066 | | W\_walang | 12.11924969 | | W\_waves | 7.358830898 | | W\_weather | 17.1708879 | | W\_with | 11.28598359 | | W\_year | 11.28598359 | | W\_yellow | 15.88362636 | |

| **Casualty and Damage** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_airport | 11.28598 | | W\_alon | 16.25365 | | W\_ayon | 14.56667 | | W\_bagsak | 15.94222 | | W\_bagyo | 13.5139 | | W\_baha | 13.72129 | | W\_barangay | 15.19811 | | W\_bayan | 15.51214 | | W\_baylon | 13.13643 | | W\_brgy | 12.06575 | | W\_brownout | 7.358831 | | W\_bubong | 16.83938 | | W\_bugsong | 13.13643 | | W\_bunkhouses | 17.06717 | | W\_buong | 13.68196 | | W\_casualty | 16.9629 | | W\_city | 13.81308 | | W\_correspondent | 11.28598 | | W\_count | 7.358831 | | W\_cross | 13.71974 | | W\_dahil | 11.44452 | | W\_dalampasigan | 14.24919 | | W\_dalawa | 17.2972 | | W\_damages | 16.69868 | | W\_dead | 15.00268 | | W\_dilg | 14.24919 | | W\_downed | 15.00268 | | W\_dulot | 7.358831 | | W\_dzmm | 12.06569 | | W\_eastern | 15.11842 | | W\_electrical | 15.00268 | | W\_establisimyento | 13.13643 | | W\_evacuees | 16.25365 | | W\_front | 11.28598 | | W\_good | 11.28598 | | W\_gov | 13.27858 | | W\_highway | 16.49466 | | W\_hypothermia | 7.358831 | | W\_ilang | 13.7455 | | W\_imprastraktura | 15.94222 | | |  |  | | --- | --- | | W\_inilikas | 16.68496 | | W\_iniwan | 15.94222 | | W\_isinara | 7.358831 | | W\_issue | 11.28598 | | W\_istruktura | 17.4388 | | W\_itinumba | 11.28598 | | W\_kahoy | 16.25365 | | W\_kasagsagan | 7.358831 | | W\_kasing | 7.358831 | | W\_katao | 15.00268 | | W\_komunikasyon | 15.54885 | | W\_kuryente | 7.893301 | | W\_lalake | 11.28598 | | W\_lates | 7.358831 | | W\_lines | 16.69868 | | W\_linya | 15.54885 | | W\_list | 11.67777 | | W\_lubog | 16.69868 | | W\_lumikas | 15.36456 | | W\_mahigit | 15.56327 | | W\_malalaking | 7.358831 | | W\_malawak | 17.22163 | | W\_mangingisda | 11.28598 | | W\_mayor | 14.73523 | | W\_mula | 13.17291 | | W\_nagdulot | 15.47884 | | W\_nagsitumbahan | 11.28598 | | W\_nailikas | 7.358831 | | W\_naitalang | 17.44081 | | W\_namatay | 13.13643 | | W\_nanatili | 13.13643 | | W\_nasawi | 17.25628 | | W\_nasira | 17.4388 | | W\_natuklap | 13.13643 | | W\_natuklapan | 15.00268 | | W\_nawasak | 7.358831 | | W\_nddrmc | 7.358831 | | W\_ngcp | 16.69868 | | W\_northeastern | 15.00268 | | W\_pabugso | 13.13643 | | |  |  | | --- | --- | | W\_pamilya | 17.38204 | | W\_pananalasa | 16.80653 | | W\_passable | 7.358831 | | W\_patay | 17.33298 | | W\_persons | 7.358831 | | W\_photo | 11.27457 | | W\_pinadapa | 7.358831 | | W\_pinsala | 17.4388 | | W\_plywood | 14.24919 | | W\_pm | 13.78627 | | W\_police | 7.358831 | | W\_post | 16.69868 | | W\_poste | 11.28598 | | W\_power | 16.69868 | | W\_press | 14.24919 | | W\_probinsya | 14.59942 | | W\_pswd | 15.94222 | | W\_red | 13.71974 | | W\_residente | 16.18527 | | W\_sagkahan | 15.88701 | | W\_san | 13.7525 | | W\_sana | 9.421552 | | W\_sinira | 15.94222 | | W\_sobrang | 11.28598 | | W\_strong | 14.52027 | | W\_tala | 11.28598 | | W\_tan | 14.24919 | | W\_tindahang | 16.25365 | | W\_topples | 15.00268 | | W\_total | 15.94222 | | W\_ulat | 15.95314 | | W\_umabot | 13.13643 | | W\_umakyat | 17.22163 | | W\_video | 16.00385 | | W\_winalis | 7.358831 | | W\_winasak | 16.69868 | | W\_yari | 16.25365 | | W\_years | 13.13643 | | W\_yero | 16.25365 | | W\_youscooper | 17.44975 | |

| **Call For Help** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_namin | 7.358831 | | W\_mauubusan | 7.358831 | | W\_ipriority | 7.358831 | | W\_kailangan | 7.358831 | | W\_phase | 7.358831 | | W\_pamimigay | 7.358831 | | W\_indigo | 7.358831 | | W\_maaapektuhan | 7.358831 | | |  |  | | --- | --- | | W\_purok | 7.358831 | | W\_paglilista | 7.358831 | | W\_local | 7.358831 | | W\_nasa | 8.742336 | | W\_talaga | 11.28598 | | W\_hindi | 12.91378 | | W\_water | 13.71974 | | |  |  | | --- | --- | | W\_than | 14.24919 | | W\_need | 14.24919 | | W\_nila | 14.24919 | | W\_church | 14.24919 | | W\_more | 14.24919 | | W\_iaccommodate | 14.24919 | | W\_yung | 15.00268 | |

| **Donation** | | |
| --- | --- | --- |
| |  |  | | --- | --- | | W\_tumatanggap | 7.358831 | | W\_bangus | 7.358831 | | W\_victims | 7.358831 | | W\_countries | 7.358831 | | W\_globe | 7.358831 | | W\_ulit | 7.358831 | | W\_handang | 7.358831 | | W\_sagip | 7.358831 | | W\_providing | 7.358831 | | W\_donasyon | 7.358831 | | W\_simula | 7.358831 | | W\_nating | 7.358831 | | W\_hinihikayat | 7.358831 | | W\_maaari | 7.358831 | | W\_calls | 7.358831 | | |  |  | | --- | --- | | W\_grocery | 7.358831 | | W\_ngiti | 7.358831 | | W\_offers | 7.358831 | | W\_biktima | 7.358831 | | W\_bigas | 7.358831 | | W\_magbigay | 7.358831 | | W\_offices | 10.11023 | | W\_evacuee | 10.11023 | | W\_volunteers | 10.59839 | | W\_nakahanda | 10.59839 | | W\_free | 10.59839 | | W\_photo | 10.85922 | | W\_ipamigay | 11.28598 | | W\_point | 11.28598 | | |  |  | | --- | --- | | W\_maapektuhan | 11.61537 | | W\_inihahanda | 12.06569 | | W\_dswd | 13.13643 | | W\_salamat | 13.27858 | | W\_food | 15.0861 | | W\_family | 15.94222 | | W\_navy | 15.94222 | | W\_municipality | 15.94222 | | W\_towns | 16.25365 | | W\_packs | 16.49466 | | W\_relief | 16.76702 | | W\_goods | 17.05885 | | W\_transported | 17.22163 | | W\_personnel | 17.22163 | |

1. Top 30% Combined Word Features of Mario and Ruby Dataset

|  |  |  |
| --- | --- | --- |
| W\_aabot 11.86238326  W\_abangan 11.28598359  W\_abo 12.79362126  W\_advisory 10.25502678  W\_affected 11.25127362  W\_agos 10.25502678  W\_ahensya 16.4946584  W\_airport 11.28598359  W\_aksidente 17.33298031  W\_alalahanin 17.23830091  W\_alerto 7.358830898  W\_alon 16.25364554  W\_along 13.41685182  W\_amin 14.75979591  W\_aming 16.77411243  W\_and 13.61250272  W\_apektado 7.98870868  W\_araw 6.751101469  W\_around 11.86238326  W\_as 13.60890466  W\_ating 9.570496119  W\_ave 12.93192301  W\_avenue 12.79362126  W\_ay 12.3559816  W\_ayon 14.56666992  W\_babala 14.55976712  W\_bagsak 15.94222303  W\_bagyong 14.36078807  W\_baha 13.7212935  W\_bahagya 11.28598359  W\_bahagyang 15.00268403  W\_bahang 14.36078807  W\_balita 14.24918846  W\_balitang 7.358830898  W\_bangus 7.358830898  W\_banta 17.02753041  W\_baterya 7.358830898  W\_bayan 15.51213645  W\_baylon 13.13643458  W\_below 10.25502678  W\_bewang 12.79362126  W\_bigas 7.358830898  W\_biktima 7.358830898  W\_bilang 16.82120952  W\_bilis 15.53480254  W\_billboard 14.24918846  W\_binabayo 7.358830898  W\_binti 6.751101469  W\_biyernes 17.34585995  W\_block 10.25502678  W\_blog 14.95970567  W\_blogger 12.79362126  W\_blvd 11.86238326  W\_boat 7.358830898  W\_boundary 6.751101469  W\_breaker 17.33298031  W\_brownout 7.358830898  W\_bugso 15.53480254  W\_bugsong 13.13643458  W\_building 11.86238326  W\_bumaba 7.358830898  W\_bumagal 15.94222303  W\_bunkhouses 17.06716561  W\_buong 14.30213027  W\_bus 11.86238326  W\_buti 11.86238326  W\_call 6.751101469  W\_calls 7.358830898  W\_casualty 16.96290317  W\_cellphones 7.358830898  W\_center 15.35700521  W\_centers 7.358830898  W\_circuit 17.33298031  W\_coast 11.28598359  W\_complex 6.751101469  W\_compound 11.82606216  W\_condo 10.25502678  W\_contact 13.62874487  W\_cor 14.34578703  W\_correspondent 11.28598359  W\_count 7.358830898  W\_countries 7.358830898  W\_cross 13.71974406  W\_cupang 10.25502678  W\_dagat 16.25364554  W\_dalampasigan 14.24918846  W\_dalawa 17.2971966  W\_daluyong 17.22163461  W\_damages 16.69868202  W\_davao 6.751101469  W\_dead 15.00268403  W\_dear 12.79362126  W\_declares 12.79362126  W\_deep 14.67223381  W\_delata 16.83938133  W\_delikado 16.83938133  W\_dibdib 14.99183707  W\_dilg 14.24918846  W\_dist 14.55976712  W\_district 14.32601167  W\_dito 14.54979868  W\_donasyon 7.358830898  W\_downed 15.00268403  W\_dpwh 7.358830898  W\_dry 14.99183707  W\_dswd 13.13643458  W\_dzmm 12.06569085  W\_eastern 15.11842142  W\_electrical 15.00268403  W\_elem 12.16895477  W\_emergency 15.94222303  W\_establisimyento 13.13643458  W\_estero 13.41685182  W\_evac 6.751101469  W\_evacuation 16.31910308  W\_evacuee 10.11022515  W\_extension 10.0817776  W\_facilitate 6.751101469  W\_families 12.93192301  W\_fifth 6.751101469  W\_flight 6.751101469  W\_flood 13.35479922  W\_floods 6.751101469  W\_floor 12.79362126  W\_follow 14.88147922  W\_forecast 14.24918846  W\_free 10.59839271  W\_friends 12.26952275  W\_front 11.28598359  W\_gabi 16.96290317  W\_galaw 15.53480254  W\_gawin 14.55976712  W\_generated 6.751101469  W\_globe 7.358830898  W\_gma 10.25502678  W\_going 14.95970567  W\_goods 17.05884859  W\_gov 13.27858367  W\_grabi 11.86238326  W\_grocery 7.358830898  W\_group 16.3192836  W\_guard 11.28598359  W\_gumagabi 13.13643458  W\_gusto 11.86238326  W\_gutter 14.61369849  W\_gym 15.94222303  W\_hagupit 17.25625022  W\_hala 6.751101469  W\_hall 14.55976712  W\_halos 11.28598359  W\_handang 7.358830898  W\_help 13.79992893  W\_helpdesk 15.94222303  W\_high 14.75845387  W\_hindi 12.91378046  W\_hinihikayat 7.358830898  W\_homes 11.86238326  W\_hotlines 14.67223381  W\_hway 6.751101469  W\_hydromet 11.28598359  W\_hypothermia 7.358830898  W\_iba 16.69868202  W\_ilikas 6.751101469  W\_imbak 16.83938133  W\_imprastraktura 15.94222303  W\_in 6.960314493  W\_inasuyan 7.358830898  W\_including 15.00268403  W\_indigo 7.358830898  W\_ingat 14.99183707  W\_inihahanda 12.06569085  W\_iniulat 11.28598359  W\_iniwan 15.94222303  W\_inyong 12.79362126  W\_ipamigay 11.28598359  W\_ipinagagamit 16.83938133  W\_ipriority 7.358830898  W\_isang 10.25502678  W\_isinailalim 16.4946584  W\_isinara 7.358830898  W\_isinasagawa 15.00268403  W\_island 14.97092052  W\_islands 15.66246385  W\_issue 11.28598359  W\_istruktura 17.43880434  W\_itinaas 17.44974971  W\_itinumba 11.28598359  W\_ito 13.73807256  W\_kaba 10.25502678  W\_kabayani 11.86238326  W\_kabuluan 10.25502678  W\_kahabaan 14.36078807  W\_kahoy 16.25364554  W\_kailangan 7.358830898  W\_kalupaan 14.24918846  W\_kalye 7.358830898  W\_kami 14.46535357  W\_kanto 6.751101469  W\_kapilya 16.83938133  W\_karangalan 14.55976712  W\_kasagsagan 7.358830898  W\_kasalukuyang 11.28598359  W\_kasi 14.46535357  W\_kasiglahan 6.751101469  W\_kasing 7.358830898  W\_kasunod 11.28598359  W\_katamtamang 7.358830898  W\_katao 15.00268403 | W\_kaugnay 7.358830898  W\_kaunti 6.751101469  W\_kawawa 10.25502678  W\_kaysa 14.24918846  W\_klase 17.23830091  W\_komunikasyon 15.54885154  W\_kph 17.43411441  W\_kuha 14.27654059  W\_kuryente 16.91226476  W\_labas 12.79362126  W\_lagpas 13.67392413  W\_lalake 11.28598359  W\_lalawigan 17.37984163  W\_lalim 13.41685182  W\_lates 7.358830898  W\_latest 13.13643458  W\_les 7.358830898  W\_light 11.28598359  W\_lilikas 7.358830898  W\_limandaang 6.751101469  W\_limay 13.41685182  W\_lines 16.69868202  W\_linya 15.54885154  W\_lipa 6.751101469  W\_list 11.67777091  W\_live 14.75845387  W\_liwasang 14.85443559  W\_local 7.358830898  W\_loob 17.42107339  W\_lubang 13.32088281  W\_luloy 10.25502678  W\_lumabas 16.83938133  W\_lumihis 15.94222303  W\_lumikas 15.36456403  W\_lunes 16.86782415  W\_maaapektuhan 7.358830898  W\_maaari 7.358830898  W\_maaaring 16.36704056  W\_maapektuhan 11.61537001  W\_maayos 7.358830898  W\_mabasa 11.86238326  W\_mag 16.83938133  W\_magbigay 7.358830898  W\_magdulot 13.41685182  W\_magkabilang 6.751101469  W\_magkumpuni 13.13643458  W\_maglandfall 14.24918846  W\_magsilikas 7.358830898  W\_magsisimulang 7.358830898  W\_magvolunteer 11.86238326  W\_mahabang 6.751101469  W\_makaiwas 17.33298031  W\_makatumba 7.358830898  W\_makita 6.751101469  W\_malalaking 7.358830898  W\_malawak 17.22163461  W\_maliliit 6.751101469  W\_malls 10.25502678  W\_mamayang 16.83938133  W\_mangingisda 11.28598359  W\_maramdaman 7.358830898  W\_marami 11.86238326  W\_mas 15.53480254  W\_masilungan 13.78626917  W\_materials 7.358830898  W\_matinding 12.26952275  W\_maulap 11.28598359  W\_mauubusan 7.358830898  W\_mayor 14.73522658  W\_meters 6.751101469  W\_metro 8.209819553  W\_motorista 11.86238326  W\_mula 13.1729139  W\_muling 15.53480254  W\_municipality 15.94222303  W\_nabaha 14.75184883  W\_nag 10.25502678  W\_nagbago 13.13643458  W\_nagdulot 15.47883756  W\_nagkansela 17.39020351  W\_nagpabuhat 11.86238326  W\_nagpapatupad 11.28598359  W\_nagsimula 13.13643458  W\_nagsisiksikang 11.28598359  W\_nagsitumbahan 11.28598359  W\_naibalik 11.28598359  W\_nailikas 7.358830898  W\_naitalang 17.44081472  W\_naka 7.358830898  W\_nakahanda 10.59839271  W\_nakakaiyak 6.751101469  W\_nakapasok 16.25364554  W\_nakataas 13.13643458  W\_nakatira 16.4946584  W\_nakatutok 15.00268403  W\_namataan 15.94222303  W\_namatay 13.13643458  W\_name 14.24918846  W\_nananatili 7.358830898  W\_nananatiling 14.36078807  W\_nanatili 13.13643458  W\_nang 17.13050486  W\_nangka 12.79362126  W\_naririnig 7.358830898  W\_nasawi 17.2562814  W\_nasira 17.43880434  W\_nating 7.358830898  W\_natitirang 11.28598359  W\_natrapped 10.25502678  W\_natuklap 13.13643458  W\_natuklapan 15.00268403  W\_navy 15.94222303  W\_nawasak 7.358830898  W\_nddrmc 7.358830898  W\_near 7.358830898  W\_needs 12.79362126  W\_ngaun 6.751101469  W\_ngayon 13.34645357  W\_ngayong 14.75979591  W\_ngcp 16.69868202  W\_ngiti 7.358830898  W\_ngyon 11.86238326  W\_ninyo 17.10733609  W\_noah 16.25364554  W\_norte 15.45638079  W\_northeast 11.28598359  W\_northeastern 15.00268403  W\_northern 16.74549957  W\_northwest 15.53480254  W\_not 14.91975572  W\_number 16.83938133  W\_numero 17.10733609  W\_of 12.75077961  W\_offers 7.358830898  W\_offices 10.11022515  W\_official 7.358830898  W\_open 7.217344415  W\_our 14.53482042  W\_out 11.86238326  W\_overnight 7.358830898  W\_pabugso 13.13643458  W\_packs 16.4946584  W\_padala 11.86238326  W\_pagkain 16.83938133  W\_paglilista 7.358830898  W\_paglusong 13.41685182  W\_pagpasok 13.13643458  W\_pahelp 13.41685182  W\_pahirap 6.751101469  W\_palibot 14.55976712  W\_pamimigay 7.358830898  W\_pananalasa 16.80653174  W\_paparating 7.358830898  W\_par 16.69868202  W\_para 15.79585525  W\_parking 10.25502678  W\_pasok 11.28598359  W\_patay 17.33298031  W\_patayin 17.33298031  W\_patungong 14.19831807  W\_paunan 7.358830898  W\_peligroso 7.358830898  W\_people 6.751101469  W\_pero 13.83547712  W\_personnel 17.22163461  W\_persons 7.358830898  W\_photo 15.40065961  W\_pinadapa 7.358830898  W\_pinalilikas 16.25364554  W\_pinapansin 6.751101469  W\_pinsala 17.43880434  W\_pls 9.78675059  W\_plywood 14.24918846  W\_pm 13.78626917  W\_po 10.30096691  W\_police 7.358830898  W\_pong 11.25127362  W\_port 11.28598359  W\_posibilidad 7.358830898  W\_posibleng 16.4946584  W\_poste 11.28598359  W\_power 16.69868202  W\_preemptive 15.00268403  W\_preso 15.53480254  W\_press 14.24918846  W\_private 14.99547006  W\_probinsya 14.59942123  W\_project 16.25364554  W\_providing 7.358830898  W\_pswd 15.94222303  W\_puddle 10.25502678  W\_pumasok 12.79362126  W\_punongkahoy 7.358830898  W\_purok 7.358830898  W\_pwedeng 13.41685182  W\_quad 6.751101469  W\_quiteria 6.751101469  W\_rainfall 15.88362636  W\_raw 6.751101469  W\_ready 15.53480254  W\_regional 14.24918846  W\_relief 16.76701904  W\_rescue 11.12833841  W\_rescuer 14.55976712  W\_rescueteam 11.28598359  W\_residente 16.18526873  W\_residenteng 7.358830898 | W\_responsibility 17.22201042  W\_rising 6.751101469  W\_river 14.55976712  W\_road 13.37649487  W\_rob 6.751101469  W\_roga 10.25502678  W\_roofs 6.751101469  W\_safe 14.55976712  W\_sagip 7.358830898  W\_sagkahan 15.8870092  W\_sainyo 11.86238326  W\_sakaling 16.54310717  W\_salamat 13.27858367  W\_samin 14.15221435  W\_sana 11.67777091  W\_sang 6.751101469  W\_sapa 6.751101469  W\_sapat 16.83938133  W\_sasakyan 12.79362126  W\_sasakyang 6.751101469  W\_save 10.79035832  W\_says 7.358830898  W\_school 11.65388519  W\_scout 6.751101469  W\_serbisyo 17.10733609  W\_service 10.25502678  W\_severe 10.25502678  W\_share 10.79035832  W\_sibuyan 16.4946584  W\_silangan 15.00268403  W\_simbahan 16.83938133  W\_simula 7.358830898  W\_sinira 15.94222303  W\_sir 13.41685182  W\_situation 7.358830898  W\_sitwasyon 14.88147922  W\_slex 6.751101469  W\_sobrang 11.28598359  W\_southern 15.15186544  W\_st 9.077456715  W\_state 15.53480254  W\_sto 7.71038382  W\_stop 10.25502678  W\_storm 16.95375724  W\_strong 14.52027052  W\_sub 6.751101469  W\_subalit 7.358830898  W\_subided 14.15221435  W\_subsided 14.99183707  W\_sulong 10.25502678  W\_sumusunod 17.10733609  W\_super 11.28598359  W\_supermalls 7.358830898  W\_suplay 15.53480254  W\_surge 16.95002486  W\_swimming 6.751101469  W\_system 10.25502678  W\_tahanan 12.79362126  W\_tala 11.28598359  W\_talaga 11.28598359  W\_tan 14.24918846  W\_tandaan 16.88229623  W\_tao 14.15221435  W\_tapat 10.25502678  W\_tatawaging 11.28598359  W\_tayo 14.9782785  W\_teleradyo 14.24918846  W\_teritoryo 13.13643458  W\_tinatayang 14.88147922  W\_tindahang 16.25364554  W\_tinupi 13.13643458  W\_today 16.48741185  W\_tong 6.751101469  W\_topples 15.00268403  W\_total 15.94222303  W\_town 11.86238326  W\_towns 16.25364554  W\_transistor 11.28598359  W\_transported 17.22163461  W\_tsansang 15.53480254  W\_tska 6.751101469  W\_tubig 14.99183707  W\_tuhod 14.93293014  W\_tulad 16.83938133  W\_tulay 14.75979591  W\_tumama 13.13643458  W\_tumana 8.80275224  W\_tumatanggap 7.358830898  W\_tumawag 16.4946584  W\_tumutok 17.38203517  W\_tuwing 17.22163461  W\_tweet 6.751101469  W\_types 14.99183707  W\_typhoon 17.11877859  W\_uhod 12.79362126  W\_ukol 14.67223381  W\_ulat 15.95313515  W\_ulit 7.358830898  W\_umabot 13.13643458  W\_umakyat 17.22163461  W\_umiwas 13.41685182  W\_units 7.358830898  W\_update 15.18667228  W\_updated 11.28598359  W\_updates 11.28598359  W\_utos 13.13643458  W\_vehicles 14.9782785  W\_victims 7.358830898  W\_volunteer 10.40035956  W\_wag 12.79362126  W\_walang 12.11924969  W\_warning 14.55976712  W\_waves 7.358830898  W\_winalis 7.358830898  W\_winasak 16.69868202  W\_wla 6.751101469  W\_yari 16.25364554  W\_year 11.28598359  W\_years 13.13643458  W\_yero 16.25364554  W\_you 9.077456715  W\_youscooper 17.44974971  W\_yung 15.00268403  W\_bubong 9.077456715  W\_food 13.68088483  W\_i 14.75184883  W\_iaccommodate 12.79362126  W\_ilang 11.61242166  W\_just 13.62874487  W\_kanina 12.79362126  W\_mahigit 12.79362126  W\_mataas 11.28598359  W\_more 12.69391489  W\_namin 7.358830898  W\_need 14.61369849  W\_nila 12.79362126  W\_pamilya 14.97850549  W\_point 13.41685182  W\_volunteers 6.751101469  W\_yellow 10.25502678  W\_all 14.99183707  W\_area 12.79362126  W\_bago 12.79362126  W\_bagong 13.41685182  W\_bagyo 13.41685182  W\_bahagi 14.36078807  W\_bandang 6.751101469  W\_barangay 11.25127362  W\_brgy 12.24199048  W\_calamity 10.25502678  W\_church 8.39094461  W\_city 13.68057124  W\_dahil 14.32601167  W\_dapat 14.55976712  W\_dulot 7.358830898  W\_evacuees 12.53579465  W\_everyone 11.86238326  W\_family 6.751101469  W\_ginawa 10.25502678  W\_good 14.55976712  W\_grabe 14.88151941  W\_habang 9.570496119  W\_handa 6.751101469  W\_hanggang 14.75184883  W\_harap 6.751101469  W\_highway 14.55976712  W\_humupa 6.751101469  W\_inilikas 8.705593154  W\_isla 7.358830898  W\_kong 14.6109891  W\_kung 14.85504145  W\_lahat 11.82606216  W\_lubog 6.751101469  W\_lugar 14.15221435  W\_lumakas 16.25364554  W\_magingat 14.67223381  W\_malapit 14.15221435  W\_may 12.90756331  W\_muna 6.751101469  W\_naman 14.75845387  W\_nasa 12.40281371  W\_news 11.86238326  W\_now 12.26952275  W\_paalala 14.85443559  W\_pagbaha 7.354701903  W\_papuntang 6.751101469  W\_passable 7.358830898  W\_patuloy 8.705593154  W\_post 16.69868202  W\_question 12.79646048  W\_red 6.751101469  W\_reinforce 10.25502678  W\_requests 10.17200895  W\_san 13.75250298  W\_sinarado 7.358830898  W\_subd 7.358830898  W\_than 14.24918846  W\_ulan 13.88812145  W\_via 7.537344055  W\_video 13.83547712  W\_village 14.07786302  W\_wala 6.993005333  W\_water 14.02339768  W\_weather 13.12667349  W\_with 6.751101469  W\_bukas 9.570496119  W\_kahit 6.751101469  W\_kayang 7.358830898 |

1. Extraction Rules

|  |
| --- |
| Caution and Advice |
| <pos:JJ> <pos:NN> <pos:PSNS> <number:ANY>  <ner:LOCATION>[as]LOCATION  <pos:JJ> <string:#1>  <pos:JJ> <string:#2>  <pos:JJ> <string:#3>  <pos:VBZ> <string:classes> <pos:IN> <pos:JJ> <pos:VBZ>  <pos:VBP> <string:classes> <pos:IN> <pos:JJ> <pos:VBZ>  <string:#walangpasok> <pos:JJ> <pos:VBZ>  <string:signal> <pos:NN> <pos:PSNS> <number:ANY>  <string:#walangpasok> <pos:PSNS> <string:klase>  <string:#walangpasok> <string:sa> <pos:PIDP> <pos:NA> <string:antas> |

|  |
| --- |
| Casualty and Damage |
| <number:ANY>[as]NUMBER <pos:NA> <pos:NA> <string:ANY> <ner:UNIT>[as]UNIT  <pos:NA> <pos:NCOM> <pos:NA> <pos:NCOM>  <number:ANY>[as]NUMBER <ner:UNIT>[as]UNIT  <ner:LOCATION>[as]LOCATION  <pos:NCOM> <pos:NA> <pos:NCOM>[as]OBJECT  <string:sa> <pos:PINP> <pos:NN>  <string:sa> <pos:JJ> <ner:LOCATION>[as]LOCATION  <string:state>[as]DETAIL <string:of>[as]DETAIL <string:calamity>[as]DETAIL  <pos:JJ> <ner:LOCATION>[as]LOCATION  <ner:LOCATION>[as]LOCATION <pos:NN>[as]LOCATION  <pos:VOBF>[as]DETAIL <pos:NA> <string:#RubyPH>[as]DETAIL  <pos:NA> <pos:NCOM>[as]OBJECT  <pos:ADUN>[as]DETAIL <pos:NCOM> <pos:NCOM>[as]DETAIL  <pos:NA> <pos:NN>[as]OBJECT  <pos:ADOT> <pos:NCOM>  <number:ANY>[as]DETAIL <string:na> <string:ang> <pos:JJ> <ner:UNIT>[as]OBJECT  <number:ANY>[as]DETAIL <pos:NNS>[as]UNIT  <pos:NCOM>[as]DETAIL <string:at>[as]DETAIL <pos:NCOM>[as]DETAIL  <string:walang>[as]DETAIL <string:naitalang> <ner:UNIT>[as]UNIT |

|  |
| --- |
| Call For Help |
| <pos:MANH> <pos:CONG> <pos:NCOM> |

|  |
| --- |
| Donation |
| <ner:LOCATION>[as]LOCATION  <pos:VOBF> <pos:NA> <pos:NA> <pos:NN>[as]DONATION  <number:ANY> <ner:UNIT>[as]UNIT  <number:ANY> <pos:NN:UN> <ner:UNIT>[as]UNIT <string:packs> |

1. Representation of Ontology in OWL Format

|  |
| --- |
| <?xml version="1.0"?>  <rdf:RDF xmlns="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#"  xml:base="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5"  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"  xmlns:owl="http://www.w3.org/2002/07/owl#"  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">  <owl:Ontology rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5"/>    <!--  ///////////////////////////////////////////////////////////////////////////////////////  //  // Object Properties  //  ///////////////////////////////////////////////////////////////////////////////////////  -->  <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#can\_be\_of\_the\_category -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#can\_be\_of\_the\_category">  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CallForHelp"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CasualtiesAndDamage"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CautionAndAdvice"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Donation"/>  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#donates -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#donates">  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Volunteer"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#from\_a -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#from\_a">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#gives\_out\_an -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#gives\_out\_an">  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Advice"/>  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CautionAndAdvice"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#has\_this\_information -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#has\_this\_information">  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#needs -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#needs">  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#reports\_a\_call\_for\_help\_from\_a -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#reports\_a\_call\_for\_help\_from\_a">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CallForHelp"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#reports\_a\_need\_for\_a -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#reports\_a\_need\_for\_a">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Donation"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Volunteer"/>  </owl:ObjectProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#reports\_damages\_to -->  <owl:ObjectProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#reports\_damages\_to">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CasualtiesAndDamage"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Object"/>  <rdfs:range rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  </owl:ObjectProperty>    <!--  ///////////////////////////////////////////////////////////////////////////////////////  //  // Data properties  //  ///////////////////////////////////////////////////////////////////////////////////////  -->    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#geoLocationOfTweet -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#geoLocationOfTweet">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#locationInTweet -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#locationInTweet">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#objectDetails -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#objectDetails">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Object"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#objectName -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#objectName">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Object"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#resourceDetails -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#resourceDetails">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#resourceName -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#resourceName">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetAdvice -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetAdvice">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Advice"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetContent -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetContent">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetDate -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetDate">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetHandle -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetHandle">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetTimestamp -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#tweetTimestamp">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#victimName -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#victimName">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#volunteerName -->  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#volunteerName">  <rdfs:domain rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Volunteer"/>  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>  </owl:DatatypeProperty>    <!--  ///////////////////////////////////////////////////////////////////////////////////////  //  // Classes  //  ///////////////////////////////////////////////////////////////////////////////////////  -->    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Advice -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Advice"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CallForHelp -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CallForHelp">  <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  </owl:Class>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CasualtiesAndDamage -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CasualtiesAndDamage">  <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  </owl:Class>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CautionAndAdvice -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CautionAndAdvice">  <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  </owl:Class>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Donation -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Donation">  <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  </owl:Class>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Object -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Object"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Volunteer -->  <owl:Class rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Volunteer"/>    <!--  ///////////////////////////////////////////////////////////////////////////////////////  //  // Individuals  //  ///////////////////////////////////////////////////////////////////////////////////////  -->    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CD-I\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CD-I\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CasualtiesAndDamage"/>  <reports\_damages\_to rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_OBJ-I\_ERV\_Elem\_School"/>  <reports\_damages\_to rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_ERV\_Elem\_School"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CFH-I\_pamilya\_ng\_mga\_sundalo -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CFH-I\_pamilya\_ng\_mga\_sundalo">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CallForHelp"/>  <reports\_a\_call\_for\_help\_from\_a rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_pamilya\_ng\_mga\_sundalo"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_D-I\_students\_of\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_D-I\_students\_of\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Donation"/>  <reports\_a\_need\_for\_a rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_RES-I\_students\_of\_ERV\_Elem\_School"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_OBJ-I\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_OBJ-I\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Object"/>  <objectDetails rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Binaha ang classrooms.</objectDetails>  <objectName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">ERV Elem School</objectName>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_RES-I\_students\_of\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_RES-I\_students\_of\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Resource"/>  <resourceDetails rdf:datatype="http://www.w3.org/2001/XMLSchema#string">1000 pcs</resourceDetails>  <resourceName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">notebooks and bags</resourceName>  <from\_a rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_students\_of\_ERV\_Elem\_School"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  <victimName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Mga estudyante ng ERV Elem School</victimName>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_pamilya\_ng\_mga\_sundalo -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_pamilya\_ng\_mga\_sundalo">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  <victimName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">pamilya ng mga sundalo</victimName>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_students\_of\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_VIC-I\_students\_of\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Victim"/>  <victimName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">students of ERV Elem School</victimName>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_A-I\_WARNING!\_Ito\_ay\_isang\_advice\_na\_tweet! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_A-I\_WARNING!\_Ito\_ay\_isang\_advice\_na\_tweet!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Advice"/>  <tweetAdvice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">WARNING! Ito ay isang advice na tweet!</tweetAdvice>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CA-I\_WARNING!\_Ito\_ay\_isang\_advice\_na\_tweet! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CA-I\_WARNING!\_Ito\_ay\_isang\_advice\_na\_tweet!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#CautionAndAdvice"/>  <gives\_out\_an rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_A-I\_WARNING!\_Ito\_ay\_isang\_advice\_na\_tweet!"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <geoLocationOfTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">418.582912321, 102.51084911</geoLocationOfTweet>  <locationInTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Brgy. Trese</locationInTweet>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <geoLocationOfTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">10.00000121, 145.345300023</geoLocationOfTweet>  <locationInTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Guadalupe</locationInTweet>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <geoLocationOfTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">9.011112321, 231.34569903</geoLocationOfTweet>  <locationInTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Makati City</locationInTweet>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Location"/>  <geoLocationOfTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">109.00540121, 45.378000003</geoLocationOfTweet>  <locationInTweet rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Quezon City</locationInTweet>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  <tweetContent rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Ang mga pamilya ng mga sundalo sa Brgy. Trese ay nangangailangan ng tulong!</tweetContent>  <tweetHandle rdf:datatype="http://www.w3.org/2001/XMLSchema#string">TheJPGarcia</tweetHandle>  <can\_be\_of\_the\_category rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CFH-I\_pamilya\_ng\_mga\_sundalo"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong!"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong!"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  <tweetContent rdf:datatype="http://www.w3.org/2001/XMLSchema#string">BWAHAHAHAHAHA RT WARNING! Ito ay isang tweet!</tweetContent>  <tweetHandle rdf:datatype="http://www.w3.org/2001/XMLSchema#string">theonlykyleeeee</tweetHandle>  <can\_be\_of\_the\_category rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CA-I\_WARNING!\_Ito\_ay\_isang\_advice\_na\_tweet!"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet!"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet!"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  <tweetContent rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Hi everyone! The students of ERV Elem School are in need of 1000 pcs of notebooks and bags!</tweetContent>  <tweetHandle rdf:datatype="http://www.w3.org/2001/XMLSchema#string">vilsonluuufffy</tweetHandle>  <can\_be\_of\_the\_category rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_D-I\_students\_of\_ERV\_Elem\_School"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags!"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags!"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Tweet"/>  <tweetContent rdf:datatype="http://www.w3.org/2001/XMLSchema#string">PRAY! RT Binaha ang classrooms ng mga estudyante ng ERV Elem School</tweetContent>  <tweetHandle rdf:datatype="http://www.w3.org/2001/XMLSchema#string">addicteduserrr</tweetHandle>  <can\_be\_of\_the\_category rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_CD-I\_ERV\_Elem\_School"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_LI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School"/>  <has\_this\_information rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School"/>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_Ang\_mga\_pamilya\_ng\_mga\_sundalo\_sa\_Brgy.\_Trese\_ay\_nangangailangan\_ng\_tulong!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <tweetTimestamp rdf:datatype="http://www.w3.org/2001/XMLSchema#string">03/21/2015:03:27:00:34</tweetTimestamp>  <tweetDate rdf:datatype="http://www.w3.org/2001/XMLSchema#string">March 21, 2015</tweetDate>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_BWAHAHAHAHAHA\_RT\_WARNING!\_Ito\_ay\_isang\_tweet!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <tweetTimestamp rdf:datatype="http://www.w3.org/2001/XMLSchema#string">12/27/2014:00:13:67:40</tweetTimestamp>  <tweetDate rdf:datatype="http://www.w3.org/2001/XMLSchema#string">December 27, 2014</tweetDate>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags! -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_Hi\_everyone!\_The\_students\_of\_ERV\_Elem\_School\_are\_in\_need\_of\_1000\_pcs\_of\_notebooks\_and\_bags!">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <tweetTimestamp rdf:datatype="http://www.w3.org/2001/XMLSchema#string">12/30/2014:01:44:09:10</tweetTimestamp>  <tweetDate rdf:datatype="http://www.w3.org/2001/XMLSchema#string">December 30, 2014</tweetDate>  </owl:NamedIndividual>    <!-- http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School -->  <owl:NamedIndividual rdf:about="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#\_TSI\_PRAY!\_RT\_Binaha\_ang\_classrooms\_ng\_mga\_estudyante\_ng\_ERV\_Elem\_School">  <rdf:type rdf:resource="http://www.semanticweb.org/kylemchalebdelacruz/ontologies/2014/9/untitled-ontology-5#Timestamp"/>  <tweetTimestamp rdf:datatype="http://www.w3.org/2001/XMLSchema#string">12/27/2014:00:14:47:24</tweetTimestamp>  <tweetDate rdf:datatype="http://www.w3.org/2001/XMLSchema#string">December 27, 2014</tweetDate>  </owl:NamedIndividual>  </rdf:RDF>  <!-- Generated by the OWL API (version 3.5.0) http://owlapi.sourceforge.net --> |

1. Resource Persons

**Nicco Louis S. Nocon**

NormAPI Proponent

noconoccin@gmail.com

**Nathaniel Oco**

Faculty Member

College of Computer Studies

National University

nathanoco@yahoo.com

**Ralph Vincent J. Regalado**

Thesis Adviser, Faculty Member

Software Technology Department

College of Computer Studies

De La Salle University

ralph.regalado@delasalle.ph

1. Personal Vitae

**Kyle Mc Hale B. Dela Cruz**

Blk 5, Lot 2A, Martires St., Brgy. Martires del 96, Pateros, Metro Manila

(0917) 880-5019

kylemchale\_delacruz@yahoo.com

**John Paul F. Garcia**

36 Bohol St., Ayala Alabang Village, Muntinlupa City

(0927) 886-6999

johnpaulgarcia1208@gmail.com

**Kristine Ma. Dominique F. Kalaw**

28 New Years Avenue, GSIS Holiday Hills Village, San Pedro, Laguna

(0927) 854-4201

tintin.kalaw@gmail.com

**Vilson E. Lu**

739-D A. Bonifacio St. Balintawak, Quezon City

(0917) 631-1374

vilson.espayos.lu@gmail.com

1. MicroMappers digital disaster response system. http://micromappers.com/ [↑](#footnote-ref-1)
2. The American Red Cross, *Web Users Increasingly Rely on Social Media to Seek Help in a Disaster*, Press Release, Washington, DC, August 9, 2010. http://newsroom.redcross.org/2010/08/09/press-release-web-users-increasingly-rely-on-social-media-to-seek-help-in-a-disaster/ [↑](#footnote-ref-2)
3. Official Gazette of the Republic of the Philippines, *Prepare for natural calamities: Information and resources from the government*, July 21, 2012. http://www.gov.ph/crisis-response/government-information-during-natural-disasters/ [↑](#footnote-ref-3)
4. Twitter, a microblogging social media platform. http://www.twitter.com/ [↑](#footnote-ref-4)