EXTRACTING AND ORGANIZING DISASTER-RELATED PHILIPPINE COMMUNITY RESPONSES FOR AIDING NATIONWIDE RISK REDUCTION PLANNING AND RESPONSE

User's Manual

A Master's Thesis Proposal
Presented to
the Faculty of the College of Computer Studies
Graduate Studies Program
De La Salle University – Manila

In Partial Fulfillment
of the Requirements for the Degree of
Master of Science in Computer Science

by
Nocon, Nicco Louis S.

Ms. Charibeth K. Cheng
Faculty Adviser

May 22, 2020

Contents

1 Introduction

This document provides instructions on how to use the Filipino text analysis tool. It is a Python Application Programming Interface (API) or library with a collection of functions. The API's main task is to extract valuable information or insights (specifically actions and target subjects) from a group of text, organize that information, and generate a report out of it. Modules were made and can be used for standalone processes.

1.1 System Requirements

This section lists the minimum hardware and software requirements needed to properly execute the system.

Table 1.1 Hardware Requirements

Hardware	Minimum	Recommended
Operating System	Windows 7 (32- or 64-bit)	Windows 10 (64-bit)
Processor	Intel ® Core™ i3	Intel ® Core™ i5
GPU	-	NVIDIA GeForce GTX 960M
RAM	4 GB	8 GB
Hard Disk Space	1.10 GB	9.5 GB[*]
Screen Resolution	1024x768	1024x768 or higher

Table 1.2 Software Package Requirements

Package	Version
boto	2.49.0
boto3	1.9.210
botocore	1.12.210
certifi	2019.6.16
chardet	3.0.4
docutils	0.15.2
et-xmlfile	1.0.1
gensim	3.8.0
idna	2.8
jdcal	1.4.1
jmespath	0.9.4
langid	1.1.6
lxml	4.4.1
nltk	3.4.4
numpy	1.17.0
openpyxl	2.6.2
pip	19.0.3
python	3.5
python-dateutil	2.8.0
python-docx	0.8.10
requests	2.22.0
s3transfer	0.2.1
scipy	1.3.1
setuptools	40.8.0
six	1.12.0
smart-open	1.8.4
strsim	0.0.3
urllib3	1.25.3

1.2 Conventions

This subsection presents different conventions used to depict elements in the API (e.g., folder names, functions, code snippets, etc.)

List of conventions include:

- Default font: Times New Roman, 11
- Application names: Default font, Bold and Italicized -- e.g., ***Sample***
- Links: Default font, Underlined -- e.g., https://www.sample.com/
- Steps/Directory: Acute angle (>), Georgia, 11 -- e.g., Step 1 > Step 2 > Step 3 > *Sample.txt*

[*] 7.4 GB for Cygwin/Moses Installations, 700 MB for PyCharm

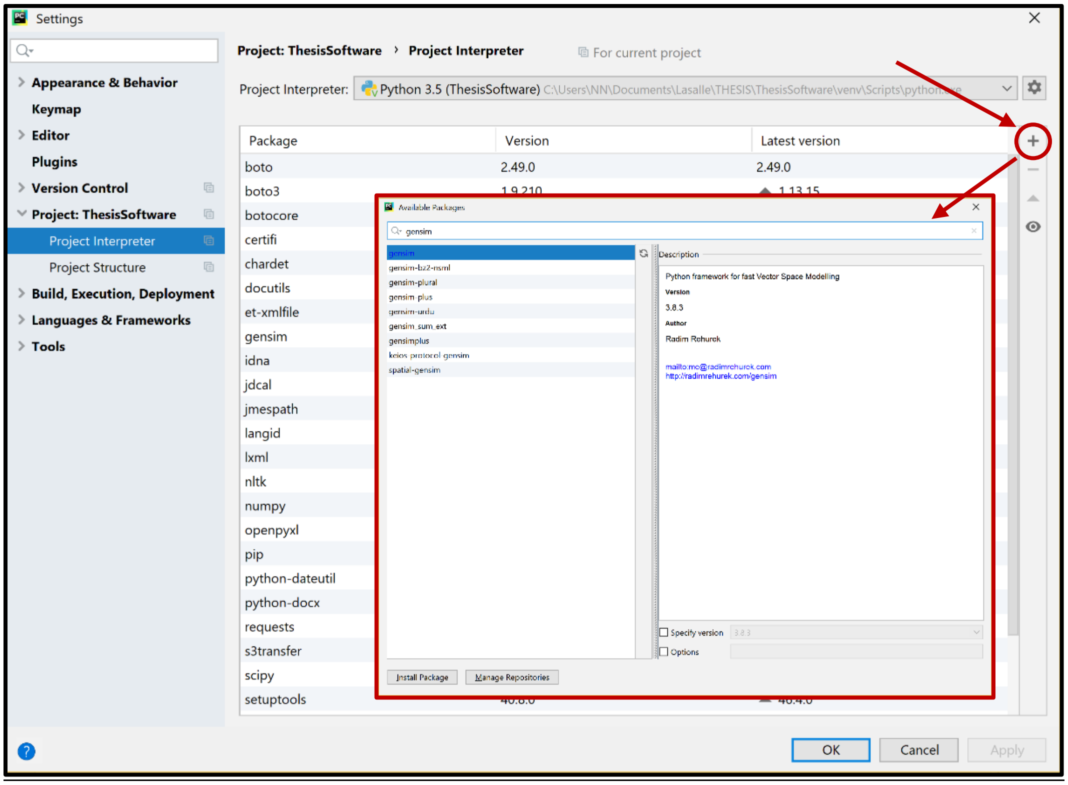
- Buttons: Enclosed with square brackets ([ ]) -- e.g., [sample]
- Folder/File names: Default font, Italicized -- e.g., *Sample.txt*
- Scripts: Segoe UI, 11 -- e.g., cd /sample
- Code Snippets: Consolas, 11 -- e.g., `import sample`
- Function names: Consolas, 11, Bold -- e.g., **`sample(param1, param2)`**

1.3 Installation

This subsection contains instructions on how to install the system, and the list necessary files and their respective directories. For best experience, use the API with ***PyCharm***, a free Python IDE for developers, and install ***Moses***, a statistical machine translation system for normalization tasks. In this subsection includes installation of both.

1.3.1 *PyCharm* and API

1) Visit and download executable file at its website: https://www.jetbrains.com/pycharm/download/
2) Run the ***PyCharm*** executable file (268 MB) and follow the steps as indicated.
3) Once installed, open ***PyCharm***.
4) Click File > Open... and choose the ***ThesisSoftware*** project.
5) After opening, check the required software packages if they are installed (File > Settings... > Project: ThesisSoftware > Project Interpreter).
6) If there are missing packages, click Install [+], look for the packages, and click [Install Package].



Project: ThesisSoftware › Project Interpreter

Project Interpreter: Python 3.5 (ThesisSoftware) C:\Users\NN\Documents\Lasalle\THESIS\ThesisSoftware\venv\Scripts\python.exe

Package	Version	Latest version
boto	2.49.0	2.49.0
boto3	1.9.210	1.13.15
botocore		
certifi		
chardet		
docutils		
et-xmlfile		
gensim		
idna		
jdcal		
jmespath		
langid		
lxml		
nltk		
numpy		
openpyxl		
pip		
python-dateutil		
python-docx		
requests		
s3transfer		
scipy		
setuptools	40.8.0	46.4.0

Available Packages: gensim

gensim
gensim-bz2-nsml
gensim-plural
gensim-plus
gensim-urdu
gensim_sum_ext
gensimplus
keios-protocol-gensim
spatial-gensim

Description

Python framework for fast Vector Space Modelling

Version

3.8.3

Author

Radim Rehurek

mailto:me@radimrehurek.com
http://radimrehurek.com/gensim

□ Specify version 3.8.3
□ Options

7) Once completed, API is ready to be used.

1.3.2 *Moses*

In order to use other functions in the normalization module, ***Moses*** statistical machine translation system must be installed. Documents that can support the installation procedure can be found on /doc/Moses Installation Guides/ folder. Here are the shortened instructions to install it on Windows OS:

1) Download necessary resources:
    - ***Cygwin***: https://cygwin.com/install.html
    - ***Giza-pp***: https://github.com/moses-smt/giza-pp
    - ***SRILM***: http://www.speech.sri.com/projects/srilm/download.html
    - ***Moses***: https://github.com/moses-smt/mosesdecoder
2) Open ***Cygwin*** setup and install default packages.
3) On top of the packages, install additional packages required by the applications above such as:
    - boost
    - bz
    - bzip
    - gcc
    - libboost
    - make
    - zlib
    - SRILM Requirements: http://www.speech.sri.com/projects/srilm/download.html
    - Moses Requirements: http://www.statmt.org/moses/?n=Development.GetStarted
4) After all packages have been installed, open *Cygwin.bat* for first time initialization.

To install ***Giza-pp***,

5) Type and enter the directory of Giza-pp (e.g., cd /giza-pp).
6) Install Giza-pp by typing and entering make all. Then, wait for installation to be completed.

To install *Boost*,

7) Type and enter the directory of Boost (e.g., cd /boost_1_54_0).
8) Install Boost by typing and entering:
    - ./bootstrap.sh
    - ./bjam
    - ./bjam install

    Then, wait for installation to be completed.

To install ***SRILM***,

9) Type and enter the directory of ***SRILM*** (e.g., cd /srilm).
10) Extract contents by typing and entering tar zxvf srilm.tgz
11) Exit ***Cygwin***.
12) Modify <your **Cygwin** path>\home\<your name>\.*bashrc* file and add:

```
export SRILM=/srilm
export MACHINE_TYPE=cygwin
export PATH=$PATH:$pwd:$SRILM/bin/cygwin
export MANPATH=$MANPATH:$SRILM/man
```

13) Modify <your **Cygwin** path>\ srilm\Makefile file and add:

```
SRILM = /srilm
```

14) Modify <your **Cygwin** path>\ srilm\common\Makefile.machine.cygwin file and replace:

```
From:                             To:

# Tcl support (part of cygwin)    # Tcl support (part of cygwin)
TCL_INCLUDE =                     NO_TCL = X
TCL_LIBRARY = -ltcl84             TCL_INCLUDE =
                                  TCL_LIBRARY =
```

15) Open *Cygwin.bat*

16) Type and enter the directory of SRILM (e.g., cd /srilm).

17) Install Giza-pp by typing and entering:

- make World
- make all
- make cleanest

Then, wait for installation to be completed.

To install ***Moses***,

18) Type and enter the directory of ***Moses*** (e.g., cd /mosesdecoder-master).

19) Install ***Moses*** by typing and entering:

- ./bjam --with-boost=C:/cygwin/usr/local --with-srilm=C:/cygwin/srilm -a

Then, wait for installation to be completed. Note: installation or configurations can be modified.

20) After installing, create a “tools” folder inside ***Moses*** (e.g., /mosesdecoder-master/tools/).

21) Insert a copy of GIZA++.exe, mkcls.exe and snt2cooc.out (these three can be found under ***Giza-pp***’s folder) inside the folder, as they would be accessed on normalization processes.

22) Finally, insert the following directories in Environment Variables > System Variables > Path:

C:\cygwin\mosesdecoder-master\bin\;C:\cygwin\bin\;C:\cygwin\srilm\bin\cygwin\;

(change the C:\cygwin path relative to the correct file location)

2 Getting Started

Since this research produced a collection of functions, all that is needed to do in order to use its modules is for a user to import the API and call its functions. There are a few ways to do this, one of which is:

1) Open ***PyCharm***.
2) Click File > Open... (for existing projects) or File > New Project... (for new ones).
3) Under File > Settings... > Project: ThesisSoftware > Project Structure, click [Add Content Root] and select the API.
4) Import the modules and call the functions. For example:

```
import fspost
fspost.set_java_path("")
print(fspost.tag_string('Saan ka pupunta?'))
```

Major API modules are listed on the succeeding sections, that is Information Extraction and Clustering. More and complete discussions about the modules and its resources are provided at /doc/*Thesis Technical Manual.pdf*. Modules that are expected on that document are Data Utilities, Normalization, Language Identification, Part-of-Speech Tagging, Information Organization, Information Ranking, and Report Generation.

3 Information Extraction

Information Extraction module contains only 2 functions. These functions enable the user to extract insights in two formats. One is insight phrases which extracts a string starting from a Verb up to a Noun. Another does the same process but formats it with only the Verb and Nouns inside a sub-list or tuple. It is safe to note that on this module, the extractions were made specific to Malasakit responses (dataset used in research). Extractions are performed in an object with the following attributes: Response ID, the Response itself, its Response Category, Language identifier, Filipino Part-of-Speech tags, container for the extracted insights, and location (a field that can be added by users).

MalasakitResponse Object

Attributes Name	Type	Description
response_id	Integer	A number indicating a response's order in the data (row number).
response	String	A string containing a response.
tag	String	A string indicating a response's category.
fspost_output	Tuple	Filipino Stanford Part-of-Speech Tagger (word, tag) tuple output.
fspost_stanford_format	String	Filipino Stanford Part-of-Speech Tagger word|tag Stanford notation.
pos	String	Filipino Stanford Part-of-Speech Tagger ‘tags only’ string.
insights_phrase	List	List of insights extracted from a response.
insights_words	List	List of lists of words (action, target, ...) insights extracted from a response.
location	String	String holder for a response’s location (can be added by users).
language	String	Language identifier of the response (e.g., tl = Tagalog, en = English).

Information Extraction Module: Functions List

Function Name	Description	Arguments	Return Type
extract_insights_phrases	Extracts phrase insights (action word to target/s). The MalasakitResponse object is updated after.	malasakit_response_list (list): The list containing MalasakitResponse objects with responses to be extracted.	Void (updates MalasakitResponse Object)
extract_insights_words	Extracts word insights or word sets (action word and target/s). The MalasakitResponse object is updated after.	malasakit_response_list (list): The list containing MalasakitResponse objects with responses to be extracted.	Void (updates MalasakitResponse Object)

Sample Code

```
# Task: Build a MalasakitResponse object.
import malasakit_response  # Import object and modules.
import fspost
import lang_id
import extract

fspost.set_java_path("")  # Initializes FSPOST.

# Set values.
response = 'Maglinis ng mga kanal at kalye o itapon ang mga basura'
response_object = malasakit_response.MalasakitResponse(1, response, 'Sanitation')
response_object.fspost_output = fspost.tag_string(response_object.response)
response_object.fspost_stanford_format = fspost.format_stanford(response_object.fspost_output)
response_object.pos = fspost.format_pos(response_object.fspost_output)
response_object.location = 'Manila, Philippines'
response_object.language = lang_id.identify_language_string(response_object.response)[0]
extract.extract_insights_phrases([response_object])  # Information Extraction.
extract.extract_insights_words([response_object])

# Display values.
print('Response ID: ', response_object.response_id)
print('Response: ', response_object.response)
print('Category: ', response_object.tag)
print('FSPOST Tuple: ', response_object.fspost_output)
print('FSPOST Stanford: ', response_object.fspost_stanford_format)
print('FSPOST POS only: ', response_object.pos)
print('Insight Phrases: ', response_object.insights_phrase)
print('Insight Word Sets: ', response_object.insights_words)
print('Location: ', response_object.location)
print('Language: ', response_object.language
```

Output

```
Response ID: 1
Response: Maglinis ng mga kanal at kalye o itapon ang mga basura
Category: Sanitation
FSPOST Tuple: [('Maglinis', 'VBW'), ('ng', 'CCB'), ('mga', 'DTCP'), ('kanal', 'NNC'), ('at', 'CCA'),
('kalye', 'NNC'), ('o', 'CCT'), ('itapon', 'VBTF'), ('ang', 'DTC'), ('mga', 'DTCP'), ('basura', 'NNC')]
FSPOST Stanford: Maglinis|VBW ng|CCB mga|DTCP kanal|NNC at|CCA kalye|NNC o|CCT itapon|VBTF ang|DTC
mga|DTCP basura|NNC
FSPOST POS only: VBW CCB DTCP NNC CCA NNC CCT VBTF DTC DTCP NNC
Insight Phrases: [1, 'Maglinis ng mga kanal at kalye', 'itapon ang mga basura']
Insight Word Sets: [[1, 'Maglinis', 'kanal', 'kalye'], [1, 'itapon', 'basura']]
Location: Manila, Philippines
Language: tl
```

4 Information Clustering

Information Clustering module contains 11 functions. Three parts can be taken from this list of functions. First is the main function that invokes the clustering algorithm (i.e., Dice's Coefficient, Word2Vec, or FastText). Then, supporting functions that can retrieve insights from the Malasakit object, remove duplicates in clusters, flatten insights in the cluster, and cluster/lexicalize target/noun words. Last is a list of functions that computes for distance or similarity values between two strings using the selected approach.

Information Clustering Module: Functions List

Function Name	Description	Arguments	Return Type
string_similarity_fasttext	Computes FastText's vector similarity (how close) between two strings.	string1 (str): The string to be compared to. string2 (str): The string to be compared to.	similarity: resulting score of pairs based on how close they are from each other (higher value is better).
string_distance_fasttext	Computes FastText's vector distance (how far) between two strings.	string1 (str): The string to be compared to. string2 (str): The string to be compared to.	distance: resulting score of pairs based on how far they are from each other (lower value is better).
string_similarity_word2vec	Computes Word2Vec's vector similarity (how close) between two strings.	string1 (str): The string to be compared to. string2 (str): The string to be compared to.	similarity: resulting score of pairs based on how close they are from each other (higher value is better).
string_distance_word2vec	Computes Word2Vec's vector distance (how far) between two strings.	string1 (str): The string to be compared to. string2 (str): The string to be compared to.	distance: resulting score of pairs based on how far they are from each other (lower value is better).
string_similarity_dice	Computes Dice's Coefficient similarity (how close) between two strings.	string1 (str): The string to be compared to. string2 (str): The string to be compared to.	similarity: resulting score of pairs based on how close they are from each other (higher value is better).
string_distance_dice	Computes Dice's Coefficient distance (how far) between two strings.	string1 (str): The string to be compared to. string2 (str): The string to be compared to.	distance: resulting score of pairs based on how far they are from each other (lower value is better).
collect_all_insights_from_object	Retrieves all insights in the MalasakitResponse object and stores them in one list.	malasakit_response_list (list): The list containing MalasakitResponse objects.	insights_list: a list containing all insights taken from the object list.

Function Name	Description	Arguments	Return Type
		insights_type (str): A character/string indicating the type of insights to be collected. 'p' for phrases and 'w' for word sets.	
`merge_cluster_insights`	Merges the insights in one cluster into a single line.	cluster (list): The list containing the current cluster. clustering_technique (str): Select a clustering technique from the following: 'dice', 'word2vec', or 'fasttext'.	cluster: a list containing modified (merged) words in the cluster's insights.
`remove_duplicate`	Removes duplicate strings in the list (cluster).	cluster_zero (list): The list containing the current cluster.	filtered_cluster_zero: a list containing the modified (filtered-off duplicates) words in the cluster.
`cluster_words`	Clusters target/noun words. Given a list it will join similar words using the ‘word1 (word2, ..., wordN)’ notation.	target_word_list (list): The list containing the words to be clustered. clustering_technique (str): Select a clustering technique from the following: 'dice', 'word2vec', or 'fasttext'.	new_target_word_list: a list containing the clustered and formatted words.
`cluster_information`	Clusters text using either Sørensen-Dice Coefficient (String Clustering), Word2Vec, or FastText Word Embeddings (Semantic Clustering) and returns a list of clusters.	malasakit_response_list (list): The list containing the MalasakitResponse objects. clustering_technique (str): Select a clustering technique from the following: 'dice', 'word2vec', or 'fasttext'.	clusters_list: a list containing the clustered insights.

Sample Code

```
# Task: Compare two words using the three clustering approaches.
import cluster  # Import modules.

# Input Strings.
string1 = 'malinis'
string2 = 'kalinisan'

# Information Clustering.
# Using Dice.
print('Dice:', cluster.string_similarity_dice(string1, string2))

# Using Word2Vec.
try:
    similarity = cluster.string_similarity_word2vec(string1, string2)
except KeyError:
    similarity = 0.0  # No operation done if not on the model, so set similarity to 0
print('Word2Vec:', similarity)

# Using FastText.
try:
    similarity = cluster.string_similarity_fasttext(string1, string2)
except KeyError:
    similarity = 0.0  # No operation done if not on the model, so set similarity to 0
print('FastText:', similarity)
```

Output

```
Dice: 0.6666666666666666
Word2Vec: 0.7630582
FastText: 0.623511
```

5 Messages

This section lists all system messages – error message, status message, information, and instruction message – that the user may encounter while using the system.

5.1 Error Messages

Message	FileNotFoundError: [Errno 2] No such file or directory: 'test.xlsx'
Description	Missing file or location.
Action	Make sure that the file is available and on the specified path.

Message	PermissionError: [Errno 13] Permission denied: 'test/test.xlsx'
Description	Restricted access on the file.
Action	Close the file before processing it.

Message	OSError: [Errno 22] Invalid argument: 'test\test.xlsx'
Description	File path format uses special character (escape) sequence '\t'
Action	Use alternative formatting such as '\\', '/' or case changes (e.g., test/test.xlsx)

<table>
<tr><td><b>Message</b></td><td>KeyError: "word 'tinalon' not in vocabulary"</td></tr>
<tr><td><b>Description</b></td><td>Given word is not in the model.</td></tr>
<tr><td><b>Action</b></td><td>Add error handling such as:
<pre>
try:
    similarity = cluster.string_similarity_word2vec(string1, string2)
except KeyError:
    similarity = 0.0
</pre>
It has not been included in string similarity/distance functions as this error could be useful for other intentions (e.g., collecting out-of-vocabulary in Tagalog model).</td></tr>
</table>

Message	AttributeError: 'list' object has no attribute 'collect_all_insights_from_object'
Description	There are no functions that could be used on the current object.
Action	Object types and structure should be the same with requirements from functions.

Message	ZeroDivisionError: division by zero
Description	Value of denominator in the computation is 0.
Action	Add error handling for ZeroDivisionError or adjust values through other means.

Message	LookupError: === NLTK was unable to find the java file! Use software specific configuration paramaters or set the JAVAHOME environment variable. ===
Description	Path of the tagger model is not found.
Action	Add `set_java_path` before calling tagger functions.

Message	ValueError: Unknown language code zz
Description	Language code to be set is invalid.
Action	All language codes given should follow ISO 639-1.

Message	Can't read C:\Users\...\[Nokhonfusion]-Filipino-Colloquialism-MT\model\moses.ini
Description	Moses.ini model configuration file is missing from the given file path.
Action	Make sure that model folder is complete and on the proper location.

Message	The system cannot find the path specified.
Description	Command line script to run ***Moses*** did not find the application in the system.
Action	Make sure that ***Moses*** is properly installed with file locations followed.

Message	Sentence Mismatch. Check Documents and Repeat the Test.
Description	Files to be evaluated does not match with each other.
Action	Make sure that files are in proper format and have the same length.

5.2 Status Messages

Message	Java path set by default
Description	Part-of-Speech tagger model file path has been successfully set to default. Tagger functionalities can be accessed.

Message	Defined parameters (per moses.ini or switch): config: C:\Users\...\model\[Nokhonfusion]-Filipino-Colloquialism-MT\model\moses.ini distortion-limit: 0 feature: UnknownWordPenalty WordPenalty PhrasePenalty PhraseDictionaryMemory name=TranslationModel0 table-limit=20 num-features=4 path=/translate/model/phrase-table.gz input-factor=0 output-factor=0 Distortion SRILM name=LM0 factor=0 path=/translate/train.dec.lm order=3 input-factors: 0 mapping: 0 T 0 weight: UnknownWordPenalty0= 1 WordPenalty0= -1 PhrasePenalty0= 0.2 TranslationModel0= 0.2 0.2 0.2 0.2 Distortion0= 0.3 LM0= 0.5 /mosesdecoder-master/bin line=UnknownWordPenalty FeatureFunction: UnknownWordPenalty0 start: 0 end: 0 line=WordPenalty FeatureFunction: WordPenalty0 start: 1 end: 1 line=PhrasePenalty FeatureFunction: PhrasePenalty0 start: 2 end: 2 line=PhraseDictionaryMemory name=TranslationModel0 table-limit=20 num-features=4 path=/translate/model/phrase-table.gz input-factor=0 output-factor=0 FeatureFunction: TranslationModel0 start: 3 end: 6 line=Distortion FeatureFunction: Distortion0 start: 7 end: 7 line=SRILM name=LM0 factor=0 path=/translate/train.dec.lm order=3 FeatureFunction: LM0 start: 8 end: 8 Start loading text SCFG phrase table. Moses format : [0.000] seconds Reading /translate/model/phrase-table.gz ----5---10---15---20---25---30---35---40---45---50---55---60---65---70---75---80---85---90---95--100 ** IO from STDOUT/STDIN Created input-output object : [3.000] seconds Translating line 0 in thread id 0x22190570 Translating: cge n nga Line 0: Collecting options took 0.000 seconds Line 0: Search took 0.000 seconds BEST TRANSLATION: sige na nga [111] [total=-4.589] core=(0.000,-3.000,3.000,-2.002,-2.076,-0.278,-0.247,0.000,-14.536) Line 0: Translation took 0.000 seconds total Name:moses VmRSS:63836 kB RSSMax:63860 kB user:3.000 sys:0.062 CPU:3.062 real:3.041
Description	Running ***Moses*** shows progress in normalizations.

Message	Setting up resources... Java path set by default Resources set! Elapsed time: 0.07712820000000001 Identifying Language... Language Identification done! Elapsed time: 2.7345193 Tagging POS... 1 / 14 2 / 14 3 / 14 4 / 14 5 / 14 6 / 14 7 / 14 8 / 14 9 / 14 10 / 14 11 / 14 12 / 14 13 / 14 14 / 14 POS Tagging done! Elapsed time: 10.276082599999999 Extracting Information... No insights (phrase) extracted at Response #: 14 Phrases done! Elapsed time: 0.00018329999999997753 No insights (words) extracted at Response #: 14 Word Set done! Elapsed time: 0.00014859999999992771 Information Extraction done! Elapsed time: 0.00035109999999996045 Clustering... Clustering done! Elapsed time: 0.005193800000000692 Ranking... Clustering done! Elapsed time: 9.90000000022917e-06 Generating Report... Report Generation done! Elapsed time: 0.1566964000000013 PROGRAM TIME: 13.3210286
Description	Full run of the API following the research’s architectural diagram displays the following status updates: resources setup, progress of the modules (i.e., start and end of execution, tagging update on current sentence, and sentences without extractions), and runtime.

5.3 Information Messages

Message	Compute Precision 1.0	Compute Recall 1.0	Compute Accuracy 1.0	Compute F-Measure 1.0
Description	Indicates the used evaluation metric.			

Message	Evaluate IE Phrases Compute Precision Compute Recall Compute Accuracy Compute F-Measure EVALUATION RESULTS -- Gold Standard Extraction Count: 1170 System Extraction Count: 1363 Total Possible Extraction Count: 1657 -- Complete Matches: 302 0.18225709112854557 Over-extractions: 149 0.08992154496077248 Under-extractions: 414 0.24984912492456246 Overlapping-extractions: 11 0.006638503319251659 Complete Mismatches: 781 0.4713337356668678 -- True Positive (TP): 876 False Positive (FP): 487 False Negative (FN): 294 True Negative (TN): 0 -- Precision: 0.6426999266324285 Recall: 0.7487179487179487 Accuracy: 0.5286662643331321 F-Measure: 0.6916699565732333 -- Word Count: 11906 True Positive (TP) Word Count: 4832 False Positive (FP) Word Count: 2097 False Negative (FN) Word Count: 1273 True Negative (TN) Word Count: 3704 -- Precision Word Count: 0.6973589262519844 Recall Word Count: 0.7914823914823915 Accuracy Word Count: 0.7169494372585251 F-Measure Word Count: 0.7414454503605953 --
Description	Running `compare_ie_phrases` show statistics of the evaluation.

Message	EVALUATION LISTS Complete Matches: ['1 / magkaisa dapat ang mga tao / magkaisa dapat ang mga tao', ..., '933 / add more drainage systems / add more drainage systems'] Over-Extractions: ['10 / sinasabi sa kung ano ang dapat gawin paghandaan ang lahat ng bibitbitin sa tuwing may sakuna / pagsunod sa sinasabi', ..., '932 / be awarespread info / spread info'] Under-Extractions: ['4 / wastong pagtatapon / pagtatapon ng basura', ..., '931 / rumesponde sa mga sakuna / papaano rumesponde sa mga sakuna'] Overlapping-Extractions: ['37 / kaylangan maging aware / maging aware sa balita', ..., '913 / maghanda nang pundo ang baranggay / maghanda nang pondo ang baranggay'] Complete Mismatches: ['3 / magkikita sa panahon', ..., '934 / awareness of every filipino'] True Positives: ['1 / magkaisa dapat ang mga tao / magkaisa dapat ang mga tao', ..., '933 / add more drainage systems / add more drainage systems'] False Positives: ['3 / magkikita sa panahon', ..., '934 / be introduce to the society'] False Negatives: ['4 / paglilinis ng kanal', ..., '934 / awareness of every filipino'] True Negatives: []
Description	Running **`compare_ie_phrases`** provide segregated lists for analysis.

Message	Evaluate IE Word Sets Compute Precision Compute Recall Compute Accuracy Compute F-Measure EVALUATION RESULTS -- Gold Standard Extraction Count: 1239 System Extraction Count: 1367 Total Possible Extraction Count: 1239 -- Exact Matches: 213 0.17191283292978207 Partial Matches: 309 0.24939467312348668 Action/Verb Matches: 109 0.08797417271993543 Target/Noun Matches: 188 0.15173527037933818 Crossover Matches: 29 0.023405972558514933 No Matches (System Output on Gold Standard): 519 0.4188861985472155 No Matches (Gold Standard on System Output): 321 0.25907990314769974 -- True Positive (TP): 848 False Positive (FP): 519 False Negative (FN): 321 True Negative (TN): 0 -- Precision: 0.6203365032918801 Recall: 0.7254063301967494 Accuracy: 0.5023696682464455 F-Measure: 0.668769716088328 --
Description	Running **`compare_ie_word_sets`** show statistics of the evaluation.

Message	EVALUATION LISTS Exact Matches: [[1, 'magkaisa', 'tao'], ..., [933, 'add', 'drainage', 'systems']] Partial Matches: ["[7, 'magkaroon', 'early', 'warning'] / [7, 'magkaroon', 'early', 'warning', 'system']", ..., "[928, 'be', 'posters'] / [928, 'be', 'posters', 'how', 'prepared']"] Action/Verb Matches: ["[16, 'maging', 'bagay'] / [16, 'maging', 'alerto']", ..., "[924, 'simulan', 'baranggay'] / [924, 'simulan', 'barangay', 'pagsugpo', 'kurapsyon']"] Target/Noun Matches: ['[13, \'ibat\', \'paraan\'] / [13, \'magkaroon\', ..., "[915, 'canned', 'goods'] / [915, 'prepare', 'canned goods', 'first aid kits']"] No Matches (System Output on Gold Standard): [[3, 'magkikita', 'panahon'], ..., [934, 'be', 'society']] No Matches (Gold Standard on System Output): [[4, 'paglilinis', 'kanal'], ..., [934, 'awareness', 'every', 'filipino']]
Description	Running **`compare_ie_word_sets`** provide segregated lists for analysis.