Comparison of Common Part-of-Speech Tagging Techniques Applied to Waray-waray Text

Fernando Ejorcadas Quiroz Jr.
Biliran Province State University
6 PUBLICATIONS 1 CITATION

Robert Roxas
University of the Philippines
10 PUBLICATIONS 2 CITATIONS

Comparison of Common Part-of-Speech Tagging Techniques Applied to Waray-waray Text

Fernando E. Quiroz, Jr. [1, *], and Robert R. Roxas [2]
[1]Department of Information Technology Education, Naval State University, Philippines
[2]Department of Computer Science, University of the Philippines – Cebu, Avenue, Philippines

Abstract - **This paper presents the result of comparing common Part-of-Speech tagging techniques applied to the Waray-waray language. Experiment involved the testing of the manually tagged Waray-waray corpus applying the four commonly used Part-of-Speech tagging algorithms namely: N-gram, TnT, Naïve Bayes, and Brill taggers. The experiment showed that the TnT tagger is most promising for the Waray-waray language.**

Keywords - part-of-speech tagging; algorithms; nlp

I. INTRODUCTION

Part-of-Speech (POS) tagging is one of the major task in Natural Language Processing (NLP) that is being used in variety of applications like grammar checkers, word disambiguation, tokenization, machine translation, and a lot more real-world problems relating to natural language understanding. Tagging refers to the method of assigning the corresponding part of speech to the words in a text (corpora). Most of the NLP researchers in the Philippines are working on Tagalog and Cebuano, which are the two major languages in the Philippines, and little has been done on the remaining 169 living languages/dialects in the country. Waray-waray is one of the top languages in the Philippines with over 2.4 million speakers and shares 4.6% of the total population covering from Samar and some parts of Leyte and Biliran [1]. With this, a customized POS tags suitable for the Waray-waray language was created and an exploration of a POS tagger suitable to the language was conducted using the fundamental and available most commonly used methods of tagging and the results were compared.

II. METHODS

The corpus utilized is a collection of religion and literature texts in Waray-waray language taken from PALITO project, which is an online repository of Philippine Languages [1]. Markup tags on the dataset were removed for these were not needed in doing the POS tagging task. The text was further broken down into sentences, where each sentence was broken down into tokens separated by spaces, and words were assumed as grammatically and semantically correct.

The final corpus consisted of 81,737 tokens. Each word was manually tagged with their corresponding POS value based on their context on the sentence. Each token was separated into two: the token/word itself (left) and a capitalized code that represents the POS tag (right). These components were separated by the symbol '|'.

Tagset used in the tagging followed the Penn Treebank tags [2], but were modified to fit the Waray-waray language such as adding a plural quantifier and removing comparative adjective. In totality, there were 33 unique tags used, wherein the right column represents the code tag, while the left column is the description of the tag. Before the corpus is fed to the taggers, it was split into 70% and 30% for training and testing tests, respectively. The 70% training set was fed to the taggers for training, where it generates a model out from it. The models were evaluated using the 30% testing

[*] fernando.quiroz@nsu.edu.ph

set. Accuracy of each tagger was computed by getting the percentage of the correct tags produced by the tagger against the original tags of the test set.

The corpus was first tested in an *n-gram* tagger, is a type of probabilistic language model for predicting the next item in such a sequence in the form of a (n−1) – order Markov model [3]. The tagged corpus was first tested using unigram, bigram, and trigram patterns. Unigrams only considered the current token, in isolation from the larger context. With bigram, it has an advantage of tagging the words in a context-based manner using bigram patterns. Trigram tagger can correctly handles token belonging to trigram pattern. Although bigram and trigram were appropriate to use in context-based tagging, both cannot handle words not belonging to their corresponding *n*-gram patterns. To address the trade-off between accuracy and coverage of each *n*-gram, these three were combined to handle the back-off of the other. The *N*-gram returns ***w*** with an assurance that all tokens were tagged. By combining the three n-gram patterns, the accuracy has improved reaching to 90.29%.

Next, the tagged corpus was tested using the TnT tagger, which is the short term of *Trigrams 'n' Tags*, a statistical Part-of-Speech tagger that is trainable on different languages and virtually any tagged set [4]. The tagger was an implementation of the Viterbi algorithm for second order Markov models, which uses linear interpolation as the main paradigm used for smoothing. The TnT tagger also maintains the beam search parameter *N* that controls the number of possible solutions the tagger maintains while trying to guess the tags for a sentence. With a much higher *N*, it will greatly increase the memory used during tagging while a much lower *N* could decrease accuracy. We tested the corpus using the default 1000 and had it increased by trial and error method to 2000. Also, we tried to lower *N* by 500, 200, 100, 50, 25, 10, and 5. The changes of N from 1000 to 2000, 500, 200, 100, 50, 25 and 10 showed that as the value of N increases, the higher memory it consumed and runs slower. There is no change in the accuracy until it reached to 5, wherein the accuracy goes down to 90.98%.

After then, Naïve Bayes was used for testing the tagged set. It implemented a feature detector that combined many of the techniques of many taggers into a single feature set [5]. Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. During training, a feature extractor was used to convert each token ***w*** from the training data to a feature set. These feature sets, which captured the basic information about each token were used to classify it. During tagging, the same feature extractor was used to convert unseen inputs to feature set. These feature sets were fed to the model and returned with tags. To make sure that the chosen tag is the right one based on probability, the cut-off probability was set to 0.2. This means, when Naïve Bayes encounter a probability of less than 20% for a tag, it will opt to choose the back-off tagger, which in this experiment is a default tagger.

Lastly, the exploration used the Brill tagger, which is an inductive method for part-of-speech tagging and can be summarized as an error-driven transformation-based tagger [6]. It is a form of supervised learning, which aims to minimize error; and a transformation-based process, in the sense that a tag is assigned to each word and changed using a set of predefined rules. It uses a series of rules to correct the result of the initial tagger. These rules were based on how many errors they correct minus the number of new errors they produce. Rules were reapplied repeatedly, until no more rules can be applied. To resolve this problem, the maximum number of rules that can be generated was set and controlled the measure on how well a rule corrects errors. Accuracy, did not change when the maximum rules were set from 25 to 200. Though they had the same accuracy, 100 was chosen as the final maximum rules because 25 was too small while 200 was too large.

III. RESULTS AND DISCUSSION

The experiments were done by running the different taggers and comparing the results produced by the taggers and the actual tags on the dataset. On the first run, the goal was to test which of the three *n*-gram taggers produced the most desirable result and it turned out that the unigram gave the highest accuracy of 89.03% but often wrongly tagged the words that had more than one tags since it only tagged based on frequency. Tri-gram had the lowest accuracy reaching to only 17.57% while bigram obtained an accuracy of 76.72% but both could not handle tokens that did not belong to the bigram pattern. Hence, the three were combined for the N-gram tagger, which used a default tagger to handle untagged tokens. Average accuracy for the N-gram is 90.29%.

On the second test, a TnT tagger used the three N-gram model, but this time, it only computed the probability of the tag given by the three models and chose the best tag according to likelihood. An experiment for the the number of beam search was made and had settled for 10 for this dataset. The tagger reached an accuracy of 91.00%.

For the next experiment, the dataset was fed to the Naïve Bayes tagger. The tagger was added by a back-off tagger on Naïve Bayes while setting the cut-off probability to 20%. However, it decreased the accuracy down to 90.78% so it was removed gaining back the accuracy of 90.98%.

For the last experiment, tagger used was Brill tagger which generated rules to correct tags. The maximum rules to 100, while minimum score was set to 3. The tagger reached an accuracy measure of 89.79%. Table 1 shows that N-gram, TnT, Naïve Bayes, and Brill taggers are capable of tagging Waray-waray text effectively achieving 90+% accuracy for training and test sets with TNT as the most promising tagger giving the highest accuracy.

Table 1. Accuracy measure of the POS tagging techniques used.

TAGGER	OPTIMUM ACCURACY
N-gram	90.29%
TnT	91.00%
Naïve Bayes	90.98%
Brill	89.79%

IV. CONCLUSION

The experiments on comparing the different POS taggers applied to Waray-waray language has been presented. The result showed that TnT tagger was the most promising tagger for the Waray-waray language than the other taggers like N-gram, Naïve Bayes, and Brill taggers. Although accuracies among the four taggers presented were relatively closer thus, there is a need to explore and tweak parameters for the taggers presented.

ACKNOWLEDGMENTS

We would like to thank Dr. Rachel Edita Roxas and Mr. Nathaniel Oco for sharing the PALITO corpus, which serves as the building block of this work.

REFERENCES

[1] Dita, S. R. (2009). Building Online Corpora of Philippine Languages. *Proceedings of the Twenty-third Pacific Asia Conference on Language, Information and Computation.* , 646-653.

[2] Santorini, B. (1990). *Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision).* Pennsylvania: University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-47.

[3] Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., & Chanona-Hernández, L. (2014). Syntactic n-Grams as Machine Learning Features for Natural Language Processing. *Expert Systems with Applications*, 853-860.

[4] Collins, M. (1996). A new statistical parser based on bigram lexical dependencies. *In Proceedings of the 34th Annual Meeting of the Association of Computational Linguistics*, (pp. 184-191). California.

[5] Martin, D. J. (2007). *Speech and Language Processing: An introduction to speech recognition, computational.*

[6] Brants, T. (2017, 08 12). *TnT -- Statistical Part-of-Speech Tagging.* Retrieved from coli: http://www.coli.uni-saarland.de/~thorsten/tnt/