Developing an Unsupervised Grammar Checker for Filipino Using Hybrid N-grams as Grammar Rules

Matthew Phillip Go
De la Salle University
2401 Taft Avenue,
Manila, Philippines
matthew_phillip_go@dlsu.edu.ph

Allan Borra
De la Salle University
2401 Taft Avenue,
Manila, Philippines
allan.borra@dlsu.edu.ph

Abstract

This study focuses on using hybrid n-grams as grammar rules for detecting grammatical errors and providing corrections in Filipino. These grammar rules are derived from grammatically-correct and tagged texts which are made up of part-of-speech (POS) tags, lemmas, and surface words sequences. Due to the structure of the rules used by this system, it presents an opportunity to have an unsupervised grammar checker for Filipino when coupled with existing POS taggers and morphological analyzers. The approach is also customized to cover different error types present in the Filipino language. The system achieved 82% accuracy when tested on checking erroneous and error-free texts.

1. Introduction

According to the philosopher and educator Kevin Browne, poor grammar implies two negative sentiments towards the writer: either he is not intelligent or he just does not care about his writing any better. Backing on this problem, there has been many researches and advances in the field of computer-aided grammar checking such as Microsoft Word, Google Docs, Grammarly, LanguageTool, and Ginger. These software solutions can detect syntactical errors such as spelling, punctuation, word forms, and word usages. However, most of these solutions have focused on the English language. There has been very few works in the Filipino language despite being a language of at least 100 million people[1]. Additionally, it is difficult to use an existing grammar checker system of one language and apply it on another since the system would have its specific design and functionalities tackling the unique phenomena of its target language.

The Filipino language, just like any other language, has its own unique phenomena which serve as a challenge in developing its own grammar checker system. It has a 'large vocabulary of root, borrowed, and derived words' caused by the arrival and/or colonization of foreign countries including: Spain, USA, and China in the Filipino land[2]. It also has a high degree of inflection and uses variety of affixes to change the part-of-speech of a root word (ex. root: *tira* 'live [on a house]', *tira* + *han* = *tirahan* 'house') or change the focus and aspect of a verb (*tirhan* 'live' – neutral aspect/object focus, *titira* 'will live' – contemplative aspect/ actor focus, *tumira* 'lived' – perfective aspect/ actor focus. Another linguistic phenomenon in Filipino is its free-word order structure. Filipino sentences, in its natural form, follow the predicate-subject sentence format (ex. *Masaya ako* – word-per-word is translated as 'Happy I') or as subject-predicate sentence format (ex. *Ako ay masaya* – word-per-word is translated as 'I [none] happy') where the word *ay* acts as a lexical marker and is usually placed after the subject and before the predicate. In the Filipino language, direct objects, adjectives and adverbs may also be written as phrases and including prepositional phrases, they also follow the free-word order and not being limited to just one position in the sentence (Ramos, 1971). For example, the sentence 'Mark ate an apple.' can be translated to: *Si Mark ay kumain ng mansanas.*, *Kumain si Mark ng mansanas.*, and *Kumain ng mansanas si Mark.* As seen in the last two translations, the direct object phrase *ng mansanas* 'apple' can be placed directly after the verb or after the subject yet both produce the exact same meaning.

[1] http://www.philstar.com/headlines/2016/01/03/1538653/philippines-population-seen-hit-104m

[2] http://ffemagazine.com/the-origin-of-the-filipino-language-wikang-filipino/

As of this writing, there are still no grammar-checking software systems for Filipino that is publicly available that cover broad-range of grammatical errors. This fact may be associated with the complex structure of the Filipino language which makes it difficult in constructing (error) grammar rules. Among the few existing grammar checkers in Filipino are: Panuring Pampanitikan (PanPam) by Jasa et al. (2007) and Language Tool for Filipino (LTF) by Oco & Borra (2011). PanPam is a syntax and semantics-based grammar checker for Filipino that makes use of error patterns as rules and lexical functional grammar as its parsing algorithm. LTF, on the other hand, uses a rule file containing error patterns in the form of regular expressions and part-of-speech tags and a dictionary file in detecting its errors and providing corresponding suggestions. Although these systems, especially LTF, could distinctly recognize grammatical errors from correct text by using error patterns, the main concern with these systems is that the parser rules, dictionaries, affix-to-root-word mappings, word-to-part-of-speech mappings, error patterns, and other files are manually defined which is a very tedious task to cover the entire language and all possible errors in it especially that the language is ever growing and the number of errors committed by writers are directly proportional to it. This concern is evident on the systems' presented limitations and results where only a small subset of errors was covered.

In other languages such as English, there are existing works such as Lexbar (Tsao & Wible, 2009), EdIt (Huang et al., 2011), Google books n-gram corpus as grammar checker (Nazar & Renau, 2012), and Chunk-based grammar checker for translated sentences (Lin et al., 2011) which are unsupervised grammar checker systems that make use of grammatically correct texts, their corresponding part-of-speech (POS) tags, and/or lemmas converted into n-gram sequences and used as grammar rules.

The Lexbar application (Tsao & Wible, 2009) generated hybrid n-grams, which are n-grams composed of words, POS tags, and lemmas. These hybrid n-grams are generated from actual tagged word sequences. For example, given phrases such as 'from her point of view' and 'from his point of view', the system will be able to generate the hybrid rule 'from *[dps]*[3] point of view'. This rule can be used to flag the phrase 'from my point of view' as grammatically correct and the phrase 'from him point of view' as incorrect. The Lexbar app was only tested on substitution-correctable errors. The EdIt system (Huang et al., 2011) also made use of hybrid n-grams (called *pattern rules*) as grammar rules but only generates the rules such as '*play ~ role* in *[Noun]*', '*play ~ role* in [*V-ing]*', and '*look forward* to [*V-ing]*[4] ' from specific lexical collocations such as '*play ~ role*' and '*look forward*'. These types of rules tackle much more specific error types in English. The key difference of EdIt with Lexbar is that it only limits the number of POS tokens in an n-gram rule to one while Lexbar can have one or more POS tokens such as the rule: 'from *[dps] [nn0]*[5]' derived from the phrases like 'from his house' and 'from her balcony'. EdIt applied its rules in detecting errors correctable by substitution, insertion, and deletion. Both Lexbar and EdIt used weighted Levenshtein edit distance algorithm in prioritizing its suggestions.

This research aims to build an unsupervised grammar checker system for Filipino using hybrid n-grams as grammar rules following a similar format as Lexbar's grammar rules. These rules will be used to detect grammatical errors in Filipino and provide suggestions such as substitution, insertion, deletion, merging, and unmerging extending the existing suggestions made by both Lexbar and EdIt.

2. Filipino Linguistic Phenomena

Aside from the free-word order structure in Filipino, there are other linguistic phenomena such as being morphologically rich, existence of compound words, and the rule in Filipino: "*Kung ano ang bigkas, siyang sulat"* 'Spell as you pronounce it' (Ortograpiyang Pambansa, 2013).

There are at least 50 affixes and other morphologies such as partial reduplication, full reduplication, and compounding that are used in Filipino. These morphologies are categorized into three: *inflectional* – changes in word form that 'accompany case, gender, number, tense, person, mood, or voice that have no effect in the word's part-of-speech'; *derivational* – changes in

[3] *dps* is the part-of-speech (POS) tag for possessive pronouns such as his, her, my, their, etc in the CLAWS5 tagset.

[4] *V-ing* is the POS tag for verbs followed by –ing in the CLAWS5 tagset.

[5] *nn0* is the POS tag for neutral nouns in the CLAWS5 tagset.

word form that changes the word's part-of-speech category; and *compounding* – 'where independent words are concatenated in some way to form a new word' (Bonus, 2003). See Table 1 for some of the different forms of the root word *kain* 'eat'.

Word	**Translation**
Verbs	
ikakain	will just eat
ikain	just eat
ipakain	feed
ipapakain	will feed
kainin	eat (something)
kinain	ate (something)
kinakain	eating (something)
kumain	(somebody) eating
Nouns	
hapagkainan	eating/dinner table
kainan	eating place
kakainan	eating place (where do-er will go later)
kinakainan	eating place (where do-er is right now)
pagkain	food
Adjective	
palakain	loves eating

Table 1: Different forms of *kain* 'eat'

There are also affixes that are separated by a hyphen (-) from its root word or morpheme (ex. *mang-akit* 'to entice' from the root *akit* 'entice'). There are also cases wherein addition or insertion of an affix to a word could alter the spelling of its base form (ex. The prefix *pang-* + *palit* 'change' = *pamalit* 'item for changing'). However, not all affixes and reduplication can be applied to any word. For instance, the root word *luto* 'cook' can use 'nag-' as prefix but *kain* 'eat' cannot. It should also be noted that there are assimilated words from English in Filipino wherein affixes are also appended to it (ex. *magce-cellphone* 'will use a cellphone', *i-file* 'to file (a document)'). The Filipino language also has its own set of compound words. There are two ways to combine words together, either with the use of a hyphen (ex. *halo-halo* '(a type of Filipino dessert)' from the word *halo* 'mix', and *kisap-mata* 'instant' from the words *kisap* 'blink' & *mata* 'eye') or just combining them as is (ex. *kapitbahay* 'neighbor' from the words *kapit* 'hold onto' & *bahay* 'house', and *hanapbuhay* 'livelihood' from the words *hanap* 'find' & *buhay* 'life') (Paz, 2003).

Another important linguistic phenomenon in Filipino is the rule: "*Kung ano ang bigkas, siyang sulat"* 'Spell as you pronounce it' (Ortograpiyang Pambansa, 2013). As the rule states, the words in Filipino are usually spelled as they are pronounced with some exceptions. This phenomenon simplifies the way Filipino words are spelled out (ex. Filipinized form of 'computer' as *kompyuter)* but also causes some spelling confusion which will be discussed in the next section.

3. Error Types

In understanding the error types that exist in Filipino writing, three references were used: The Cambridge Learner Corpus (Nicholls, 1999), Wikapedia (2015), and a parallel corpus of 1252 erroneous-and-correct word and phrase pairs from sentences written by Filipino university students.

The Cambridge Learner Corpus contains 16 million words from English examination scripts by learners of English containing different types of errors. The corpus categorized the error types into general and specific errors. The proponents noticed that some error categories would have its Filipino counterpart such as wrong form used, missing word/phrase, word/phrase needs replacing, unnecessary word/phrase, punctuation errors, countability errors, determiner agreement, incorrect verb inflection, spelling errors, and other error categories also exist in Filipino.

Wikapedia (2015) is a booklet created by the Presidential Communications Development and Strategic Planning Office of the Philippines containing correct usage of affixes, words, and phrases in Filipino which people may find confusing. One example described in the book would be the use of *ng*, a function word defining possession (ex. *aso ng kapitbahay* 'dog of neighbor') and in a direct object phrase (ex. *kumain ng mansanas* 'ate an apple') vs the use of *nang* which is commonly used before an adverb (ex. *kumain nang mabilis* 'ate fast'). The usage of these two words is confusing because it is pronounced almost exactly the same. Other examples contained in the booklet are proper usage of affixes and words, morphophonemics, usage of hyphens and spaces, and others.

After analyzing the parallel corpus of 1252 erroneous-correct word/phrase pairs, it is found that majority of the errors fall under spelling errors, incorrect usage of affixes/reduplication which is mostly caused by usage of hyphens and spaces, and wrong word usage.

It is observed that one reason the students made spelling errors is because of the way a word is pronounced which is usually simplified for conversational use. Some of these simplified words, see Table 2, are still not accepted in formal Filipino writing which cause spelling errors. Another cause of spelling errors is the confusion whether to spell an English borrowed word in its English version or convert it to its Filipinized spelling version.

There were many instances of affix errors where the students were confused whether a word is an affix of a word, a separate word, or if there should be a hyphen between the affix and the root word. A few of the affix errors also show the confusion of students in selecting an appropriate affix of a verb when used for a certain focus and/or aspect. See Table 3.

The students also committed several mistakes in identifying which word to use in certain situations which is caused by unfamiliarity with Filipino syntax rules. See Table 4.

Other errors that exist in the parallel corpus include the lack of space between words (ex. *pa rin* 'still' incorrectly written as *parin*), compound words that was separated by a space (ex. *araw-araw* 'everyday' incorrectly written as *araw araw*) and punctuation errors where some commas or periods are missing.

Correct Word	**Misspelled as**	**Reason**
noon 'before'	*nuon*	Pronounciation
mayroon 'have'	*meron*	Pronounciation
anong 'what'	*anung*	Pronounciation
iyong	*yung*	Pronounciation
tingnan 'look'	*tignan*	Pronounciation
kumpanya 'company'	*companya*	Filipinization
iskolarship 'scholarship'	*scholarship*	Filipinization
risertser 'researcher'	*researcher*	Filipinization

Table 2: Spelling Errors

Correct Word	**Misspelled as**	**Reason**
Pangkain 'used for eating'	Pang kain	Extra Space
Tagtuyo 'drought'	Tag-tuyo	Extra Hyphen
Ikawalo 'eighth'	Ika-walo	Extra Hyphen
i-predict 'to predict'	ipredict	Missing Hyphen
mas malaki 'bigger'	masmalaki	Missing Space
inilagay sa kahon 'placed in a box'	*nilagay* sa kahon	Incorrect Affix used for a verb focus

Table 3: Affix Errors

Confused between:	
ng 'of'	*nang* '(function word before an adverb)'
may 'has (used before nouns, verbs, adjectives and adverbs)'	*mayroon* 'has (used before grammatical particles, personal pronouns, and adverbs of place'
suffix *–ng* 'used in place of *na* if word preceding it ends in a vowel	*na* '(type of grammatical particle)'

Table 4: Wrong Word Usage

4. Overview of the Grammar Checker

The grammar checker named Gramatika that is discussed in this paper utilizes the existing implementation of the Lexbar application by Tsao & Wible (2009) and extends it to cover more error types, some of which are unique in the Filipino language. It uses n-grams as rules, commonly referred to as hybrid n-grams, from grammatically correct texts consisting of words, POS tags, and lemmas to detect grammatical errors and provide suggestions containing possible corrections. The production of POS tags, and lemmas can be produced by existing POS taggers and morphological analyzers [6] for Filipino making the system unsupervised such that new grammatically correct texts can be fed through these systems and to Gramatika to easily increase the number of grammar rules.

[6] See Rabo & Cheng (2006) and Bonus (2003)

4.1 Rules Learning

Even though Gramatika also uses hybrid n-grams similar to Lexbar's (Tsao & Wible, 2009) and slightly similar to EdIt's (Huang et al., 2011), the approach in deriving the hybrid n-grams is different. Gramatika uses a clustering approach as opposed to Lexbar's pruning and EdIt's collocations-based approaches. The n-gram sizes used as rules range from 2 to 7. For example, given an incorrect phrase *para sa bata ang laruan ni iyon.* '?that? toy is for the kid', if Gramatika has the hybrid 7-gram '*para_sa [NNC] [DTC] [NNC] na [PRO].'* [7], then it can immediately suggest to change the word *ni* '(a grammatical particle used before a personal proper noun)' to *na* '(a grammatical particle used around adjectives, pointing pronouns, and others)' which produces the corrected version: *para sa bata ang laruan na iyon* 'that toy is for the kid' which is a more appropriate suggestion than the suggestion produced by the trigram *[NNC] ni [NNP]* [8] to change *iyon* to a proper noun (ex. Mark) producing the corrected version: *para sa bata ang laruan ni Mark* 'Mark's toy is for the kid'. The use of larger n-gram sizes increases the context from which a suggestion can be based from.

In the clustering approach, all n-gram sequences are retrieved from grammatically correct texts and are stored in the database. During the storing process, the frequency of all POS tag sequences is counted. POS tag sequences exceeding the threshold of 2 are retrieved and the word n-grams are grouped as clusters. For each n-gram clusters, the module checks if there are any token slot that can be generalized to POS level. For example, if a cluster has the instances *nagpunta sa bayan* 'went to the town' and *bumisita sa bahay* 'visited the house', the first and third tokens can be *generalized* because it meets the minimum difference threshold of 2. This produces the hybrid n-gram *[VBTS] sa [NNC]* which can be used to flag the phrase *umupo sa silya* 'sat on the chair' as grammatically correct or used to detect grammatical errors. The n-gram rules are stored in the database as sequences of words, POS tags, lemmas, and a Boolean sequence denoting which token slots are *generalized*. This is done to allow Gramatika to provide word-specific suggestions and to also identify the appropriate transformed word to a specific POS -lemma mapping.

4.2 Error Detection

In detecting grammatical errors and producing suggestions based on the hybrid n-grams, a weighted Levenshtein edit distance algorithm is used. This algorithm is commonly used in spell checking to compute how many edits it will take to convert a potentially misspelled word to a correct word in the dictionary. It has also been used by EdIt (Huang et al., 2011) in providing corrections by substitution, insertion, and deletion. In Gramatika, the edit distance algorithm is extended to detect errors and provide suggestions correctable by substitution, insertion, deletion, spelling correction, unmerging, and merging. The error types that exists in Filipino are grouped based on the six suggestion types, see Table 5.

Correction	Error Types
Substitution	Affix/Form errors, wrong word/punctuation usage (includes preposition, determiners, and others)
Spelling Correction	Misspelled words, misuse/lack of hyphens
Insertion	Missing words and punctuations
Deletion	Unnecessary words and punctuations
Unmerging	Incorrectly merged words requiring unmerging of words or removal of hyphens
Merging	Incorrectly unmerged word requiring removal of space or insertion of hyphen between texts

Table 5: Correction and Error Types

In producing suggestions, Gramatika parses the input, which is POS and lemma-tagged, into n-grams starting from size 7 down to 2. For each input n-gram, it retrieves hybrid n-gram rules "similar" to the input n-gram from the database. A rule is considered "similar" to an input n-gram if at least *n–2* POS tokens of it are equal to the POS tokens in the input n-gram. Three sizes of the rules are also retrieved for each input n-gram: rules that are of equal size to the input n-gram to be used for substitution and spelling correction suggestions, rules that are one token size larger to produce insertion and unmerging suggestions, and rules that are one token size smaller to produce deletion and merging suggestions. If an

[7] Based from the Rabo & Cheng (2006) tag set, NNC = common noun, DTC = determiner for common nouns, PRO = pronoun pointing to an object

[8] NNP = proper noun

input n-gram is "equal" to a rule n-gram of the same size, then it is considered grammatically correct, which is denoted by an edit distance value of 0. "Equal" tokens, in this context, are defined as tokens having the same POS tag if the POS tag of the rule n-gram token is *generalized* or tokens that are equal in surface word level.

A substitution suggestion is outputted when all tokens in the rule n-gram except one are equal to its respective tokens in the input n-gram in the same index. The unequal token is categorized depending on its error type: affix errors or wrong word usage. In detecting affix errors, the system checks if the lemma of the unequal input token is equal to the respective rule token. If the lemmas are equal but the words and/or POS tags are not equal, then it is assumed that there is an affix usage error with respect to the rule n-gram used. For example, given the input *kumain siya bukas* 'he ate tomorrow' is compared against the rule n-gram *[VBTF] [PRS] bukas*[9], where one instance of the POS tag *[VBTF]* is the word *kakain* 'will eat', since *kumain* 'ate' and *kakain* has the same lemma *kain* 'eat', the error is classified as an affix usage error which will produce a suggestion 'Change *kumain* to *kakain*' to produce the phrase *kakain siya bukas* 'He will eat tomorrow'. For wrong word usage errors, Gramatika suggests that the unequal rule token should replace the unequal input token. Consider the example: *Bumili si bata ng laruan* '?the? child bought a toy'. It is considered grammatically incorrect because the word *si* is used as a determiner for proper nouns. Using the trigram rule *[VBTS] ang [NNC]* [10], the correct determiner for common nouns *ang* will be suggested to replace the incorrect word *si*. Corresponding edit distance values will be assigned to the outputted substitution suggestions as seen in Table 6.

Error Type	Weight	Error Type	Weight
Incorrect Affix/ Form	0.6	Wrong Word Same POS	0.8
Spelling Error	0.65	Wrong Word Diff. POS	0.95
Incorrectly Merged	0.7	Missing Word	1.0
Incorrectly Unmerged	0.7	Unnecessary Word	1.0

Table 6: Edit Distance Values

A spelling suggestion is similar to the substitution suggestion criteria where in all tokens except one should be equal to its respective tokens. The unequal tokens are compared using a character-level edit distance algorithm. If it meets the spelling difference threshold, then the token in the rule n-gram is outputted as a spelling correction suggestion. This approach is slightly similar to the traditional way of spell checking which is dictionary look up, but this uses context in providing an appropriate suggestion. For example, given the input *Bakt ?*, using dictionary look-up alone, two possible suggestions can be produced: *Bakat* 'a mark/trace' or *Bakit* 'why'. Using a context-based approach and the rule *[PRQ] ?*[11] where the word *bakit* is an instance of the POS tag *[PRQ]*, Gramatika will suggest the word *Bakit* to form the correct sentence *Bakit?* 'Why?'. This suggestion is assigned an edit distance value of 0.65, as seen in Table 6.

An insertion suggestion is outputted when all tokens in the input n-gram find their corresponding equal tokens in the rule n-gram and one token from the rule n-gram does not have a matched token. For example, given the input n-gram *a b d* and rule n-gram *a b c d*, tokens *a, b,* and *d* are equal tokens and *c* does not have a match which is outputted as an insertion suggestion. This suggestion is given an edit distance value of 1.0, as seen in Table 6.

A deletion suggestion is outputted when all tokens in the rule n-gram find their corresponding equal tokens in the input n-gram and one token from the input n-gram does not have a matched token. This one token will be suggested to be deleted. This suggestion is also given an edit distance value of 1.0, as seen in Table 6.

An unmerging suggestion is outputted when n-1 tokens in the input n-gram are equal to its respective tokens in the rule n-gram leaving one token in the input n-gram and two adjacent tokens in the rule n-gram without matching equal tokens. The system concatenates the two adjacent tokens in the rule n-gram together using a hyphen or removal of the space and checks if it equates to the spelling of the input token. If equal, then an unmerging suggestion is produced. For example, given the input four-gram *a b cd e*, and rule five-gram *a b c d e*, the tokens *a, b,* and *e* are equal tokens. After which, the tokens *c* and

[9] VBTF = contemplative verb, PRS = singular pronoun
[10] VBTS = perfective verb, NNC = common noun
[11] PRQ = question/interrogative pronoun

d in the rule n-gram is concatenated, since it equals the token *cd* in the input n-gram, then the unmerging suggestion is outputted. This suggestion is given an edit distance value of 0.7, as seen in Table 6.

A merging suggestion is outputted when n - 1 tokens in the rule n-gram are equal to its respective tokens in the input n-gram leaving one token in the rule n-gram and two adjacent tokens in the input n-gram without matching equal tokens. The system concatenates the two adjacent tokens in the input n-gram together using a hyphen or removal of the space and checks if it equates to the spelling of the rule token. If yes, then a merging suggestion is produced. For example, given the input five-gram *a b c d e*, and rule four-gram *a b cd e*, the tokens *a, b,* and *e* are equal tokens. After which, the tokens *c* and *d* in the input n-gram is concatenated, since it equals the token *cd* in the rule n-gram, then a merging suggestion is outputted. This suggestion is also given an edit distance value of 0.7, as seen in Table 6.

It should be noted that the edit distance values used are arbitrary and is mainly used for prioritization of suggestions only as the edit distance threshold is set to 1. In cases that there would be ties in terms of edit distance values (ex. three *wrong word different POS tag* suggestions), the frequency of how many times each suggestion is produced by the n-gram rules is also considered.

5. Results and Analysis

The Gramatika system is tested on 70 phrases (35 erroneous and 35 error-free) retrieved from PanPam (Jasa et al., 2007) Wikapedia (2015), translated documents by Filipino university students, and news articles. A small corpus of 2,668 complex sentences which consists of 70,312 tokens (14,575 unique tokens) is used in training of the n-gram rules which result to 83% accuracy, 93% precision, and 71% recall. Table 7 shows a summary of figures. 18 out of 25 erroneous phrases were marked as erroneous, and 23 out of 25 error-free phrases were marked as error-free.

Sentences	Correctly Flagged	Incorrectly Flagged	Total
Erroneous	25	10	35
Error-free	33	2	35
Total	58	12	70

Table 7: Grammar Checking Results

This result shows significant potential for an unsupervised grammar checker that currently only uses a small corpus of grammatically correct sentences.

In detecting errors, the system was able to produce word-specific suggestions for most errors except for one instance: *Noong 2006, mananalo ang* 'Last 2006, (something/someone) will win', having a verb aspect/tense and adverb of time disagreement, where the system only suggest to replace the word *mananalo* 'will win' with any perfective verb *[VBTS]* because it did not have the word- POS-lemma mapping *nanalo* 'won'-[VBTS] *panalo* 'win'. Other detected errors with produced word suggestions include affix errors (ex. *linaan* – root word *laan* 'allot' cannot use the infix '-in-' -> *naglaan* 'alloted'), wrong word usages (ex. *nang* vs *ng*, *para_sa* vs *para_kay*), spelling errors (ex. *kikumpara -> kinukumpara* 'comparing', *lalake -> lalaki* 'man', *nag-simula -> nagsimula* 'started', *nagka-loob -> nagkaloob* 'given', *skolar -> iskolar* 'scholar', *pwesto -> puwesto* 'position'), merging/unmerging errors (ex *maka kuha -> makakuha* 'to get', *parin -> pa rin* 'still'), and missing punctuation (ex. *Mayo 31 2016 -> Mayo 31, 2016* 'May 31, 2016').

Gramatika failed to produce correct suggestions for some erroneous inputs because of several reasons. One reason is that some tags in the Rabo & Cheng (2006) tag set are *too* generalized. For instance, the sequence *para sa Mark* 'for the Mark' was incorrectly flagged as grammatically correct which is supposedly *para kay Mark* 'for Mark' because *para sa* is used for common nouns and proper nouns of place. This occurred because all proper nouns only use the tag [NNP] and an n-gram rule *para_sa [NNP]* was produced from phrases with proper nouns of place such as *para sa Amerika,* and *para sa Taiwan*. Another reason is that some words or phrase sequences were not part of the training corpus. For example, the phrase *taga Manila siya* was not corrected to *taga-Manila siya* 'He's from Manila' because the word *taga-Manila* was not in the database, thus preventing a merging suggestion. The system also failed to detect the grammatical error in the phrase *Maganda si.* 'is beautiful.' which can be corrected as *Maganda si Maria.* 'Maria is beautiful.' because the corpus sentences were all complex sentences and thus the correction for the simple sentence was not produced. The system also incorrectly flagged the phrase *na bibigay ang* where *bibigay* is an invalid contemplative form *[VBTF]* of the word

bigay 'give' which is supposedly *magbibigay* 'will give' because there is no *magbibigay-[VBTF]-bigay* mapping in the database. This is caused by the word *bibigay* being given the tag *[VBTF]* since it follows the partial reduplication rule for the said tag and the hybrid sequence *na [VBTF] ang* being considered grammatically-correct based from sequences such as *na gagampanan ang* 'will take the role of'.

The system performed well in correctly flagging error-free phrases even though many of the words were not in the training corpus. This result can be attributed to the POS tags of the input phrases and the *generalized* rules. For instance, the phrase *ang pagtakbo ng mayor* 'mayor's candidacy' was flagged correctly because of the n-gram rule *ang [NNC] ng [NNC]* that came from instances such as *ang kulay ng apoy* 'fire's color', *ang proseso ng produksyon* 'production's process'. However, there are two instances wherein the system incorrectly flagged error-free phrases: *Siya ay masaya.* 'She is happy' because simple sentences that contain the word *ay* did not occur in the training corpus and *Kinuha ko ito dito.* 'I got it from here' because neither the word sequence *ko ito* nor the POS sequence *[PRS] [PRO]* is not in the n-gram rules.

6. Summary & Future Works

This paper presents an unsupervised grammar checker system for Filipino that uses grammatically correct texts as grammar rules in the form of hybrid n-grams. It achieved 83% accuracy, 93% precision, and 71% recall in detecting broad-range of grammar errors in Filipino using only a small corpus of 2,668 sentences which can potentially be further improved as the size and variety of training sentences increases.

Future works for this system includes exploring other tag sets or modifying existing ones to create more specific tag groups for the system to avoid errors caused by tag groups that are *too* general. Corpus size may also be increased to produce more n-gram rules and word- POS-lemma mappings used for grammar checking.

Acknowledgment

This research work is supported by the Department of Science & Technology – Philippines and De la Salle University – Manila as part of its Interdisciplinary Signal Processing for Pinoys: Software Applications for Education (ISIP:SAFE) research program.

References

Chung-Chi Huang, Mei-Hua Chen, Shih-Ting Huang, Jason Chang (2011). EdIt: A Broad-coverage Grammar Checker Using Pattern grammar. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (p. 26-31).

Consuelo J. Paz. (2003). Compounding Old and New Words in Filipino. University of the Philippines.

Diane Nicholls. 1999. The Cambridge Learner Corpus – error coding and analysis for lexicography and ELT. Cambridge University Press.

Don Erick Bonus. 2003. The Tagalog Stemming Algorithm (TagSA). Master's Thesis. De la Salle University, Manila.

Michael Jasa, Justin O. Palisoc, and Martee M. Villa. 2007. Panuring Pampanitikan (PanPam): A Sentence Syntax and Semantic Based Grammar Checker for Filipino. Undergraduate Thesis. De La Salle University, Manila.

Nathaniel Oco and Allan Borra. 2011. A Grammar Checker for Tagalog using LanguageTool. Proceedings of the 9th Workshop on Asian Language Resources Collocated with IJCNLP 2011.

Nai-Lung Tsao & David Wible. 2009. A Method for Unsupervised Broad-Coverage Lexical Error Detection and Correction. Proceedings of the NAACL HLT Workshop on Innovative Use of NLP for Building Educational Applications, p. 51-54.

Nay Yee Lin, Khin Mar Soe, and Ni Lar Thein. 2011. Developing a chunk-based grammar checker for translated English sentences. In Proceedings of PACLIC-2011 (p. 245-254).

Ortograpiyang Pambansa. 2013. Komisyon sa Wikang Pilipino.

Rogelio Nazar & Irene Renau. 2012. Google Books N-gram Corpus used as a Grammar Checker. Proceedings of the EACL 2012 Workshop on Computational Linguistics and Writing, p. 27–34.

Teresita V. Ramos. 1971. Makabagong Bararila ng Pilipino. Rex Book Store.

Vladimir Rabo and Charibeth K. Cheng. (2006). TPOST: A Template-based Part-of-Speech Tagger for Tagalog. Journal of Research in Science, Computing, and Engineering, 3(1).

Wikapedia (2015), Manila: Lexmedia Digital Corporation.