Balarila: Deep Learning for Semantic Grammar Error Correction in Low-Resource Settings

Paolo Espiritu, Joshue Jadie, Andre Ponce, and Charibeth Cheng
College of Computer Studies
De La Salle University, Manila
{paolo_edni_v_espiritu, joshue_jadie, andre_ponce, charibeth.cheng}@dlsu.edu.ph

Abstract

While there are many grammar checkers available for various languages, especially the English language, those that exist for the low-resource Filipino language can only effectively correct lexical errors. There is yet to be a publicly available Filipino grammar checker that can also address semantic errors, which are more complex. As such, this study found an opportunity to introduce *Balarila*, a deep learning-based Filipino GEC model inspired by the GECToR approach. To address the absence of a training and test dataset, an automated error generation pipeline was devised, creating synthetic datasets of error-free and error-filled Filipino sentences sourced from various online news sources. Tagalog BERT and RoBERTa models were fine-tuned in two stages using this generated corpus. Evaluation metrics included precision, recall, and $F_{0.5}$ scores for GEC, and a multi-class confusion matrix for GED. The top-performing model, RoBERTa Tagalog Large, achieved an $F_{0.5}$ score of 70.75, while the RoBERTa Tagalog Base, with a $F_{0.5}$ score of 69.00, demonstrated cost-effectiveness in training. The created datasets can also be used as a benchmark for Filipino grammar checker models.

1 Introduction

Writing sentences in a certain language requires skills that are only developed through a lot of practice. A sufficient understanding of the rules and syntax of a language is necessary to avoid breakdowns in communication. The Filipino language is not exempted from such a necessity.

Grammar checkers such as Grammarly are said to be beneficial in aiding individuals hone the different aspects of their writing skills such as sentence construction, vocabulary usage, proper grammar, and language mechanics (Ghufron and Rosyida, 2018; Jayavalan and Razali, 2018). Using such tools is one of the many ways one can develop better writing skills.

There already exists a Filipino grammar checker called *Gramatika* – which utilizes rule-based methods combined with statistical machine translation (SMT) (Go et al., 2017). However, it is limited by the availability of expert-annotated corpora and by other limitations that come with SMT-based implementations (Solyman et al., 2021; Chollampatt and Ng, 2018). Furthermore, as Gramatika was released in 2017, there is yet to be a Filipino grammar checker that adopts state-of-the-art approaches such as transformer-based models.

In a recent study, a sequence tagging approach was introduced to simplify the task of sequence generation. The proposed model only utilized a transformer encoder and some basic linear layers. The results showed that the inference speed of the model improved 10 times compared to transformer-based seq2seq systems (Omelianchuk et al., 2020).

However, these models require large amounts of data for training. This poses a problem for low-resource languages such as Filipino. Workarounds were created to address this such as synthetic dataset creation (Grundkiewicz et al., 2019) and large-scale corpus creation (Cruz and Cheng, 2021).

In this paper, an opportunity has been found to develop a transformer encoder-based model that will effectively detect and correct grammatical errors in the Filipino language. Furthermore, a demonstration of how a synthetic error-free and error-filled Filipino text dataset can be used as a benchmark for grammar error detection and correction is also provided.

2 Related Literature

GECToR (Omelianchuk et al., 2020) is a transformer encoder-based modelthat approaches GEC as an iterative sequence tagging task instead of sequence generation. The approach essentially reduces the task into a language-understanding problem, which only needs a transformer encoder

Proceedings of the First Workshop in South East Asian Language Processing, pages 21–29

stacked with linear layers. Given a target sentence, the sequence tagger model predicts the tag-encoded transformations for each token. The predicted tags are then applied to the target sentence through post-processing to get the modified sentence. Since some corrections in a sentence may depend on previous corrections, the same process is executed to the modified sentence to correct it further. This is repeated until the sentence is fully corrected.

The model predicts two types of token-level transformations: basic transformations and *g*-transformations (Omelianchuk et al., 2020). Basic transformations are common token-level edit operations such as keeping the token unchanged, deleting the token, and replacing the token with a new token. Meanwhile, *g*-transformations are custom-designed transformations that perform task-specific operations such as changing the case of the current token and splitting the token into two new tokens.

The model training is performed through three stages: (1) pre-train the model on synthetic data, (2) fine-tune the model on a corpus full of grammatically incorrect sentences, and (3) fine-tune the model on a corpus of mixed grammatically correct and incorrect sentences. With pre-trained transformer encoders such as BERT (Devlin et al., 2018), only training stages 2 and 3 are used. Model optimization was done using the Adam optimizer (Kingma and Ba, 2015) with default hyperparameters (Omelianchuk et al., 2020).

3 Methodology

3.1 Covered Grammatical Errors

Two categories of grammatical errors were covered by this study: (1) grammar errors and (2) spelling errors. These were derived from a book and previous studies that tackle Filipino grammatical errors (Komisyon sa Wikang Filipino, 2013; Go and Borra, 2016; Octaviano et al., 2016).

3.1.1 Grammar Errors

- *Morphological Errors.* These are grammatical errors that may be caused by words that went through morphological changes (Octaviano et al., 2016). Due to various morphological processes, certain Filipino words alter their spelling. For example, when the prefix /pang-/ is attached to the word *bili* 'buy', it yields the word *pambili* 'a resource used to by something' since the original word begins with /p/, /b/, or /m/.

- *Wrong use of nang vs. ng.* These are grammatical errors that may be caused by the improper use of the words *nang* and *ng*. These words are commonly interchanged mainly due to their similar pronunciations.

3.1.2 Spelling Errors

- *Duplicate Words.* These are spelling errors caused by mistakenly repeating a word that should not be repeated. Words such as *ang* 'the', *ng* 'of', and *mga* (denotes the plural form of a word) are some of the commonly duplicated words (Octaviano et al., 2016).

- *Missing Spaces.* These are spelling errors caused by improper merging of Filipino words due to a missing space. For example, *pa rin* meaning 'still' is commonly written as *parin* similar to how 'going to' is sometimes written as 'gonna'.

- *Extra Spaces.* These are spelling errors caused by not properly merging Filipino words together. For example, *pinakamalaki* 'biggest' is erroneously spelled as *pinaka malaki*.

- *Wrong Use of Hyphens.* These are spelling errors caused by the wrong use of hyphens - which may be used to separate prefixes from the base word. Hyphens are used when the base word is a proper noun, loan word, or starts with a vowel (Octaviano et al., 2016). For example, *nag-usap* is erroneously written as *nagusap*.

- *Wrong Use of Enclitics.* These are spelling errors caused by confusion on the rules of enclitics starting with /d/ and /r/. For example, the words *din* and *rin* both mean 'also'. However, *rin* is used when the previous words ends with a vowel or semi-vowel, otherwise *din* is used.

3.2 Balarila

The model adopted the approach of the GECToR model (Omelianchuk et al., 2020). It performs the GEC task through iterative sequence tagging instead of sequence generation. The approach consisted of two (2) phases: fine-tuning and inference.

3.2.1 Fine-tuning

As shown in Figure 1, a pre-trained transformer encoder such as BERT was fine-tuned for the GEC

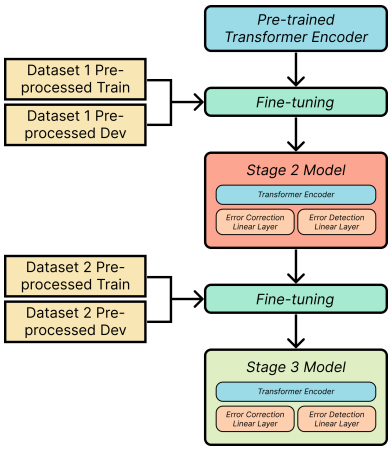


Figure 1: Fine-tuning Pipeline

task. In the second training stage, two (2) linear layers were added on top of the model first which are the error detection and error correction linear layers. This was done to produce the task outputs. Afterwards, the model was fine-tuned with error-filled Filipino sentences (Dataset 1). In the third training stage, the model was fine-tuned with both error-filled and error-free Filipino sentences (Dataset 2). These training stages were performed in sequence as crucial to the model's performance (Omelianchuk et al., 2020). Fine-tuning the model first on error-filled sentences allowed the model to effectively learn the different types of Filipino errors covered, the contexts of the errors, and the patterns of correct Filipino grammar. For model optimization, the Adam optimizer (Kingma and Ba, 2015) with default hyperparameters was utilized.

3.2.2 Inference

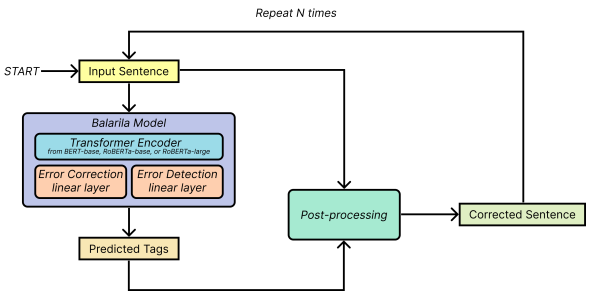


Figure 2: Inference Sequence Tagging Pipeline

Figure 2 shows Balarila's inference pipeline as adopted from GECToR (Omelianchuk et al., 2020). To correct the grammatical errors in an input sentence, the input sentence is first passed into the fine-tuned transformer encoder model to be converted into a context vector. The context vector is then passed to the error detection and correction linear layers, with Softmax layers on the top to predict the transformation tags per token. Based on the predicted tags, transformations are applied to each token through post-processing afterwards to produce the output sentence.

As the technique is iterative, the output sentence may not be fully corrected yet in the first iteration. The process is then repeated to the output sentence to correct it further. Table 1 shows an example of how the iterative sequence tagging technique is used to correct an input sentence.

# of Iterations	Sentence
0	*Punta niya sa mall kahapon*
1	*Punta niya sa mall kahapon.*
2	*Punta siya sa mall kahapon.*
3	*Pumunta siya sa mall kahapon.*

Table 1: Iterative sequence tagging example

In three (3) iterations, the sentence was fully corrected by adding a period in the end, replacing *niya* with *siya*, and changing *punta* to *pumunta*. This implied that the higher the number of iterations, the better the result will be. However, it is to be taken into account that the higher the number of iterations, the lower the inference speed of the model. The number of iterations should be configured properly to produce an accurate and fast model (Omelianchuk et al., 2020). For Balarila, the default number of iterations set by GECToR was used, which is five (5). The model corrects one error per iteration. With this approach, errors that can only be corrected based on previous corrections are corrected on the succeeding iterations.

3.2.3 Transformation Tags

Table 2 shows the transformation tags used by the model to correct each covered grammatical error. Most of the tags were adopted from the GECToR (Omelianchuk et al., 2020) model, and a new set of tags was also introduced. These new tags were used for other errors that were not covered by the default set of tags such as Filipino morphological errors and punctuation errors.

The tags used in morphological errors represent Tagalog verb form transformations. Tagalog verb forms are created using four (4) aspects which are completed, incompleted, contemplated, and recently completed. The focus of the verb is also included thus the verb and its aspect can either be in actor-focus or object-focus form. See Table 3 for some examples under each.

Error Types	Tags
Wrong Use of *nang vs. ng*	$REPLACE_*nang*, $REPLACE_*ng*
Wrong Use of Enclitics	$REPLACE_*daw*, $REPLACE_*din*, $REPLACE_*dito*, $REPLACE_*diyan*, $REPLACE_*doon*, $REPLACE_*raw*, $REPLACE_*rin*, $REPLACE_*roon*, $REPLACE_*rito*, $REPLACE_*riyan*
Wrong Use of Hyphens	$MERGE_HYPHEN, $TRANSFORM_INSERT_HYPHEN, $TRANSFORM_SPLIT_HYPHEN
Wrong Use of Spaces	$MERGE_SPACE, $TRANSFORM_SPLIT_SPACE
Duplicate Words	$DELETE
Morphological Errors	$TRANSFORM_VERB_BASE, $TRANSFORM_VERB_COMPACT, $TRANSFORM_VERB_COMPOBJ, $TRANSFORM_VERB_CONTACT, $TRANSFORM_VERB_CONTOBJ, $TRANSFORM_VERB_INCACT, $TRANSFORM_VERB_INCOBJ, $TRANSFORM_VERB_RECCOMP
Wrong Use of Punctuation Marks	$ADD_PUNC_EMARK, $ADD_PUNC_PERIOD, $ADD_PUNC_QMARK, $CHANGE_PUNC_EMARK, $CHANGE_PUNC_PERIOD, $CHANGE_PUNC_QMARK
Improper Word Casing	$TRANSFORM_CASE_CAPITAL, $TRANSFORM_CASE_LOWER
Missing Words	$APPEND_$t_1$
Wrong Use of Ang and Ng Pronouns	$REPLACE_*nila*, $REPLACE_*niya* $REPLACE_*sila*, $REPLACE_*siya*

Table 2: Error Types and Transformation Tags

Tag	Verb Form	Examples
BASE	Base	luto, sagot, sukat
COMPACT	Completed Aspect + Actor Focus	nagluto, sumagot, nagsukat
INCACT	Incompleted Aspect + Actor Focus	nagluluto, sumasagot, nagsusukat
CONTACT	Contemplated Aspect + Actor Focus	magluluto, sasagot, magsusukat
IMPACT	Imperative Aspect + Actor Focus	magluto, magsagot, magsukat
COMPOBJ	Completed Aspect + Actor Focus	niluto, sinagot, sinukat
INCOBJ	Incompleted Aspect + Object Focus	niluluto, sinasagot, sinusukat
CONTOBJ	Contemplated Aspect + Object Focus	lulutuin, sasagutin, susukatin
IMPOBJ	Imperative Aspect + Object Focus	lutuin, sagutin, sukatin
RECCOMP	Recently Completed Aspect	kaluluto, kasasagot, kasusukat

Table 3: Tagalog Verb Form Transformation Tags

3.3 Data Collection

The datasets used for this study are composed of sentences scraped from various publicly available mainstream Filipino news websites - specifically *Abante*, *Bandera*, *GMA*, and *Pilipino Star Ngayon*. In total, 58,464 news articles were scraped, and these articles contained a total of 510,411 raw sentences. It is worth noting that some of these articles are written in *Taglish* - which is a combination of English and Tagalog words. Furthermore, in order to prevent re-introducing the Filipino RoBERTa models (Cruz and Cheng, 2021) to the same news articles they were trained on, only news articles dated January 2021 and onward were included in scraping. The cutoff for the scraped data was May 2023, since this was when the scraping was last performed before the models were finalized.

3.4 Data Cleaning

Some cleaning were performed on the collected raw sentences:

- *Incorrect Enclitics.* Some sentence entries incorrectly use entries (i.e., raw vs. daw, roon vs. doon) prior to scraping. These were corrected using a rule-based enclitic-correcting algorithm.

- *Multiple Sentences.* Some sentence entries contained multiple sentences within themselves separated by periods. These sentences were further split based on the period character's location.

- *Invalid Characters.* Some sentence entries contained invalid characters such as emojis. As such, these sentences were dropped.

- *English-dominant Sentences.* Some sentence entries were dominated by English words. With this, sentences that are composed of at least 50% Filipino words were retained, while the rest were dropped. The 50% threshold was set because using Filipino sometimes still requires borrowed English words.

3.5 Error Automation

In corrupting the cleaned sentences, a variety of approaches were used depending on the grammatical error to be reproduced. It is also worth noting that this pipeline was designed with the assumption that only one error was to be introduced per corrupted sentence.

- *Word Replacement.* Replace a target word with another. Applicable to morphological errors, wrong use of *nang* vs. *ng*, and wrong use of enclitics.

- *Character Replacement.* Replace a single character with another. Applicable to wrong use of hyphens, wrong use of punctuation marks, and improper word casing.

- *Punctuation Removal.* Remove the punctuation mark at the end of the sentence. Applicable to the wrong use of punctuation marks.

- *Word Duplication.* Duplicate a target word within the sentence. Applicable to duplicate word errors.
- *Word Deletion.* Delete a target word within the sentence. Applicable to missing word errors.
- *Word Stitching.* Stitch two words together by removing the space or hyphen in-between them. Applicable to missing spaces and wrong use of hyphens errors.
- *Word Splitting.* Insert a single space in-between a pair of syllables in a target word. Applicable to extra spaces errors.
- *Multiple Sentences.* Given a multi-sentence input, a single sentence is randomly chosen and is corrupted using either the character replacement or punctuation removal approach.

3.6 Datasets

A total of two (2) datasets were needed with respect to the second and third fine-tuning stages in the GECToR model's approach (Omelianchuk et al., 2020). A dataset creation algorithm was used to create the needed datasets using the corrupted sentences produced by the error automation algorithm.

The first dataset is intended for fine-tuning on error-filled sentences alone. The second dataset is intended for fine-tuning on both error-free and error-filled sentences. Each sentence in these datasets had its corrected counterpart as its label – even the error-free ones.

For each transformation tag, it was ensured that each tag was equally represented within the datasets. A 30,000 upper limit on the number of sentences under each transformation tag. However, there were certain tags that contained less than 30,000 sentences, since there was a lack of sentences needed in corrupting for those tags. This was a limitation with the data scraped. An example of this is the $REPLACE_*riyan* tag – which only had a total of 472 corrupted sentences. This is because the word *riyan* was not being used that much in the scraped news articles.

Furthermore, a 17:83 ratio was maintained between error-free and error-filled sentences for each transformation tag. This implies that, if applicable, a given transformation tag contains approximately 5,000 error-free and 25,000 error-filled sentences.

Dataset	Error-free Sentences	Error-filled Sentences	TOTAL
1	0	601,256	601,256
2	155,419	150,283	305,702
TOTAL	155,419	751,539	**906,958**

Table 4: Balarila dataset error-free and error-filled sentences composition

As for the two datasets, a 0:100 error-free and 80:20 error-filled ratio was established between the two. This results in an approximately equal number of error-free and error-filled sentences in Dataset 2. The actual number of sentences per dataset can be seen in Table 4. The whole Balarila dataset contains a total of 906,958 sentences.

The datasets were then split into three (3) subsets each: train, dev, and test with a 70:15:15 ratio of sentence distribution. As for the test sets, its sentences were further grouped according to: (1) its transformation tag and (2) whether it is grammatically correct or not. This is for the easier evaluation of the models under each tag.

Finally, for the distribution of errors, aside from the imposition of a 30,000-sentence cap, the dataset creation algorithm also goes through the sentences under each transformation tag and distributes the error-free and error-filled sentences onto the train, dev, and test subsets with respect to the aforementioned 0:100, 80:20, and 70:15:15 ratios. Since this splitting was performed for each transformation tag, this also ensures that each tag was split in the same manner. Thus, this also ensures better equality and representation for each tag.

3.7 Data Pre-processing

Before feeding the datasets into the models, each source-target pair of sentences were pre-processed first with respect to the pre-processing algorithm used in the GECToR (Omelianchuk et al., 2020) model. This is done to efficiently determine and attach the transformation tags needed to convert the source tokens into their corresponding target tokens.

3.8 Experiment Setup

To achieve the best-performing model, an experimental setup involving the three (3) transformer encoder models was prepared. Specifically, the BERT Tagalog Base (BERT-Base) (Cruz and Cheng, 2019), RoBERTa Tagalog Base (RoBERTa-Base) (Cruz and Cheng, 2021), and RoBERTa Tagalog Large (RoBERTa-Large) (Cruz and Cheng, 2021) models were fine-tuned and tested using the dataset

discussed in Subsection 3.6. Furthermore, all of the models were fine-tuned and tested on an NVIDIA RTX A6000 GPU using the GECToR model's default fine-tuning and predicting hyperparameters.

3.9 Evaluation

To evaluate the GEC performance of the models, the precision, recall, and $F_{0.5}$ score metrics from the CoNLL-2014 Shared Task (Ng et al., 2014) were adopted.

For the models' GED performance, a multi-class confusion matrix was used and visualized with a heatmap. With the 39 pre-defined unique tags enumerated in Table 2, the task of the model was to classify the erroneous token in a sentence to its corresponding transformation tag. The lighter the color of the tile in the confusion matrix, the better the performance of the model in detecting errors.

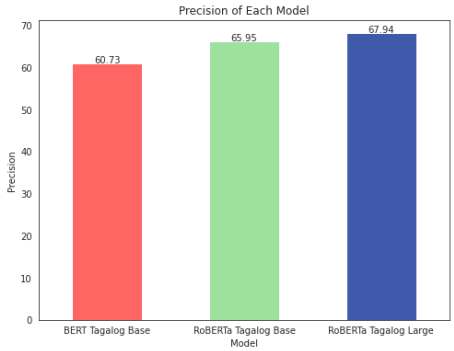
The GED performance was tested on error-free and error-filled datasets separately. To determine the accuracy of the model on error-free sentences, the prediction should correspond with *'NO CHANGES'* since it is expected that the model should not make any corrections on these sentences. When the models are tested on the incorrect sentences, the accuracy is ascertained when the predicted tag matches the target tag.

4 Results & Analysis

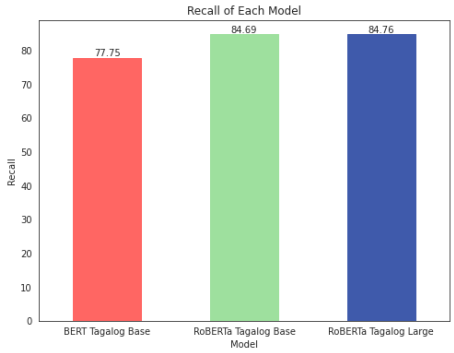
The GEC performance results of the three (3) models are shown in Figure 3. In terms of precision, RoBERTa-Large obtained the highest score of 67.94 as illustrated in Figure 3a. On the other hand, Figure 3b shows the performances of the models in regard to recall where RoBERTa-Base and RoBERTa-Large achieved similar scores of 84.69 and 84.76 respectively. For the $F_{0.5}$ scores as displayed in Figure 3c, RoBERTa-Large also obtained the highest score of 70.75 in comparison to RoBERTa-Base's score of 69.00. Moreover, BERT-Base had the poorest performance in every GEC score among the three (3) models.

As for the GED performance of the three (3) models, Table 6 shows the summary of the results from the confusion matrices. These are the grouped average scores of the transformation tags per error type on both error-free and error-filled datasets.

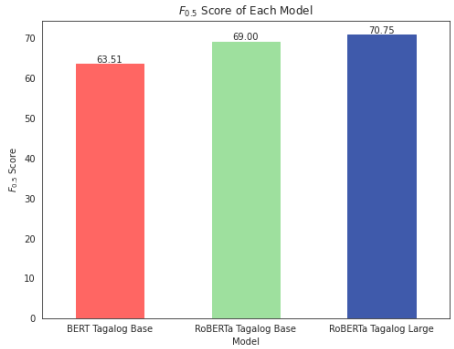
As observed in Table 6a, all three (3) models faced difficulties in identifying errors associated with duplicate words, morphological errors, wrong use of punctuation marks, and missing words.



(a) Precision



(b) Recall



(c) $F_{0.5}$ Score

Figure 3: GEC Performance of the Three Models

The difficulty in detecting duplicate words may be attributed to the error automation algorithm used which is Word Duplication. The problem that may arise from this algorithm is the vagueness of the duplicated word in a sentence since there are no restrictions on when to duplicate a word. This may have resulted in a poor performance towards correcting errors related to Duplicate Words since the models would find it difficult to determine when to delete a word in a sentence given all the unique Filipino words it was introduced to.

For the morphological errors, its error automation randomly replaces a verb by either changing its aspect, focus, or both. With six (6) possible morphological transformations for a single Tagalog

Covered Errors	BERT-Base	RoBERTa-Base	RoBERTa-Large
Wrong Use of *nang* vs. *ng*	96.21%	96.86%	96.73%
Wrong Use of Enclitics	97.48%	87.43%	84.85%
Wrong Use of Hyphens	98.39%	98.61%	98.82%
Wrong Use of Spaces	98.82%	99.28%	99.35%
Duplicate Words	99.48%	98.95%	98.82%
Morphological Errors	91.59%	92.87%	91.61%
Additional Errors			
Wrong Use of Punctuation Marks	95.77%	94.78%	93.58%
Improper Word Casing	98.45%	98.32%	98.49%
Missing Words	98.04%	96.47%	96.99%
Wrong Use of *Ang* and *Ng* Pronouns	89.89%	96.91%	97.41%

(a) Error-Free Dataset

Covered Errors	BERT-Base	RoBERTa-Base	RoBERTa-Large
Wrong Use of *nang* vs. *ng*	94.31%	95.72%	96.45%
Wrong Use of Enclitics	97.24%	98.06%	95.66%
Wrong Use of Hyphens	89.83%	94.65%	93.84%
Wrong Use of Spaces	82.46%	88.42%	88.55%
Duplicate Words	61.31%	80.99%	79.12%
Morphological Errors	56.43%	71.87%	73.13%
Additional Errors			
Wrong Use of Punctuation Marks	67.51%	79.98%	80.76%
Improper Word Casing	93.52%	96.35%	96.31%
Missing Words	27.18%	48.73%	49.40%
Wrong Use of *Ang* and *Ng* Pronouns	95.53%	97.96%	97.78%

(a) Error-Filled Dataset

Table 6: Grouped GED Performance Results of the Three Models

verb, there are occurrences wherein the corrupted sentence may still be grammatically sound which can confuse the models. An example of this would be *'Tira nila ng bola kahapon.'* wherein it was expected that the words *Tira* and *ng* were replaced with *Tinira* and *ang* respectively. Instead, the models replaced *Tira* with *Tumira* and *nila* with *sila*, resulting in *'Tumira sila ng bola kahapon.'* which is still grammatically correct. This may have been the cause regarding the models' performances in this error type for both error-free and error-filled sentences.

Moreover, the models struggled in identifying the wrong use of punctuation marks specifically for EMARK transformation tags as there are no common indicators when a Filipino sentence should be ended with an exclamation mark. Although, the models performed well for the PERIOD and QMARK tags since the PERIOD tags are often used to end a sentence, and the QMARK tags are hinted by many words such as *sino*, *saan*, *ano*, *bakit*, and *ba*, which EMARK does not have. An example of this would be *'Ang galing naman niya.'* wherein it was expected for the models to replace the period with an exclamation mark. However, the models did not perform the correction and treated the sentence as non-erroneous, which is still correct.

For the missing words error, the error automation algorithm used for this error type is Word Deletion which had similar issues with Word Duplication. With all the unique Filipino words it learned to add to the sentence, the models also struggled with adding the appropriate Filipino word given the context of the sentence. This also caused the models to obtain a lower score in this error type.

It was also discovered that some scraped sentences were already erroneous before even going through the error automation pipeline. An example of this are sentences that incorrectly used enclitics - which were fixed in the data cleaning pipeline. However, for the other unaddressed errors, it went through the error automation pipeline with the assumption that it was grammatically correct. As such, there may be some corrupted sentences within the dataset that contain more than one (1) errors. This could have potentially affected the results of the study in a variety of ways such as:

- **A model being able to correct the unexpected error.** Though this may be more practical in real-world uses, this would result in lower performance scores for the model in the context of this study. This is because the gold standard edits assumed that the only errors that the corrupted sentences had were the ones introduced by the error automation pipeline.

- **A model being mistakenly taught that the erroneous sentence is correct.** This could potentially lead to more confusion for the models - mainly due to the inconsistencies with how correct and incorrect sentences are being presented to it. Though this would result in better performance scores in the context of this study, this is not practical when it comes to real-world usage.

There was also an inconsistency towards correcting the same erroneous sentence when more errors were added. An example of this is when given the erroneous sentence *'Ang ado bo ay kinain niya kahapon.'*, wherein the words *ado* and *bo* must be merged together, this will not be corrected by the models anymore when the period ('.') is removed from the erroneous sentence. The inconsistency might have been caused by the dataset's lack of instances to represent the same error in a different setting. In the case of the example sentence, the dataset did not have enough instances to represent the error in sentences without a period at the end. Solving this issue is challenging with the current error automation since only one error was introduced per corrupted sentence.

Overall, RoBERTa-Large was the best model for the Filipino GEC task. Even though BERT-Base and RoBERTa-Base outperformed RoBERTa-Large in some error types as observed in the GED results, RoBERTa-Large generally performed better than them in terms of the GEC scores. This means that when it comes to producing corrections, RoBERTa-Large's outputs were the closest to the gold-standard sentences in the dataset compared to BERT-Base and RoBERTa-Base's outputs.

However, RoBERTa-Large often encountered memory issues during training which caused some modifications to the model's training parameters. The parameter adjustments were necessary due to the lack of available GPU Memory (VRAM) which caused the model's fine-tuning to abruptly terminate intermittently. Such problems were less recurring for BERT-Base and RoBERTa-Base since these two models are smaller in size. With this, RoBERTa-Base was the most cost-effective model since it only had a 1.75% $F_{0.5}$ score difference in comparison to RoBERTa-Large despite utilizing fewer resources during training.

5 Conclusion & Recommendations

In this study, a proof-of-concept deep learning-based model named Balarila was built to detect and correct grammatical errors in the Filipino language effectively. With the adoption of GECToR (Omelianchuk et al., 2020)'s approach, three (3) Balarila models were created and fine-tuned for the task. Each model utilized the open-source pre-trained BERT Tagalog Base (BERT-Base) (Cruz and Cheng, 2019), RoBERTa Tagalog Base (RoBERTa-Base) (Cruz and Cheng, 2021), and RoBERTa Tagalog Large (RoBERTa-Large) (Cruz and Cheng, 2021) transformer encoder models respectively.

Furthermore, an error automation pipeline was also built to create a synthetic dataset of grammatically incorrect Filipino sentences. It was then utilized in fine-tuning and testing the three (3) Balarila models. Two (2) fine-tuning stages from GECToR (Omelianchuk et al., 2020) were adopted: first on a dataset of error-filled sentences then on a dataset of both error-filled and error-free sentences. The created datasets can also be used as a benchmark for Filipino grammar error detection and correction.

However, there are several limitations to this study. First is the coverage of errors. The grammatical errors that Balarila only covers are as follows: duplicate words, morphological errors, wrong use of *nang* vs. *ng*, wrong use of spaces, wrong use of hyphens, wrong use of enclitics, wrong use of punctuation marks, improper word casing, missing words, and wrong use of *ang* and *ng* pronouns.

Another limitation is the rule-based error automation. The generation of a synthetic dataset for this study was hindered by a number of problems such as the vagueness of Word Duplication and Word Deletion algorithms used for the duplicate and missing word errors.

Several recommendations are suggested based on this study's scope and findings. With the limited types of errors covered by Balarila, the first recommendation is greater error coverage. It is recommended that future researchers cover more Filipino grammatical errors in order to produce a more robust and comprehensive model.

The next recommendation is to introduce a more sophisticated error automation algorithm that will improve the performance towards the four (4) error types wherein Balarila performed poorly, as well as resolve the inconsistency towards correcting the same erroneous sentence. An example of this is to target specific words which commonly trigger duplicate and missing words errors, which could be determiners like *ang* and *mga*. Another is to introduce multiple errors upon corrupting a sentence, which was not done.

The use of real data for building and training a GEC model for the Filipino language could also help to remove the bias or inaccuracy that may have been caused by the error automation. As Balarila was trained on a synthetically generated dataset, future researchers are recommended to use a dataset collected from real-world sources in order to represent the actual diversity of data and produce more accurate corrections.

It is also recommended to perform hyperparameter tuning. As an alternative to increasing the dataset size, hyperparameter tuning can also help to improve the model's performance by finding the optimal values for the hyperparameters used during training and prediction. It can also possibly reduce training time, increase model robustness by making it less sensitive to changes in the data, and improve the model's generalization of the data to produce more accurate corrections. In short, conducting hyperparameter tuning can lead to a significant improvement in the model's GEC and GED performance.

Acknowledgements

This research is funded by the Philippine Department of Science and Technology through its 2021 Junior Level Science Scholarships MERIT program.

References

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. Association for Computational Linguistics.

Jan Christian Blaise Cruz and Charibeth Cheng. 2019. Evaluating language model finetuning techniques for low-resource languages.

Jan Christian Blaise Cruz and Charibeth Cheng. 2021. Improving large-scale language models and resources for filipino. *arXiv preprint arXiv:2111.06053*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1:4171–4186.

Muhammad Ali Ghufron and Fathia Rosyida. 2018. The role of grammarly in assessing english as a foreign language (efl) writing. *Lingua Cultura*.

Matthew Philip Go, Nicco Nocon, and Allan Borra. 2017. Gramatika: A grammar checker for the low-resourced filipino language. In *TENCON 2017 - 2017 IEEE Region 10 Conference*.

Matthew Phillip Go and Allan Borra. 2016. Developing an unsupervised grammar checker for Filipino using hybrid n-grams as grammar rules. pages 105–113.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 252–263.

Kalpana Jayavalan and Abu Bakar Razali. 2018. Effectiveness of online grammar checker to improve secondary students' english narrative essay writing. *International Research Journal of Education and Sciences (IRJES)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Komisyon sa Wikang Filipino. 2013. *Ortograpiyang Pambansa*. Accessed: 2022/10/17.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Manolito Octaviano, Matthew Phillip Go, Allan Borra, and Nathaniel Oco. 2016. A corpus-based analysis of filipino writing errors. In *2016 International Conference on Asian Language Processing (IALP)*, pages 95–98.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Aiman Solyman, Wang Zhenyu, Tao Qian, Arafat Abdulgader Mohammed Elhag, Muhammad Toseef, and Zeinab Aleibeid. 2021. Synthetic data with neural machine translation for automatic correction in arabic grammar. *Egyptian Informatics Journal*, 22(3):303–315.