Automated Grammar Error Correction for Urdu using Deep Learning

Anonymous ACL submission

Abstract

Automated Grammar Error Correction (GEC) is an active area of research within the field of Natural Language Processing (NLP), yet its scope remains restricted to English and other resource-rich languages. Urdu is a language that is widely spoken in South Asia. However, due to the lack of annotated datasets no work has been in field of GEC for Urdu language. This paper presents an GEC model for Urdu. In addition, we also present a dataset that contains 1200 pairs of grammatically correct and incorrect sentences in Urdu that was manually curated from children books. Moreover, we also scrapped 400 children stories from Rekhta, an Urdu Literary website, and introduced errors probabilistically to create a dataset with 36,000 pairs of grammatically correct and incorrect sentences. The model that we used was mT5, which is a multilingual version of T5 transformer based model presented by Google. We trained the model in two stages. First, we trained the model on the manually curated dataset. Then, we trained the same model on the dataset that was scrapped from web. Finally, we tested the model by on Wikipedia Edit History dataset containing only grammatical errors which were identified using ERRANT. F0.5 Score, GLEU, Recall and Precision were used as evaluation criteria. The F0.5 scores for the test dataset after fine tuning the MT5 Base model on Raw + Synthetic Dataset are: NOUN INFL 0.63, ADP INFL 0.76, VERB INFL 0.73, VERB FORM 0.66, ADJ INFL 0.76, and PRON INFL 0.74.

Additionally, our study is the first to focus on GEC systems, as to the best of our knowledge, no prior work has been done in this field.

1 Introduction

Automated Grammar Error Correction (GEC) is a natural language processing task that identifies and corrects grammatical errors in text to improve writing clarity and quality. GEC has a variety of applications, including language learning and teaching, writing support, text editing and proofreading, language translation, and content creation and publication (Naghshnejad et al., 2020).

Recently, significant progress has been made in the field of GEC for English and other resource rich languages because of the availability of annotated datasets. However, in case of Indo-Aryan languages such as Urdu, limited progress has been made in the field of GEC because of the lack of availability of annotated datasets.

This paper presents an Automated GEC model for the Urdu language, aiming to improve accurate and fluent written communication. The motivation for undertaking this research comes from the lack of tools like Grammarly for Urdu, which can automatically correct grammatical errors in written English and help users communicate effectively. In addition, an Automated GEC will have a positive impact on content creation, digital communication tools, and learning platforms for Urdu.

Apart from the GEC model, this paper presents an annotated dataset for GEC in Urdu language. The remainder of this paper is structured as follows. Section 2 provides an overview of background information and related work in the field of GEC. Section 3 outlines the preparation of the training and test dataset. Section 4 goes into the details of our proposed methodology, outlining our system design, chosen deep learning models, evaluation criteria, optimization strategies, and implementation details. Section 5 presents the experimental setup for training the model. Section 6 discusses the results obtained, analyzing the effectiveness of our approach. Finally, Section **??** highlights the limitations of the study.

2 Background and Related Work

2.1 Related Work

Although there has been significant progress in the field of GEC for English and other resource rich languages, to the best of our knowledge no has work has been in the field of GEC for Urdu language. Consequently, our literature review focuses on works on GEC for other languages. In addition, we also focus on works that generate synthetic datasets for low resource languages.

In their paper, (Naghshnejad et al., 2020) presented a general survey of the recent Deep Learning based approaches for Grammar Error Handling. Their findings can be in seen Table 1.

Model	Precision	Recall	$F_{0.5}$
RNN NMT (Zheng & Briscoe, 2016)	-	-	39.0
CNN (Chollapatt & Ng, 2018)	65.5	33.1	54.8
RNN+Transformer (Junczys-Dowmunt, 2018)	66.8	34.5	56.3
Copy-augmented Transformer (Zhao, et al., 2019)	71.6	38.7	61.2
PIE (Awashthi, et al., 2019)	68.3	43.2	61.2

Table 1: Table reproduced from (Naghshnejad et al., 2020)

In their paper, (Solyman et al., 2019) proposed a Deep Learning based GEC model for the Arabic language. The authors introduced an encoder-decoder model utilizing multiple convolutional layers and an attention mechanism. They tested the proposed model on the Qatar Arabic Language Bank (QALB) test corpus. Precision, recall, and F1 score were used as evaluation criteria. The model achieved a precision score of 70.23%, a recall score of 72.10%, and an F1 score of 71.14%.

After focusing on GEC systems for different languages, we will now focus on different strategies to create synthetic GEC datasets.

The deliberate injection of errors into grammatically correct sentences has emerged as a critical strategy for overcoming the limited availability of training data. Errors can injected by using a variety of approaches including rule-based systems and round-trip translation (Izumi et al., 2004; Budi Irmawati, 2017; Foster and Andersen, 2009). The limitation of deliberate injection of grammatical errors is that artificial errors should mirror actual errors closely in order to create a dataset is reliable for training and reflective of real-world language use.

Another strategy that is commonly used is the extraction of edit histories from the websites that maintain public revision histories such as Wikipedia. Synthetic dataset generated using this strategy mimics real world dataset because the edits represent actual grammatical mistakes made by humans. However, since these edits also contains other than grammatical mistakes, they need to be filtered. Consequently, make this process challenging (Grundkiewicz and Junczys-Dowmunt, 2014; Boyd, 2018; Faruqui et al., 2018).

In their paper, (Sonawane et al., 2020) combine the two strategies mentioned above to generate a synthetic dataset containing inflectional errors for Hindi language. In addition, they train a base Transformer model and two state of the art English GEC model to create a baseline for GEC in Hindi Language. $F_0.5$ and GLEU score are used as an evaluation criteria. The training dataset is create by using a rule based framework whereas the test dataset is filtering Wikipedia Edit History in Hindi using ERRANT. Since Hindi is similar to Urdu, we follow similar approach as taken by (Sonawane et al., 2020) to develop a GEC model for Urdu.

2.2 Error Annotation Toolkit (ERRANT)

ERRANT (**ERR**or **AN**notation **T**oolkit) is an automatic tool for annotating grammatical errors given an original and corrected sentence pair. (Bryant et al., 2017). ERRANT works by extracting edits from parallel original and corrected sentences and then classifying them according to a dataset-agnostic rule-based framework. ERRANT was initially designed for the English language, but now it has been modified for other languages such as Hindi (Sonawane et al., 2020).

2.3 WikiEdits

WikiEdits is a (Grundkiewicz and Junczys-Dowmunt, 2014) software uses Wikipedia revision histories to extract a parallel corpus of errors. Using this software, we extracted edits in Urdu from a Wikipedia Revision dump dated October 1, 2023. After extracting the edits, we filtered the edits using the following constraints:

- Sentence length should be between 4 and 27.
- Only substitution operations with a Levenstein edit distance of less than 0.3 will be considered.

2.4 Urdu Grammar

Urdu, being a morphologically rich language, employs a complex system of inflections to convey grammatical relationships and meanings. Inflectional errors occur when these grammatical modifications are applied incorrectly, leading to sentences that are grammatically incorrect or unclear. The following categories of inflectional errors are particularly significant in Urdu:

- **NOUN INFL**: Noun inflection errors involve incorrect modifications of nouns to indicate gender, number, or case. For example, using a masculine form of a noun where a feminine form is required, or using a singular noun where a plural is necessary.
- **ADP INFL**: Adposition inflection errors pertain to the incorrect use of prepositions or postpositions that indicate relationships between different parts of a sentence. Errors in adpositions can lead to ambiguity or incorrect interpretations of the sentence structure.
- **VERB INFL**: Verb inflection errors encompass incorrect changes to verbs to reflect tense, aspect, mood, or agreement with the subject in terms of number and gender. These errors can distort the intended time, manner, or completeness of an action.
- **VERB FORM**: Verb form errors involve the use of incorrect verb conjugations or non-standard verb forms. This can include the use of the wrong verb tense or an inappropriate verb form for the grammatical context, affecting the clarity and correctness of the sentence.
- **ADJ INFL**: Adjective inflection errors occur when adjectives fail to agree with the nouns they modify in terms of gender, number, or case. For instance, using a masculine adjective with a feminine noun or a singular adjective with a plural noun.
- **PRON INFL**: Pronoun inflection errors involve the incorrect use of pronouns in terms of case, number, or gender. Pronouns must correctly match the nouns they refer to, and errors in this area can lead to confusion and misinterpretation of the sentence.

3 Dataset

As no work has been done in the field of GEC for Urdu language because of the lack of annotated dataset, we decide to gather one. Our dataset consists of two main parts:

Error Type	Examples
VERB:FORM	جانا(jana) → گیا(gaya), کرنا(karna) → کیا(kiya) *go [inf. → past], do [inf. → past]*
VERB:INFL	ہوا(hua) → ہوئی(hui), کرتا(karta) → کرتے(karte) *happen [m.sing. → f.sing.], do [m.sing. → m.pl.]*
NOUN:INFL	صوبہ(subah) → صوبے(subay), کتا(kutta) → کتے(kutte) *province [nom. → oblique], dog [nom. → oblique]*
ADP:INFL	کا(ka) → کی(ki), کا(ka) → کے(ke) *of [m.sing. → f.sing.], of [m.sing. → pl.]*
PRON:INFL	اسکا(uska) → اسکی(uski), اپنا(apna) → آپکو(aapko) *his [m.sing. → f.sing.], you [erg. → dat.]*
ADJ:INFL	چھوٹا(chhota) → چھوٹے(chhote), دوسرا(dusra) → دوسرے(dusre) *small [m.sing. → m.pl.], other [m.sing. → m.pl.]*

Table 2: Types of Inflectional Errors in Urdu with Examples

1. Raw dataset consisting of 1200 pairs of grammatically correct and incorrect sentences gathered from a variety of primary Urdu textbooks.
2. Synthetic consisting of 3600 pairs of grammatically correct and incorrect sentences, collected by web scrapping children stories from Rekhta and probabilistically introducing errors.

In addition, we also collected a test dataset for evaluating model using WikiEdits and ERRANT (Bryant et al., 2017).

3.1 Raw dataset

We initially collected 1200 pairs of correct and incorrect Urdu sentences. These pairs of sentences were taken from different primary text books which are already verified by multiple Urdu experts. Examples of sentence pairs from the raw dataset can be seen in Table 3.

Input	**Output**
ایک دم چلے جاؤ	فوراً چلے جاؤ
غصّہ سے اس کا چہرہ سرخ ہو گیا	غصّے سے اس کا چہرہ سرخ ہو گیا
اس کا حساب بے باک کر دیا گیا	اس کا حساب بے باق کر دیا گیا
اس نے گھر جانا ہے	اسے گھر جانا ہے
آپ میرے گھر تشریف لاؤ	امی ابو گھر پر نہیں ہیں
اسے کرکٹ کھیلنا نہیں آتی	آپ میرے گھر تشریف لے آؤ
وہ امتحان میں فعل ہو گیا	اسے کرکٹ کھیلنا نہیں آتا
وہ عورت بڑی لڑاکی ہے	وہ عورت بڑی لڑاکا ہے
اس نے مجھ سے بے کلامی کی	اس نے مجھ سے بد کلامی کی

Table 3: Examples of grammatically correct and incorrect sentence pairs from the raw dataset.

Figure 1: Word Cloud of the most common grammatical errors in Urdu Language.

However, this process was very cumbersome as the different Urdu primary textbooks were only available in paperback format and the sentences had to by manually. Consequently, we had to create a synthetic dataset.

3.2 Synthetic Error Generation

In order to create a synthetic dataset, we first scrapped 400 different children stories in Urdu from the popular Urdu website Rekhta and then split each story into separate sentences. Rekhta is an Urdu literary web portal started by Rekhta Foundation, a non-profit organisation dedicated to the preservation and promotion of the Urdu literature. We chose children stories for creating our dataset because they are relatively simpler and are more structured than other texts in Urdu literature, making model training easier.

After splitting each stories into sentences, we found the most frequent words in the dataset in order to introduce synthetic errors. However, since these common words did not always correlate with the common grammatical errors in the Urdu language, we asked experts in Urdu language to provide us with the list of most common grammatical errors in Urdu Language. These errors can be in the Word Cloud in Figure 1

After determining the list of the most common grammatical errors, we introduced these errors into each sentence using a two part process as follows:

1. Determine the total number of errors to be introduced in the sentence. The details of this process are highlighted in Algorithm 1.

2. Randomly select a word from the predefined list of grammatical errors that already exists in the sentence, and then replace it with the incorrect word. Decrement the number of errors by 1. Repeat the process until the number of errors is equal to 0. The details of this process are highlighted in Algorithm 2.

Algorithm 1 GenerateNumberofErrors

$probability \leftarrow$ Random integer between 0 and 100
if $probability > 80$ **then**
  return 3
else if $probability > 50$ **then**
  return 2
else
  return 1
end if

Algorithm 2 GenerateErrorInSentence(sentence)

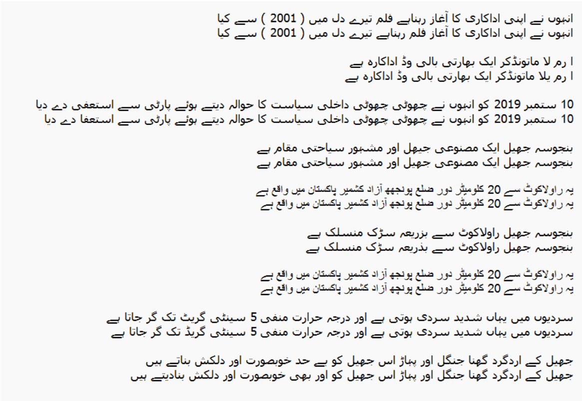
Initialise dictionary $words$
Initialise list $indices$ from 0 to the length of the dictionary $words$
Shuffle $indices$
$number \leftarrow$ GenerateNumberOfErrors()
for each i in $indices$ **do**
  $replace \leftarrow words[i]$
  if $replace$ in $sentence$ **then**
    $replacement \leftarrow words[replace]$
    Swap $replace$ with $replacement$ in $sentence$
    $number \leftarrow number - 1$
  end if
  if $number = 0$ **then**
    return $sentence$
  end if
end for

In the end, our resulting dataset consisted of 36,000 pairs of grammatically correct and incorrect sentences.

3.3 Test Dataset

In order to create a test dataset, we first gathered a corpus of Wikipedia Revision using WikiEdits. Some samples that were obtained using WikiEdits can be viewed in Figure 2

Since the dataset that was obtained, also contained errors other than grammatical errors, we filtered it using ERRANT to extract the grammatical errors. The model that we used for Urdu Part Of Speech (POS) tagging is the StanfordNLP tagger (Qi et al., 2018). The tagger returned Extended Part Of Speech (XPOS) and Universal Part Of Speech (UPOS) both for each word, we used the UPOS


انہوں نے اپنی اداکاری کا آغاز رہناہے فلم تیرے دل میں (2001) سے کیا
انہوں نے اپنی اداکاری کا آغاز فلم رہناہے تیرے دل میں (2001) سے کیا

ا رم لا ماتونڈکر ایک بھارتی بالی وڈ اداکارہ ہے
ا رم یلا ماتونڈکر ایک بھارتی بالی وڈ اداکارہ ہے

10 ستمبر 2019 کو انہوں نے چھوٹی چھوٹی داخلی سیاست کا حوالہ دیتے ہوئے پارٹی سے استعفی دے دیا
10 ستمبر 2019 کو انہوں نے چھوٹی چھوٹی داخلی سیاست کا حوالہ دیتے ہوئے پارٹی سے استعفا دے دیا

بنجوسہ جھیل ایک مصنوعی جیھل اور مشہور سیاحتی مقام ہے
بنجوسہ جھیل ایک مصنوعی جھیل اور مشہور سیاحتی مقام ہے

یہ راولاکوٹ سے 20 کلومیٹر دور ضلع پونجھھ آزاد کشمیر پاکستان میں واقع ہے
یہ راولاکوٹ سے 20 کلومیٹر دور ضلع پونجھ آزاد کشمیر پاکستان میں واقع ہے

بنجوسہ جھیل راولاکوٹ سے بزریعہ سڑک منسلک ہے
بنجوسہ جھیل راولاکوٹ سے بذریعہ سڑک منسلک ہے

یہ راولاکوٹ سے 20 کلومیٹر دور ضلع پونجھ آزاد کشمیر پاکستان میں واقع ہے
یہ راولاکوٹ سے 20 کلومیٹر دور ضلع پونچھ آزاد کشمیر پاکستان میں واقع ہے

سردیوں میں یہاں شدید سردی ہوتی ہے اور درجہ حرارت منفی 5 سینٹی گریٹ تک گر جاتا ہے
سردیوں میں یہاں شدید سردی ہوتی ہے اور درجہ حرارت منفی 5 سینٹی گریڈ تک گر جاتا ہے

جھیل کے اردگرد گھنا جنگل اور پہاڑ اس جھیل کو بے حد خوبصورت اور دلکش بناتے ہیں
جھیل کے اردگرد گھنا جنگل اور پہاڑ اس جھیل کو اور بھی خوبصورت اور دلکش بنادیتے ہیں

Figure 2: Samples from the Wikipedia Revision History Dataset

tags because there is no tagset conversion from XPOS to UPOS for Urdu. The distribution of the frequency of the different error types as determined by ERRANT in the Wikipedia Revision History dataset can be seen in Figure 3

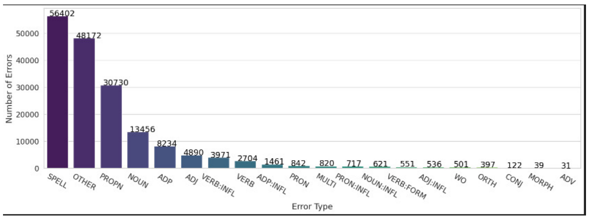


Figure 3: Frequency of error types as determined by ERRANT in the Wikipedia Revision History dataset.

However, for the purpose for this study we will only consider INFLECTIONAL and VERB:FORM errors as these are more common grammatical errors. In addition, we discarded other edits because of due to errors in POS tagging. For example, Dates were incorrectly tagged as ADP, ADJ. In the end, our test dataset contained approximately 9,000 pairs of grammatically correct and incorrect sentences. We also checked the dataset for offensive content by random sampling some pairs and then checking them manually.

4 Methodology

4.1 System Design

In order to develop an automatic GEC model for Urdu language, we decided use to the MT5 (Xue et al., 2021) developed by Google Research owing to its promising performance for GEC for low resource languages (Gomez et al., 2023).

Our MT5 model was trained in multiple stages. Initially, we trained a pre-trained MT5 model solely on the Raw Dataset. Subsequently, we conducted further training on the entire dataset, encompassing both the Raw and Synthetic datasets. Finally, the model underwent evaluation using the test dataset. Evaluation metrics including F0.5 score, GLEU, Recall, and Precision were computed for each error type identified by ERRANT, both with and without the synthetic dataset.

4.2 Transformer Model

The T5 (Text-To-Text Transfer Transformer) model is a transformer-based architecture introduced by Google Research (Raffel et al., 2019). Unlike previous models that were task-specific, T5 is designed to handle various natural language processing tasks through a unified text-to-text framework. It achieves this by framing all tasks as text-to-text transformations, where both inputs and outputs are in natural language text format. T5 is trained on large-scale datasets using a multi-task objective, enabling it to perform well across a wide range of NLP tasks such as translation, summarisation, and question answering.

The Multilingual T5 (MT5) model is an extension of T5 that is specifically trained on multilingual data (Xue et al., 2021). MT5 is pre-trained on a diverse range of languages including Urdu, allowing it to understand and generate text in multiple languages. This multilingual capability is achieved by incorporating language-specific tokens during training, enabling the model to handle language switching seamlessly. MT5 has been shown to perform competitively across various language tasks, making it a valuable tool for multilingual applications.

4.3 Evaluation Criteria

Based on the literature review that we conducted, we decided to the evaluate the performance of our model using the following criteria:

1. F0.5 Score

2. Precision

3. Recall

4. Generalised Language Evaluation Understanding (GLEU) (Napoles et al., 2015)

4.3.1 F0.5 Score

The F0.5 score is a weighted harmonic mean of precision and recall, with more emphasis on precision.

$$F_{0.5} = 1.25 \cdot \frac{\text{precision} \cdot \text{recall}}{0.25 \cdot \text{precision} + \text{recall}} \quad (1)$$

4.3.2 Precision

Precision quantifies the number of correct positive predictions made out of all positive predictions made.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (2)$$

4.3.3 Recall

Recall measures the number of correct positive predictions made out of all actual positives in the dataset.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (3)$$

4.3.4 GLEU

GLEU (Generalized Language Understanding Evaluation) is a metric used to evaluate the performance of grammatical error correction systems. It compares the generated (corrected) sentence to a reference sentence (the original, grammatically correct version) by looking at how many n-grams (sequences of words of length n) they share (Napoles et al., 2015). It is calculated as follows:

1. All sub-sequences of 1, 2, 3, or 4 tokens in the output and target sequences (n-grams) are recorded.

2. Recall is calculated as the number of matching n-grams to the number of total n-grams in the target (ground truth) sequence.

$$\text{Recall} = \frac{|\text{ Matching n-grams }|}{|\text{ Total target n-grams }|} \quad (4)$$

3. Precision is computed as the ratio of the number of matching n-grams to the number of total n-grams in the generated output sequence.

$$\text{Precision} = \frac{|\text{ Matching n-grams }|}{|\text{ Total generated n-grams }|} \quad (5)$$

4. The GLEU score is the minimum of recall and precision. Its range is always between 0 (no matches) and 1 (all matches).

$$\text{GLEU} = \min(\text{Recall}, \text{Precision}) \quad (6)$$

4.4 Implementation Details

In order to implement our model, we used Hugging Face's *Transformers* library and the *PyTorch* framework. Furthermore, we trained our model on the NVIDIA RTX TITAN GPU with 24GB Ram.

4.5 Optimization Strategies

In order to speed up and stabilize the training process, we employed the following strategies:

1. Gradient Clipping
2. Gradient Accumulation
3. Mixed Precision Training

4.5.1 Gradient Clipping

Gradient clipping is a technique used to prevent the exploding gradient problem during training. It involves setting a threshold value, and if the norm of the gradients exceeds this threshold, the gradients are scaled down proportionally to ensure they do not grow too large. This helps stabilize the training process and prevents model parameters from diverging.

4.5.2 Gradient Accumulation

Gradient accumulation is a strategy to effectively utilize hardware resources during training, particularly when working with limited GPU memory. Instead of updating the model's parameters after processing each batch, gradients are accumulated over multiple batches before performing a single parameter update. This reduces the frequency of parameter updates and allows for larger effective batch sizes without increasing memory requirements.

4.5.3 Mixed Precision Training

Mixed precision training leverages the capabilities of modern GPUs to accelerate training by using lower precision floating-point numbers (e.g., half-precision floating-point numbers) for certain computations while maintaining higher precision for others. This technique reduces memory usage and computational overhead, resulting in faster training times. Additionally, mixed precision training often includes automatic loss scaling to mitigate numerical stability issues associated with lower precision arithmetic.

5 Experiments

5.1 Train Test Split

The train and validation splits were created by splitting the Raw and Synthetic Dataset with a 90:10

ratio. The entire dataset from Wikipedia Revision after filtration by ERRANT was used as the test dataset.

5.2 Model Selection

The MT5 has multiple variants. The different variants along with the number of parameters are listed as follows:

1. MT5 Small: 250M parameters
2. MT5 Base: 580M parameters
3. MT5 Large: 1.3B parameters
4. MT5 Extra Large: 2.5B parameters
5. MT5 XXL: 5B parameters

We first tried the MT5 Large but it crashed due to out of memory error. Then we tried the MT5 Small , but model was outputting random characters. Consequently, we only used the MT5 Base model.

5.3 Experimental Setup

We trained the model using a two step process. First, we fine tuned the MT5 Base model only on the Raw Dataset for 180 epochs. Second, we further trained the MT5 Base model on the Raw+Synthetic Dataset for 60 epochs. During the entire training the process, rest of the hyperparameters were kept constant and be seen in Table 4.

Hyperparameter	**Value**
Optimizer	Adam
Learning Rate	3×10^{-4}
Weight Decay	1×10^{-5}
Scheduler	Step Learning Rate
Step Size	10
Learning Rate Multiplicative Factor	0.5
Batch Size	4
Gradient Accumulation Steps	4
Max Gradient Norm	1.0
Mixed Precision Training	Float16 for loss propagation and Float32 for weights

Table 4: Experimental Setup

The entire training process took us approximately 33 hours for one run, so we didn't perform multiple runs.

6 Results

The individual performance of MT5 Base model for different error types, both with and without fine tuning on the Synthetic Dataset is summarised in Table 5. While analysing these results, we identified some notable patterns.

6.0.1 Overall Trends

The results indicate a clear trend of improvement when the Synthetic dataset is combined with the Raw dataset during the training process. Across all metrics, the scores for the training only using the **Raw + Synthetic** Datasets are consistently higher than those training on the **Raw** dataset alone. This suggests that incorporating synthetic data enhances the model's overall performance, thus enabling it to deal with grammatical errors more effectively. Furthermore, since the improvements are not limited to a specific metric or error type, but are rather broad, shows the robustness of synthetic error generation for creating a GEC dataset.

6.0.2 Comparison of different evaluation criteria

Across different evaluation criteria, the inclusion of synthetic data leads to significant improvements. GLEU scores, which measure the fluency and accuracy of generated text, show a general upward trend, indicating better language generation. The F0.5 scores, which emphasizes precision, also sees a significant increase, reflecting improved accuracy in correcting errors. Precision exhibit the most dramatic enhancements, suggesting that the model's predictions are more accurate with synthetic data. Recall also rises consistently, highlighting the model's improved capability in identifying and correcting a broader range of errors, ensuring that fewer errors are missed.

6.0.3 Comparison of different error types

The inclusion of synthetic dataset impacted some error types more than others. Some of the key observations are as follows:

- Pronoun inflection (PRON INFL) and noun inflection (NOUN INFL) benefit significantly in terms of both precision and recall.
- Adjective inflection (ADJ INFL) shows the highest overall gains
- Improvements in verb form (VERB FORM) are notable, especially in the F0.5 score.

Overall, based on these observations we can conclude that synthetic data can be particularly beneficial for injecting grammatical errors.

7 Limitations

In this section, we highlight the major limitations of our work.

	Training only on Raw Dataset						Training on Raw + Synthetic Dataset					
	NOUN INFL	ADP INFL	VERB INFL	VERB FORM	ADJ INFL	PRON INFL	NOUN INFL	ADP INFL	VERB INFL	VERB FORM	ADJ INFL	PRON INFL
Average GLEU Score	0.51	0.61	0.65	0.61	0.62	0.7	0.6	0.72	0.75	0.71	0.73	0.74
Average F0.5 Score	0.4	0.5	0.49	0.42	0.47	0.57	0.63	0.76	0.73	0.66	0.76	0.74
Average Precision	0.63	0.69	0.69	0.65	0.66	0.74	0.79	0.86	0.85	0.79	0.87	0.85
Average Recall	0.58	0.63	0.62	0.58	0.6	0.69	0.79	0.86	0.85	0.8	0.87	0.85

Table 5: Average GLUE, F0.5, Recall and Precision for various error types on the test dataset.

As stated earlier, there was no existing dataset for Urdu GEC. This meant that we had to curate a dataset using manually. However, creating a dataset manually is a cumbersome process. As a result, we scrapped a dataset from the website and artificial injected grammatical errors. Consequently, our model only deals with substitution INFLECTIONAL errors.

During the data collection phase, we scrapped children stories from as they contained shorter and simpler sentences, in order to facilitate model training. However, this means that our model can only deal with simple sentences of a moderate length at one time.

As mentioned earlier, our algorithm for artificially injecting errors is designed such that a sentence contain either 1, 2 or 3 grammatical. However, ERRANT returns only a single error per sentence. This means that we cannot effectively evaluate our model for sentences that contain multiple errors.

References

Adriane Boyd. 2018. Using Wikipedia edits in low resource grammatical error correction. In *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 79–84, Brussels, Belgium. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Yuji Matsumoto Budi Irmawati, Hiroyuki Shindo. 2017. Generating artificial error data for indonesian preposition error corrections. *International Journal of Technology*, 8(3):549–558.

Manaal Faruqui, Ellie Pavlick, Ian Tenney, and Dipanjan Das. 2018. WikiAtomicEdits: A multilingual corpus of Wikipedia edits for modeling language and discourse. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 305–315, Brussels, Belgium. Association for Computational Linguistics.

Jennifer Foster and Øistein E. Andersen. 2009. Generrate: Generating errors for use in grammatical error detection. In *BEA@NAACL*.

Frank Gomez, Alla Rozovskaya, and Dan Roth. 2023. A low-resource approach to the grammatical error correction of ukrainian. pages 114–120.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2014. The wiked error corpus: A corpus of corrective wikipedia edits and its application to grammatical error correction. In *Advances in Natural Language Processing*, pages 478–490, Cham. Springer International Publishing.

Emi Izumi, Kiyotaka Uchimoto, and Hitoshi Isahara. 2004. The overview of the SST speech corpus of Japanese learner English and evaluation through the experiment on automatic detection of learners' errors. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Mina Naghshnejad, Tarun Joshi, and Vijayan N. Nair. 2020. Recent trends in the use of deep learning models for grammar error handling. *CoRR*, abs/2009.02358.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, Beijing, China. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal Dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Aiman Solyman, Zhenyu Wang, and Qian Tao. 2019. Proposed model for arabic grammar error correction based on convolutional neural network. In *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, pages 1–6.

Ankur Sonawane, Sujeet Kumar Vishwakarma, Bhavana Srivastava, and Anil Kumar Singh. 2020. Generating inflectional errors for grammatical error correction in Hindi. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 165–171, Suzhou, China. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. *Preprint*, arXiv:2010.11934.