# Beyond English: Evaluating LLMs for Arabic Grammatical Error Correction

**Sang Yun Kwon**[ξ] **Gagan Bhatia** [ξ] **El Moatez Billah Nagoudi**[ξ]
**Muhammad Abdul-Mageed**[ξ,ω]
[ξ]Deep Learning & Natural Language Processing Group, The University of British Columbia
[ω]Department of Natural Language Processing & Department of Machine Learning, MBZUAI
{skwon01@student.,gagan30@student.,muhammad.mageed@}ubc.ca

## Abstract

Large language models (LLMs) finetuned to follow human instruction have recently exhibited significant capabilities in various English NLP tasks. However, their performance in grammatical error correction (GEC), especially on languages other than English, remains significantly unexplored. In this work, we evaluate the abilities of instruction finetuned LLMs in Arabic GEC, a complex task due to Arabic's rich morphology. Our findings suggest that various prompting methods, coupled with (in-context) few-shot learning, demonstrate considerable effectiveness, with GPT-4 achieving up to 65.49 $F_1$ score under expert prompting (approximately 5 points higher than our established baseline). Despite these positive results, we find that instruction finetuned models, regardless of their size, are still outperformed by fully finetuned ones, even if they are significantly smaller in size. This disparity highlights substantial room for improvements for LLMs. Inspired by methods used in low-resource machine translation, we also develop a method exploiting synthetic data that significantly outperforms previous models on two standard Arabic benchmarks. Our best model achieves a new SOTA on Arabic GEC, with 73.29 and 73.26 $F_1$ on the 2014 and 2015 QALB datasets, respectively, compared to peer-reviewed published baselines.

## 1 Introduction

As interest in second language learning continues to grow, ensuring the accuracy and effectiveness of written language becomes increasingly significant for pedagogical tools and language evaluation (Rothe et al., 2021; Tarnavskyi et al., 2022). A key component in this respect is grammatical error correction (GEC), a sub-area of natural language generation (NLG), which analyzes written text to automatically detect and correct diverse grammatical errors. Figure 1 shows an instance of GEC from Mohit et al. (2014). Despite the growing attention to GEC, it is predominantly studied within

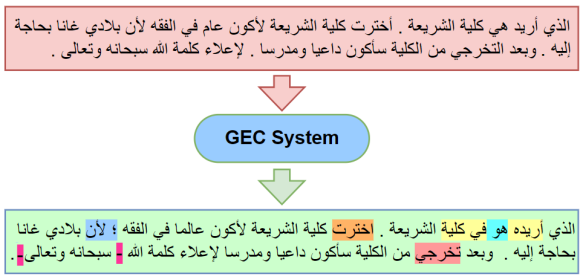


Figure 1: An example of an Arabic GEC system showcasing six types of errors: *character replacement*, *missing word*, *hamza error*, *missing punctuation*, *additional character*, and *punctuation confusion*.

the English language. Extending GEC systems to other languages presents significant challenge, due to lack of high-quality parallel data and/or inherent challenges in these languages. Recognizing this, our work focuses on Arabic. In addition to being less-explored for GEC (Mohit et al., 2014; Rozovskaya et al., 2015a; Mohit et al., 2014; Rozovskaya et al., 2015a; Solyman et al., 2022; Alhafni et al., 2023), Arabic has complex grammar and rich morphology that present significant challenges and further motivate our work.

Focusing primarily on English, the field of GEC has witnessed significant advancements, specifically with the emergence of sequence-to-sequence (seq2seq) (Chollampatt and Ng, 2018; Gong et al., 2022) and sequence-to-edit approaches (seq2edit) (Awasthi et al., 2019; Omelianchuk et al., 2020) achieving SoTA results in the CONLL-2014 (Ng et al., 2014) and the BEA-2019 shared task (Bryant et al., 2019), respectively. In spite of the efficacy of these approaches, they rely heavily on large amounts of labeled data. This poses issues in low-resource scenarios (Feng et al., 2021). Yet, scaled up language models, *aka* large language models (LLMs) have recently demonstrated remarkable potential in various NLP tasks. The core strength of LLMs lies in their capacity to gen-



*Proceedings of the The First Arabic Natural Language Processing Conference (ArabicNLP 2023)*, pages 101–119
December 7, 2023 ©2023 Association for Computational Linguistics

eralize across a wide range of languages and tasks, and in-context learning (ICL), enabling them to handle various NLP tasks with just a few examples (i.e., few-shot learning). A key strategy for LLMs is *instruction fine-tuning*, where they are refined on a collection of tasks formulated as instructions (Wei et al., 2022a). This process amplifies the models' ability to respond accurately to directives, reducing the need for few-shot examples (Ouyang et al., 2022; Wei et al., 2022b; Sanh et al., 2021).

Given the ability of LLMs to adeptly address the low-resource challenge, we investigate them in the context of GEC. Focusing primarily on ChatGPT, we examine the effectiveness of various prompting strategies such as few-shot chain of thought (CoT) prompting (Kojima et al., 2022) and expert prompting (Xu et al., 2023). Our research extends the realm of GEC research by concentrating on the unique challenges posed by Arabic. Drawing upon the work of Junczys-Dowmunt et al. (2018a), we frame these challenges within the context of a low-resource MT task. We then carefully conduct a thorough comparison of the different methodologies employed in addressing GEC in Arabic. Our key contributions in this paper are as follows:

1. We conduct a comprehensive investigation of the potential of LLMs for tasks involving GEC in Arabic.

2. We methodically investigate the utility of different prompting methods for generating synthetic data with ChatGPT for GEC.

3. We further carry out in-depth comparisons between several approaches (seq2seq, seq2edit, and instruction fine-tuning) for Arabic GEC (AGEC), allowing us to offer novel insights as to the utility of these approaches.

The rest of this paper is organized as follows: In Section 2, we review related work with a particular emphasis on Arabic. In Section 3, we outline our experimental setups. We present our experiments on LLMs and prompting strategies in Section 4. In Section 5, we introduce our seq2seq approach along with data augmentation techniques; Section 6 discusses our seq2edit approach. In Section 7, we conduct a comprehensive analysis of our best model. We discuss our results in Section 8, and conclude in Section 9.

## 2 Related Work

**Progress in GEC.** Pretrained Transformer models have reframed GEC as an MT task, achieving SoTA results (Ng et al., 2014; Felice et al., 2014; Junczys-Dowmunt et al., 2018b; Grundkiewicz et al., 2019). In contrast, sequence2edit approaches view the task as text-to-edit, converting input sentences into edit operations to produce corrected sentences (Malmi et al., 2019; Awasthi et al., 2019; Omelianchuk et al., 2020). These approaches both streamline the training process and enhance model accuracy. Further progress has also been made through methods such as instruction fine-tuning (Chung et al., 2022) and innovative prompting techniques, such as CoT (Kojima et al., 2022) and Expert (Xu et al., 2023) prompting. Recent applications of LLMs, like ChatGPT in GEC, highlight their potential. We provide further details on each of these methods in Appendix A.

**Arabic GEC.** Challenges in AGEC stem from the complexity and morphological richness of Arabic. Arabic, being a collection of a diverse array of languages and dialectal varieties with Modern Standard Arabic (MSA) as a contemporary variety, is further complicated by the optional use of diacritics. This introduces orthographic ambiguity, further complicating GEC in Arabic (Abdul-Mageed et al., 2020; Belkebir and Habash, 2021). Despite these challenges, progress in AGEC has been made. This includes development of benchmark datasets through the QALB-2014 and 2015 shared tasks (Mohit et al., 2014; Rozovskaya et al., 2015b; Habash and Palfreyman, 2022), and introduction of synthetic datasets (Solyman et al., 2021, 2023). As for model development, character-level seq2seq models (Watson et al., 2018) and other novel approaches are shown to be effective on AGEC L1 data. Further details about progress in AGEC are provided in Appendix A. Despite this progress, no exploration has been undertaken into the utility of using ChatGPT (or other LLMs) for AGEC. Moreover, substantial work remains in exploring synthetic data generation, including the use of LLMs and the adoption of diverse machine learning approaches. Our research aims to address these gap.

## 3 Experimental Setup

### 3.1 Datasets

In this study, we make use of the QALB-2014 (Mohit et al., 2014) and 2015 (Rozovskaya et al., 2015b) datasets to evaluate the performance of our

| Dataset | Statistics | Train | Dev | Test | Level |
|---|---|---|---|---|---|
| QALB-2014 | Number of sents. | 19,411 | 1,017 | 968 | L1 |
| | Number of words. | 1,021,165 | 54,000 | 51,000 | L1 |
| | Number of error. | 306,000 | 16,000 | 16,000 | L1 |
| QALB-2015 | Number of sents. | 310 | 154 | 920 | L2 |
| | Number of words. | 43,353 | 24,742 | 48,547 | L2 |
| | Number of error. | 13,200 | 7,300 | 13,000 | L2 |

Table 1: Statistics for QALB-2014 and 2015 Train, development (Dev), and Test datasets.

models. Both datasets make use of the QALB corpus (Zaghouani et al., 2014), a manually corrected collection of Arabic texts. These texts include online commentaries from Aljazeera articles in MSA by L1 native speakers, as well as texts produced by L2 learners of Arabic. Both the QALB 2014 and 2015 datasets are split into training (Train), development (Dev), and test (Test) sets based on their annotated dates. QALB 2015 includes L1 commentaries and L2 texts that cover different genres and error types. For the purposes of our study, we exclusively use the L1 test set (2015), as we focus on sentence-level AGEC, where L2 test sets are document-level. We used Train, Dev, and Test splits described in Table 1.

### 3.2 Evaluation

**Metrics.** For evaluation, we utilize the overlap-based metric MaxMatch ($M^2$) (Dahlmeier and Ng, 2012), which aligns source and hypothesis sentences based on Levenshtein distance , selecting maximal matching edits, scoring the precision (P), recall (R), and $F_1$ measure. Moreover, we report the $F_{0.5}$ score , a variation of the $F_1$ score that places twice as much weight on precision than on recall. This reflects a consensus, in alignment with recent works on GEC, that precision holds greater importance than error correction in GEC systems. Importantly, we use the exact scripts provided from the shared task for evaluation, ensuring consistency with other studies.

### 3.3 Models & Fine-tuning

**LLMs.** To evaluate the capabilities of LLMs for AGEC, we prompt and fine-tune LLMs of varying sizes, including LLaMA-7B (Touvron et al., 2023), Vicuna-13B (Chiang et al., 2023), Bactrian-$X_{bloom}$-7B (Li et al., 2023), and Bactrian-$X_{llama}$-7B (Li et al., 2023). For experiments with ChatGPT, we use the official API to prompt ChatGPT-3.5 Turbo and GPT-4. We instruction fine-tune each smaller model for 4 epochs using a learning rate of 2e-5 and a batch size of 4. We then pick the best-performing model on our Dev, then report on our blind Test.

**Seq2seq models.** Our baseline settings for seq2seq models include AraBart (Eddine et al., 2022) and $AraT5_{v2}$ (Nagoudi et al., 2022), both of which are text-to-text transformers specifically tailored for Arabic. We also evaluate the performance of the mT0 (Muennighoff et al., 2022) and mT5 (Xue et al., 2020) variants of the T5 model (Raffel et al., 2020), both configured for multilingual tasks. Each model is fine-tuned for 50 epochs, with an early stopping patience of 5 using a learning rate of 5e-5 and a batch size of 32. These models serve as the baseline for comparison throughout our experiments.

**Seq2edit models.** $ARBERT_{v2}$ and $MARBERT_{v2}$ (Abdul-Mageed et al., 2021) serve as the baselines for our seq2edit experiments. We fine-tune each model for 100 epochs for each training stage, employing a learning rate of 1e-5 and a batch size of 4, with an early stopping patience of 5.

All models are trained for three runs, with seeds of 22, 32, and 42. We then select the best-performing model based on our Dev data for blind-testing on the Test sets. *We report the mean score of the three runs, along with its standard deviation.* Results on the Dev set, and more details regarding hyperparameters are provided in Appendix 15, and Appendix 14.

## 4 LLMs and Prompting Techniques

This section outlines our experiments designed to instruction fine-tune LLMs and explore different prompting methods for ChatGPT in the context of AGEC. We begin by experimenting with various prompting strategies using ChatGPT, comparing its performance against smaller LLMs and our listed baselines. We evaluate the performance of ChatGPT-3.5 Turbo (ChatGPT) and GPT-4, under two prompting strategies: ***Few-shot CoT*** (Fang et al., 2023) and ***Expert Prompting*** (Xu et al., 2023). We now describe our prompting strategies.

### 4.1 ChatGPT Prompting

**Preliminary experiment.** Initially, we experiment with a diverse set of prompt templates to assess ChatGPT's capabilities in zero-shot learning as well as two aspects of few-shot learning: vanilla few-shot and few-shot CoT (Fang et al., 2023). We also experiment with prompts in both English and Arabic. However, we discover that the responses from these prompt templates contain extraneous

explanations and are disorganized, necessitating substantial preprocessing for compatibility with the $M^2$ scorer. This problem is particularly notable in the zero-shot and Arabic prompt setups, which fails to yield output we can automatically evaluate.

**Few-shot CoT.** Adopting the few-shot CoT prompt design strategy from Kojima et al. (2022) and Fang et al. (2023), we implement a two-stage approach. Initially, we engage in *'reasoning extraction'*, prompting the model to formulate an elaborate reasoning pathway. This is followed by an *'answer extraction'* phase, where the reasoning text is combined with an answer-specific trigger sentence to form a comprehensive prompt. In our few-shot CoT settings, we include labeled instances from the Dev set in our prompts to implement ICL, facilitating learning from examples (Brown et al., 2020). This involves providing erroneous sentences, labeled *<input> SRC </input>*, along with their corrected versions, labeled *<output> TGT </output>*, from the original Dev set.

**Expert prompting.** Xu et al. (2023) introduces a novel strategy, which leverages the expert-like capabilities of LLMs. This method involves assigning expert personas to LLMs, providing specific instructions to enhance the relevance and quality of the generated responses. Following the framework of Xu et al. (2023), we ensure that our AGEC correction tool exhibits three key characteristics: being ***distinguished***, ***informative***, and ***automatic*** during the *'reasoning extraction'* stage of our prompt. To achieve this, we employ a distinct and informative collection of various error types as proposed in the Arabic Learner Corpus taxonomy (Alfaifi and Atwell, 2012). We then prompt to automate the system by instructing it to operate on sentences labeled with *<input>* and *<output>* tags. Both prompts are illustrated in Figure 2.

### 4.2 ChatGPT Results.

Table 2 presents the performance of ChatGPT under different prompting strategies, compared to the baseline settings. We observe improvements, particularly as we progress from the one-shot to five-shot configurations for both the few-shot CoT and expert prompting (EP) strategies. Under the CoT prompt, ChatGPT's $F_{1.0}$ score increases from 53.59 in the one-shot setting to 62.04 in the five-shot setting. A similar upward trend is evident with the EP strategy, where the $F_{1.0}$ score rises from 55.56 (one-shot) to 63.98 (five-shot). Among all experiments involving ChatGPT, the three-shot and five-shot settings of GPT-4, CoT, achieve the highest scores, with $F_{1.0}$ of 63.98 and 65.49, respectively.

| Settings | Models | Exact Match | | | |
|---|---|---|---|---|---|
| | | P | R | $F_{1.0}$ | $F_{0.5}$ |
| Baselines | mT0 | 70.76 $^{\pm 0.03}$ | 50.78 $^{\pm 0.07}$ | 59.12 $^{\pm 0.05}$ | 65.59 $^{\pm 0.03}$ |
| | mT5 | 70.64 $^{\pm 0.12}$ | 50.16 $^{\pm 0.05}$ | 58.66 $^{\pm 0.05}$ | 65.30 $^{\pm 0.09}$ |
| | AraBART | 70.71 $^{\pm 0.06}$ | 60.46 $^{\pm 0.04}$ | 65.18 $^{\pm 0.07}$ | 68.39 $^{\pm 0.08}$ |
| | $AraT5_{v2}$ | **73.04** $^{\pm 0.10}$ | **63.09** $^{\pm 0.15}$ | **67.70** $^{\pm 0.12}$ | **70.81** $^{\pm 0.11}$ |
| + CoT | ChatGPT (1-shot) | 58.71 | 49.29 | 53.59 | 56.55 |
| | ChatGPT (3-shot) | 64.60 | 60.37 | 62.41 | 63.71 |
| | ChatGPT (5-shot) | 64.70 | 59.59 | 62.04 | 63.61 |
| + EP | ChatGPT (1-shot) | 60.49 | 51.37 | 55.56 | 58.42 |
| | ChatGPT (3-shot) | 65.83 | 61.41 | 63.54 | 64.90 |
| | ChatGPT (5-shot) | 66.53 | 61.62 | 63.98 | 65.49 |
| + CoT | GPT4 (1-shot) * | – | – | – | – |
| | GPT4 (3-shot) | 69.31 | 59.24 | 63.88 | 67.03 |
| | GPT4 (5-shot) | 69.46 | 61.96 | 65.49 | 67.82 |

Table 2: Performance of ChatGPT under different prompting strategies on QALB-2014 Test set.* Results for QALB-2015 Test and GPT4 1-shot are not included due to the high cost in producing these results, and a pattern has already been established showing that performance increases as we increase the number of N-shot examples. More details are in Appendix B.2.

### 4.3 Instruction-Finetuning LLMs

**Fine-tuning LLMs.** To instruct fine-tune *relatively* large models, *henceforth* just LLMs, we first train these models on the translated Alpaca dataset (Taori et al., 2023) [1] to allow the models to gain deeper understanding of the Arabic language and its complexities. Following this, we further fine-tune the models on the QALB dataset, to specifically target the task of GEC. Then, we employ well-structured task instructions and input prompts, enabling the models to take on GEC tasks. Each model is assigned a task, given an instruction and an input for output generation. We provide an illustration of the instructions we use for model training in Appendix B.

**LLM results.** As shown in Figure 3, larger models such as Vicuna-13B and models trained on multilingual datasets like Bactrian-$X_{llama}$-7B, and Bactrian-$X_{bloom}$-7B exhibit an overall trend of better performance, achieving $F_1$ of 58.30, 50.1, and 52.5, respectively. Despite these improvements, it is noteworthy that all models fall short of ChatGPT's. This reaffirms ChatGPT's superior ability on AGEC.

## 5 Data Augmentation

Motivated by the significant improvements observed in low-resource GEC tasks in languages

[1] We translate the Alpaca datasets using NLLB MT model (Costa-jussà et al., 2022)

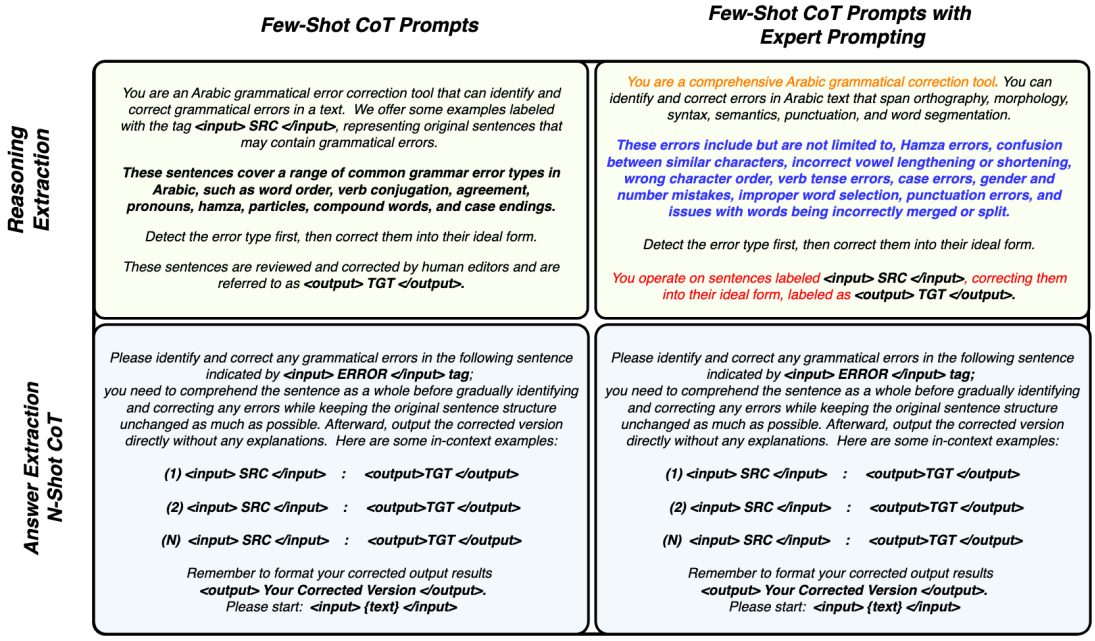


Figure 2: Illustration of Few-Shot CoT and Expert Prompts for Arabic Grammatical Error Correction.

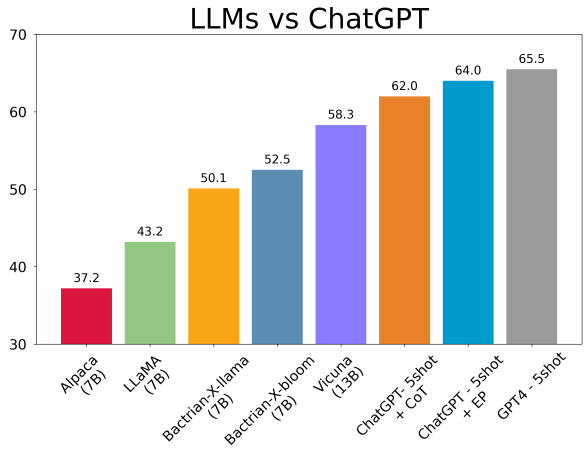


Figure 3: Comparison of $F_1$ scores between LLMs and ChatGPT on the QALB-2014 Test set.

such as German, Russian, and Czech through synthetic data (Flachs et al., 2021), and recognizing the recent efforts to develop synthetic data for AGEC (Solyman et al., 2021), we experiment with three distinctive data augmentation methods.

**ChatGPT as corruptor.** With slight adaptation to our original prompt, we engage ChatGPT as an AI model with the role of introducing grammatical errors into Arabic text to generate artificial data. We randomly sample 10,000 correct sentences from the QALB-2014 Train set and, using the taxonomy put forth by the Arabic Learner Corpus (Alfaifi and Atwell, 2012), prompt ChatGPT to corrupt these, creating a parallel dataset. We refer to the resulting dataset as **syntheticGPT**.

**Reverse noising.** We adopt a *reverse noising* approach (Xie et al., 2018), training a reverse model that converts clean sentences $Y$ into noisy counterparts $X$. This involves implementing a standard beam search to create noisy targets $\hat{Y}$ from clean input sentences $Y$. Our approach incorporates two types of reverse models: the first trains on both QALB-2014 and 2015 gold datasets, and the second on the syntheticGPT dataset. Subsequently we generate a parallel dataset using commentaries from the same newspaper domain as our primary clean inputs, matching the original Train data. We name the respective parallel datasets **reverseGold**, and **reverseGPT**.

**Data augmentation evaluation.** To evaluate the efficacy of ChatGPT in generating artificial data, we select $10,000$ parallel sentences from syntheticGPT, $10,000$ examples from reverseGPT, and $10,000$ parallel sentences from the original training set. We then further fine-tune each model on the original training dataset and the two synthetically generated reverse noised datasets, aiming to assess if these artificially crafted datasets can replace the gold standard training set. Figure 4 shows our results. In our initial tests (Figure 4.a), fine-tuning the AraT5$_{v2}$ model exclusively on the $10,000$ sentences from syntheticGPT, registers an $F_1$ of $65.87$, and reverseGPT an $F_1$ score of $46.85$ falling behind the original QALB 2014 training data (which records an $F_1$ of $68.34$). Following this, when fur-

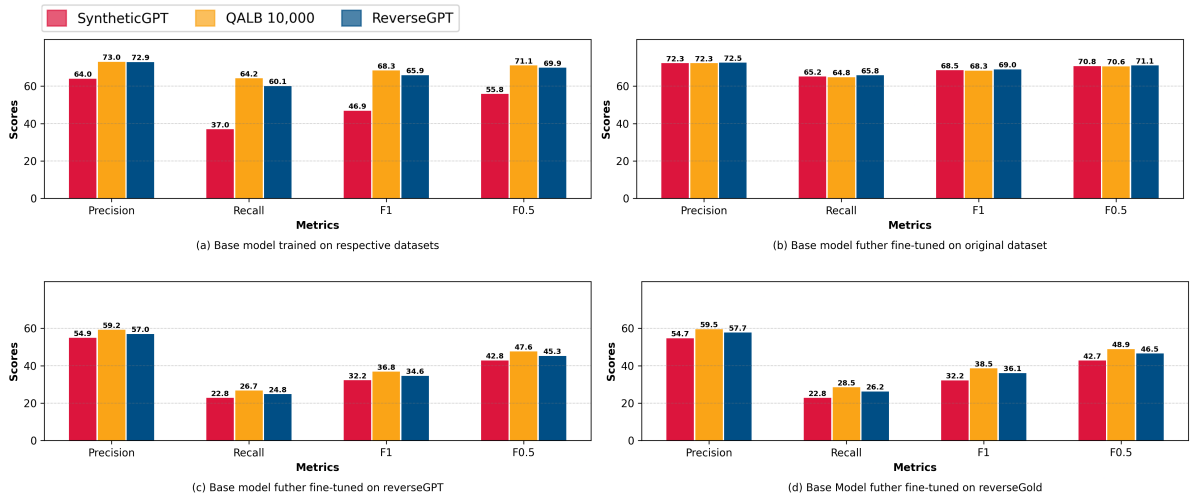


Figure 4: Scores of models fine-tuned on $10,000$ parallel sentences from different sources: Original training data, syntheticGPT, and reverseGPT evaluated on the QALB-2014 Test set.

ther fine-tuned on the original training set (Figure 4.b). We find that both syntheticGPT and the reverseGPT surpass model fine-tuned on equivalent-sized gold dataset, with $F_1$ of 69.01 and 68.54, respectively. This confirms the utility of ChatGPT for generating synthetic data. Conversely, when we further fine-tune the model with the two reverse noised datasets (see Figures 4.c and d), we observe a sharp decline in performance. This emphasizes the critical importance of relevant, high-quality synthetic data over randomly generated samples.

### 5.1 Decoding Methods.

Decoding strategies for text generation are essential and can vary based on the task (Zhang et al., 2023). We compare three decoding strategies to identify the best method for AGEC task. As shown in Table 3, we compare *greedy decoding* (Germann, 2003) (temperature=0), *Beam search* (Freitag and Al-Onaizan, 2017) (num_beams=5, temperature=1), and *Top-P sampling* (Holtzman et al., 2019) (top-p=0.8, top-k=75, and temperature=0.8). With the highest scoring strategy identified, we scale up our data augmentation experiments, by generating sets of 5million and 10million reverseGold datasets. In addition to these datasets, we utilize the complete AGEC dataset from Solyman et al. (2021) (referred to as AraT5$_{v2}$ (11M) in our experiments) for further evaluation.

Outlined in Table 4, AraT5$_{v2}$ shows consistent improvement as the number of training samples increases from 5M to 11M. Results indicate Top-P sampling is the best decoding method for GEC, exhibiting a balance between number of correct edits and total number of edits made.

| Strategy | QALB-2014 | | | | QALB-2015 | | | |
|---|---|---|---|---|---|---|---|---|
| | P | R | $F_1$ | $F_{0.5}$ | P | R | $F_1$ | $F_{0.5}$ |
| **Greedy** | $74.09^{\pm0.57}$ | $65.63^{\pm0.59}$ | $69.60^{\pm0.54}$ | $72.23^{\pm0.55}$ | $67.41^{\pm0.82}$ | $66.85^{\pm0.97}$ | $67.13^{\pm0.82}$ | $67.30^{\pm0.80}$ |
| **Beam** | $75.47^{\pm1.11}$ | $68.61^{\pm1.26}$ | $71.87^{\pm1.19}$ | $73.99^{\pm1.14}$ | $70.54^{\pm0.44}$ | $68.04^{\pm0.14}$ | $69.27^{\pm0.24}$ | $70.03^{\pm0.35}$ |
| **Top-p** | $76.94^{\pm0.67}$ | $69.26^{\pm0.73}$ | $72.90^{\pm0.68}$ | $75.27^{\pm0.67}$ | $72.64^{\pm0.32}$ | $74.21^{\pm0.75}$ | $73.41^{\pm0.51}$ | $72.94^{\pm0.39}$ |

Table 3: Performance of AraT5$_{v2}$ (11M) on QALB-2014 and 2015 Test set under different decoding methods.

| Datasets | QALB-2014 | | | | QALB-2015 | | | |
|---|---|---|---|---|---|---|---|---|
| | P | R | $F_1$ | $F_{0.5}$ | P | R | $F_1$ | $F_{0.5}$ |
| **M1** | $71.35^{\pm0.14}$ | $64.45^{\pm0.41}$ | $67.73^{\pm0.17}$ | $69.85^{\pm0.04}$ | $69.65^{\pm0.57}$ | $64.74^{\pm0.57}$ | $67.11^{\pm0.14}$ | $68.61^{\pm0.33}$ |
| **M2** | $73.14^{\pm0.26}$ | $67.48^{\pm1.07}$ | $70.23^{\pm0.15}$ | $72.37^{\pm1.05}$ | $70.26^{\pm1.16}$ | $65.74^{\pm1.37}$ | $67.93^{\pm1.27}$ | $69.31^{\pm1.20}$ |
| **M3** | $\mathbf{76.94}^{\pm0.67}$ | $\mathbf{69.26}^{\pm0.73}$ | $\mathbf{72.90}^{\pm0.68}$ | $\mathbf{75.27}^{\pm0.67}$ | $\mathbf{72.64}^{\pm0.32}$ | $\mathbf{74.21}^{\pm0.75}$ | $\mathbf{73.41}^{\pm0.51}$ | $\mathbf{72.94}^{\pm0.39}$ |

Table 4: Performance of AraT5$_{v2}$ models using the 'Top-P' decoding method on QALB-2014 and 2015 Test sets, on different amounts of training data. M1 : AraT5$_{v2}$ (5M), M2 : AraT5$_{v2}$ (10M), M3 : AraT5$_{v2}$ (11M)

## 6 Sequence Tagging Approach

In this section, we detail our methods to adapt the GECToR model (Omelianchuk et al., 2020) to experiment with the seq2edit approach.

**Token-level transformations.** We first perform token-level transformations on the source to recover the target text. ***'Basic-transformations'*** are applied to perform the most common token-level edit operations, such as keeping the current token unchanged (`$KEEP`), deleting current token (`$DELETE`), appending new token t_1 next to the current token $x_i$ (`$APPEND_t1`) or replacing the current token $x_i$ with another token t_2 (`$REPLACE_t2`). To apply tokens with more task-specific operations, we employ ***'g-transformations'***

| Methods | Models | QALB-2014 | | | | QALB-2015 | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ |
| Seq2Seq | mT0 | 70.76 $^{\pm 0.03}$ | 50.78 $^{\pm 0.07}$ | 59.12 $^{\pm 0.05}$ | 65.59 $^{\pm 0.03}$ | 68.11 $^{\pm 0.20}$ | 59.68 $^{\pm 0.12}$ | 63.61 $^{\pm 0.15}$ | 66.23 $^{\pm 0.18}$ |
| | mT5 | 70.64 $^{\pm 0.12}$ | 50.16 $^{\pm 0.05}$ | 58.66 $^{\pm 0.05}$ | 65.30 $^{\pm 0.09}$ | 68.20 $^{\pm 0.10}$ | 59.02 $^{\pm 0.15}$ | 63.28 $^{\pm 0.04}$ | 66.14 $^{\pm 0.11}$ |
| | AraBART | 70.71 $^{\pm 0.06}$ | 60.46 $^{\pm 0.04}$ | 65.18 $^{\pm 0.07}$ | 68.39 $^{\pm 0.08}$ | 68.39 $^{\pm 0.09}$ | 67.95 $^{\pm 0.02}$ | 65.62 $^{\pm 0.05}$ | 66.76 $^{\pm 0.07}$ |
| | $AraT5_{v2}$ | 73.04 $^{\pm 0.10}$ | **63.09** $^{\pm 0.15}$ | **67.70** $^{\pm 0.12}$ | **70.81** $^{\pm 0.11}$ | 71.40 $^{\pm 0.90}$ | **72.83** $^{\pm 1.11}$ | **72.11** $^{\pm 0.99}$ | **71.68** $^{\pm 0.93}$ |
| Seq2edit | ARBERTv2 | 73.89 $^{\pm 0.35}$ | 48.33 $^{\pm 0.33}$ | 58.43 $^{\pm 0.35}$ | 66.82 $^{\pm 0.35}$ | 73.10 $^{\pm 0.29}$ | 55.40 $^{\pm 1.15}$ | 63.03 $^{\pm 0.86}$ | 68.70 $^{\pm 0.56}$ |
| | $ARBERT_{v2}^{\dagger}$ | **74.39** $^{\pm 0.22}$ | 47.62 $^{\pm 0.30}$ | 58.07 $^{\pm 0.29}$ | 66.87 $^{\pm 0.26}$ | **74.20** $^{\pm 0.28}$ | 53.80 $^{\pm 0.59}$ | 62.37 $^{\pm 0.49}$ | 68.96 $^{\pm 0.39}$ |
| | $MARBERT_{v2}$ | 73.53 $^{\pm 0.24}$ | 48.21 $^{\pm 0.39}$ | 58.24 $^{\pm 0.36}$ | 66.54 $^{\pm 0.30}$ | 72.90 $^{\pm 0.21}$ | 54.90 $^{\pm 0.52}$ | 62.63 $^{\pm 0.42}$ | 68.41 $^{\pm 0.31}$ |
| | $MARBERT_{v2}^{\dagger}$ | 74.21 $^{\pm 0.16}$ | 46.45 $^{\pm 0.25}$ | 57.14 $^{\pm 0.24}$ | 66.29 $^{\pm 0.20}$ | 74.00 $^{\pm 0.17}$ | 52.70 $^{\pm 0.34}$ | 61.56 $^{\pm 0.29}$ | 68.46 $^{\pm 0.23}$ |

Table 5: Performance of the seq2edit approach compared to baselines on the QALB-2014 and QALB-2015 Test sets. $^{\dagger}$: Models trained on 3-stage training.

such as the (`$MERGE`) tag to merge the current token and the next token into a single one. Edit space after applying token-level transformations results in `KEEP` (725K `op`), `$REPLACE_t2` (201K `op`), `$APPEND_t1` (75K `op`), `$DELETE` (13K `op`), and `$MERGE` (5.7K `op`) tags.

**Preprocessing and fine-tuning.** We start the preprocessing stage by aligning source tokens with target subsequences, preparing them for token-level transformations. We then fine-tune $ARBERT_{v2}$ (Elmadany et al., 2022) and $MARBERT_{v2}$ (Abdul-Mageed et al., 2021) on the preprocessed data. We adhere to the training approach detailed in the original paper (Omelianchuk et al., 2020), adopting its three-stage training and setting the iterative correction to three. More details about the fine-tuning procedure can be found in Appendix C.

**Sequence tagging evaluation.** As shown in Table 5, $ARBERT_{v2}$ and $MARBERT_{v2}$, exhibit high precision (e.g., $ARBERT_{v2}$'s three-step training is at 74.39 precision). However, relatively lower recall scores indicate challenges in ability of the two models to detect errors. Unlike the findings in the original paper, our implementation of a three-stage training approach yields mixed results: while accuracy improves, recall scores decrease, leading to a drop in the overall $F_1$ score (by 0.36 for $ARBERT_{v2}$ and 1.10 for $MARBERT_{v2}$, respectively). Consequently, all models fall behind the 'seq2seq' models. We note that both $ARBERT_{v2}$ and $MARBERT_{v2}$ surpass mT0 and mT5 in terms of $F_{0.5}$ scores, highlighting their abilities in correcting errors with precision.

## 7 Error Analysis

### 7.1 Error type evaluation.

We use the Automatic Error Type Annotation (ARETA) tool (Belkebir and Habash, 2021) to assess our models' performance on different error types. We focus on seven errors types: *Orthographic*, *Morphological*, *Syntactic*, *Semantic*, *Punctuation*, *Merge*, and *Split*. Examples of each error types alongside their translations can be found in Table 6. We examine top models from each approach, including $ARBERT_{v2}$ (3-step), GPT-4 (5-shot) + CoT, and $AraT5_{v2}$(11M). Figure 5 illustrates the performance of selected models under each error type. $AraT5_{v2}$(11M), surpasses all other models across all error categories. In particular, it excels in handling *Orthographic* (ORTH) errors, *Morphological* (MORPH) errors, and *Punctuation* (PUNCT) errors, consistently achieving over 65 $F_1$ score. However, it is worth observing that all models encounter challenges with *Semantic* (SEM) and *Syntactic* (SYN) errors. These disparate outcomes underscore the significance of selecting the appropriate model based on the error types prevalent in a specific dataset.

| Error Type | Incorrect Sentence | Correct Sentence |
|---|---|---|
| **Orthographic** | الرجل يرب الفرس . | الرجل يركب الفرس . |
| | *The man rears the horse.* | *The man rides the horse.* |
| **Punctuations** | الرجل ، يركب الفرس . | الرجل يركب الفرس . |
| | The man, rides the horse. | The man rides the horse. |
| **Syntax** | وجد رجلا يركب فرس . | وجد رجلا يركب فرسا . |
| | He found a man riding a hors. | He found a man riding a horse. |
| **Merge** | غداالرجل سيركب الفرس . | غدا الرجل سيركب الفرس . |
| | Tomorrowtheman will ride the horse. | Tomorrow the man will ride the horse. |
| **Splits** | غدا الرجل يرکب الفرس . | غدا الرجل يركب الفرس . |
| | The man ri des the horse. | The man rides the horse. |
| **Semantic** | الرجل يجلس في ظهر الفرس . | الرجل يجلس على ظهر الفرس . |
| | The man is sitting in the horse's back. | The man is sitting on the horse's back. |
| **Morphological** | غدا الرجل ركب الفرس . | غدا الرجل سيركب الفرس . |
| | Tomorrow the man rode the horse. | Tomorrow the man will ride the horse. |

Table 6: Examples of Merge, Morphological, Orthographic, Punctuation, Semantic, Split, and Syntactic errors, along with their corresponding corrections and English translations.

### 7.2 Normalization methods.

In addition to the *'Exact Match'* score, we also analyze system performance under different normalization methods. Namely, we assess the system on normalized text (1) without Alif/Ya errors, (2) without punctuation, and (3) free from both Alif/Ya and punctuation errors. Examples of text under each setting can be found in Appendix D.1.

### 7.3 Normalisation results

Looking at Table 7, in the 'No punctuation' setting, all models perform better, reflecting models' limitations in handling punctuation which is due to absence of clearly agreed upon punctuation rules in Arabic. Moreover, the datasets used are based on commentaries where punctuation is inherently inconsistent and varied. Another noteworthy obser-

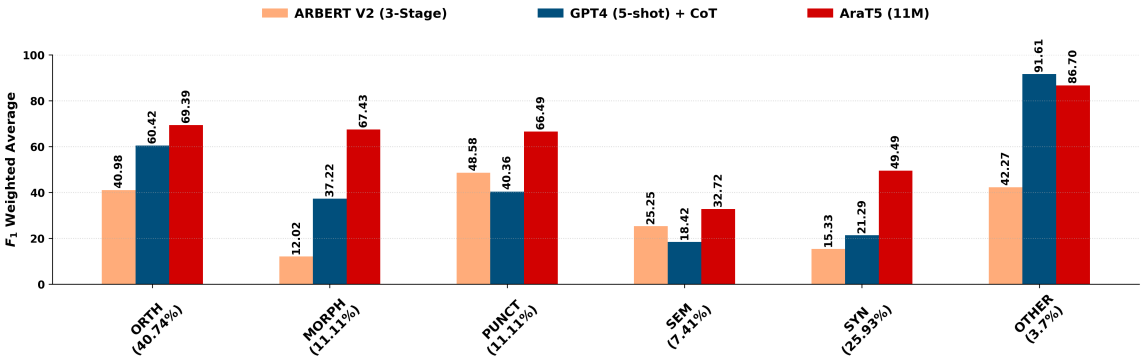


Figure 5: Best model $F_1$ scores for each approach on specific error types in the QALB-2014 Test set.

| Test Set | Models | Exact Match | | | | No Alif / Ya Errors | | | | No Punctuation | | | | No Punctuation and Alif / Ya Errors | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ |
| QALB-2014 | Solyman et al. (2021) | **79.06** | 65.79 | 71.82 | **75.99** | - | - | - | - | - | - | - | - | - | - | - | - |
| | Mohit et al. (2014) | 73.34 | 63.23 | 67.91 | 71.07 | **64.05** | 50.86 | 56.7 | **60.89** | 76.99 | 49.91 | 60.56 | 69.45 | 76.99 | 49.91 | 60.56 | 69.45 |
| | GPT4 (5-shot) | 69.46 | 61.96 | 65.49 | 67.82 | 58.44 | 51.47 | 54.73 | 56.90 | 74.59 | 78.15 | 76.33 | 75.28 | 60.06 | 65.75 | 62.78 | 61.12 |
| | ARBERT$_{v2}$ (3-step) | $74.17^{\pm 0.22}$ | $47.34^{\pm 0.30}$ | $57.79^{\pm 0.29}$ | $66.62^{\pm 0.26}$ | $64.90^{\pm 0.57}$ | $41.86^{\pm 0.24}$ | $50.89^{\pm 0.17}$ | $58.46^{\pm 0.33}$ | $76.90^{\pm 0.85}$ | $46.33^{\pm 0.58}$ | $57.83^{\pm 0.66}$ | $67.94^{\pm 0.75}$ | $56.66^{\pm 0.57}$ | $29.30^{\pm 0.61}$ | $38.62^{\pm 0.39}$ | $47.74^{\pm 0.03}$ |
| | AraT5$_{v2}$ (11m) | $76.94^{\pm 0.67}$ | $\mathbf{69.26}^{\pm 0.73}$ | $\mathbf{72.90}^{\pm 0.68}$ | $75.27^{\pm 0.67}$ | $62.42^{\pm 0.68}$ | $\mathbf{52.56}^{\pm 0.51}$ | $\mathbf{57.06}^{\pm 0.08}$ | $60.16^{\pm 0.38}$ | $\mathbf{86.52}^{\pm 0.50}$ | $\mathbf{82.90}^{\pm 0.17}$ | $\mathbf{84.67}^{\pm 0.25}$ | $\mathbf{85.77}^{\pm 0.39}$ | $\mathbf{79.44}^{\pm 0.51}$ | $\mathbf{67.40}^{\pm 0.53}$ | $\mathbf{72.92}^{\pm 0.52}$ | $\mathbf{76.70}^{\pm 0.52}$ |
| QALB-2015 | Solyman et al. (2021) | **80.23** | 63.59 | 70.91 | **76.24** | - | - | - | - | - | - | - | - | - | - | - | - |
| | Rozovskaya et al. (2015a) | 88.85 | 61.76 | 72.87 | 81.68 | **84.25** | 43.29 | **57.19** | **70.84** | **85.8** | **77.98** | **81.7** | **84.11** | **80.12** | **58.24** | **67.45** | **74.52** |
| | ChatGPT (3-shot) + EP | 52.33 | 47.57 | 49.83 | 54.10 | 37.93 | 39.97 | 38.92 | 32.95 | 53.38 | 56.63 | 54.96 | 54.00 | 33.33 | 46.77 | 38.92 | 35.36 |
| | ARBERT$_{v2}$ (3-step) | $73.92^{\pm 0.28}$ | $53.15^{\pm 0.59}$ | $61.84^{\pm 0.49}$ | $68.56^{\pm 0.39}$ | $57.14^{\pm 0.21}$ | $39.17^{\pm 0.76}$ | $46.47^{\pm 0.47}$ | $52.34^{\pm 0.13}$ | $66.90^{\pm 0.17}$ | $61.50^{\pm 0.50}$ | $64.09^{\pm 0.28}$ | $65.74^{\pm 0.18}$ | $71.18^{\pm 0.16}$ | $39.00^{\pm 0.87}$ | $50.39^{\pm 0.75}$ | $61.09^{\pm 0.49}$ |
| | AraT5$_{v2}$ (11m) | $72.10^{\pm 0.31}$ | $\mathbf{73.59}^{\pm 0.70}$ | $\mathbf{72.84}^{\pm 0.40}$ | $72.40^{\pm 0.30}$ | $55.80^{\pm 0.30}$ | $\mathbf{43.51}^{\pm 0.50}$ | $48.89^{\pm 0.22}$ | $52.81^{\pm 0.11}$ | $85.82^{\pm 0.31}$ | $72.85^{\pm 0.25}$ | $78.81^{\pm 0.28}$ | $82.87^{\pm 0.30}$ | $75.08^{\pm 0.13}$ | $53.30^{\pm 0.93}$ | $62.34^{\pm 0.60}$ | $69.40^{\pm 0.26}$ |

Table 7: Results on QALB-2014, QALB-2015 Test sets under Normalization Methods.

vation is the drop in $F_1$ scores when Alif/Ya errors are removed. This can be attributed to the fact that Alif/Ya errors are relatively simpler compared to other error categories. Moreover, AraT5$_{v2}$ is trained on formal texts such as AraNews (Nagoudi et al., 2020) and Hindawi Books [2], which contain proper Alif/Ya indicating the model's proficiency with the correct usage of these letters.

## 8 Discussion

**LLMs and ChatGPT.** ChatGPT demonstrates remarkable ability to outperform other fully trained models by learning from only a few examples, particularly five-shot under both few-shot CoT and EP prompting strategies. Nevertheless, ChatGPT's performance lags behind AraT5$_{v2}$ and AraBART, suggesting potential areas for improvements in prompting strategies to fully exploit ChatGPT models. Models such as Vicuna-13B as well as those trained on multilingual datasets like Bactrian-X$_{llama}$-7B and Bactrian-X$_{bloom}$-7B, tend to perform better. However, these models fail to match ChatGPT's performance which reinforces ChatGPT's superiority in this domain.

**Seq2seq approach.** Despite being smaller in size, pretrained Language Models (PLMs) often outperform LLMs, especially models specifically trained for Arabic tasks, such as AraT5$_{v2}$ and AraBART. In contrast, mT0 and mT5, both of which are multilingual models, are surpassed by ChatGPT when using both prompting strategies from 3-shot, but still outperform smaller LLMs such as LLaMA, Alpaca and Vicuna. Moreover, the results underscore the advantages of synthetic data for PLMs, as evidenced by the consistent improvement in scores with additional data.

**Seq2edit approach.** These models exhibit high precision scores and relatively low recall scores, suggesting their strengths in making corrections rather than detecting errors. This trend can be explained by the absence of *g-transformations*. For instance, in the case of English GECToR models, *g-transformations* enable a variety of changes, such as case alterations and grammatical transformations. However, in this work we only rely on the 'merge' *g-transformations* from the GECToR model as crafting effective *g-transformations* for Arabic, a language with rich morphological features, poses significant challenges, limiting the model's ability to effectively detect errors. Developing specific *g-transformations* for Arabic could significantly improve performance in these models.

**Data augmentation.** Data augmentation results underscore the potential of synthetic data, generated by ChatGPT, in enhancing model performance. Our findings reveal that not just the quantity, but the quality of synthetic data, is crucial for achieving optimal performance. The relative underperfor-

[2] www.hindawi.org/books

| Test Set | Models | Exact Match | | | |
|---|---|---|---|---|---|
| | | P | R | $F_{1.0}$ | $F_{0.5}$ |
| QALB-2014 | Solyman et al. (2021) | **79.06** | 65.79 | 71.82 | **75.99** |
| | Mohit et al. (2014) | 73.34 | 63.23 | 67.91 | 71.07 |
| | GPT4 (5-shot) | 69.46 | 61.96 | 65.49 | 67.82 |
| | ARBERT$_{v2}$ (3-step) | $74.17^{\pm 0.22}$ | $47.34^{\pm 0.30}$ | $57.79^{\pm 0.29}$ | $66.62^{\pm 0.26}$ |
| | AraT5$_{v2}$ (11m) | $76.94^{\pm 0.67}$ | $\mathbf{69.26}^{\pm 0.73}$ | $\mathbf{72.90}^{\pm 0.68}$ | $75.27^{\pm 0.67}$ |
| QALB-2015 | Solyman et al. (2021) | **80.23** | 63.59 | 70.91 | **76.24** |
| | Rozovskaya et al. (2015a) | 88.85 | 61.76 | 72.87 | 81.68 |
| | ChatGPT (3-shot) + EP | 52.33 | 47.57 | 49.83 | 54.10 |
| | ARBERT$_{v2}$ (3-step) | $73.92^{\pm 0.28}$ | $53.15^{\pm 0.59}$ | $61.84^{\pm 0.49}$ | $68.56^{\pm 0.39}$ |
| | AraT5$_{v2}$ (11m) | $72.10^{\pm 0.31}$ | $\mathbf{73.59}^{\pm 0.70}$ | $\mathbf{72.84}^{\pm 0.40}$ | $72.40^{\pm 0.30}$ |

Table 8: Results on QALB-2014, QALB-2015 Test sets compared to recent works.

mance of models further trained with synthetically generated data examples emphasizes this conclusion. Improvements we observe when expanding the dataset from 5M to 10M and from 10M to 11M are similar, even though the quantity of additional data vary. This can be attributed to the quality of the sources as the data for 5M and 10M were derived from noisier online commentaries, while the 11M data was derived from the OSIAN corpus (Zeroual et al., 2019). Furthermore, our results on decoding methods on scaled datasets indicate that the chosen method can significantly influence precision and recall, emphasizing the need to select the right method depending on the specific task at hand.

**Best model in comparison.** Although our main objective is not to develop the best model for AGEC, our AraT5$_{v2}$ (11M) model as detailed in Table 8 excels in comparison to previous SOTA (71.82 vs. 72.90). It is worth noting that contemporaneous work by Alhafni et al. (2023) introduces a new alignment algorithm that is much better than that employed by the shared task evaluation code we use. They also present an AGEC model. In personal communication with the authors, they confirmed their alignment algorithm through which we can perform direct and fair comparisons, and the data split on ZAEBUC dataset (Habash and Palfreyman, 2022) will be released once their work is published through peer-review. Different from their work, our models are also dependency-free. For example, we do not exploit any morphological analyzers.

## 9 Conclusion

This paper provided a detailed exploration of the potential of LLMs, with a particular emphasis on ChatGPT for AGEC. Our study highlights ChatGPT's promising capabilities, in low-resource scenarios, as evidenced by its competitive performance on few-shot setttings. However, AraT5$_{v2}$ and AraBART still exhibit superior results across various settings and error types. Our findings also emphasize the role of high-quality synthetic data, reinforcing that both quantity and quality matter in achieving optimal performance. Moreover, our work unveils trade-offs between precision and recall in relation to dataset size and throughout all the other experimental settings. These insight, again, could inform future strategies for improving GEC systems. Although our exploration of ChatGPT's performance on AGEC tasks showcases encouraging results, it also uncovers areas ripe for further study. Notably, there remains significant room for improvement in GEC systems, particularly within the context of low-resource languages. Future research may include refining prompting strategies, enhancing synthetic data generation techniques, and addressing the complexities and rich morphological features inherent in the Arabic language.

## 10 Limitations

We identify the following limitations in this work:

1. This work is primarily focused on MSA and does not delve into dialectal Arabic (DA) or the classical variety of Arabic (CA). While there exist DA resources such as the MADAR corpus (Bouamor et al., 2018), their primary application is for dialect identification (DID) and machine translation (MT), making them unsuitable for our specific AGEC objectives. A more comprehensive coverage could be achieved with the development and introduction of datasets specifically tailored for the dialects in question.

2. This work aimed to examine the potential of LLMs, with an emphasis on ChatGPT, by comparing them to fully pretrained models. However, uncertainty surrounding the extent of Arabic data on which ChatGPT has been trained, poses challenges for direct comparisons with other pretrained models. Additionally, LLMs are primarily fine-tuned for English-language data. While prior studies have demonstrated their effectiveness in other languages, the limited amount of pretraining data for non-English languages complicates a straightforward comparison.

3. The scope of this work is primarily centered on sentence-level GEC. This limitation arose due to the official ChatGPT API, at the time

of our study, allowed a maximum of 4,097 tokens, making it unsuitable for longer texts and precluding document-level GEC tasks. However, it's worth noting that document-level correction, offers a broader context that's vital for addressing certain grammatical inconsistencies and errors (Yuan and Bryant, 2021). With the recent introduction of a newer API that accommodates extended texts, future endeavors can potentially address document-level GEC, utilizing datasets such as QALB-2015 L2 and the newly introduced ZAEBUC corpus.

## 11 Ethics Statement and Broad Impact

**Encouraging research development and contributing to a collaborative research culture.** Progress in AGEC has been stagnant for a long time due to the lack of benchmark datasets. This can be attributed to the extensive time and cost involved in creating these datasets. As a result, advancing AGEC has proven challenging. With the recent development of LLMs and their capabilities, there is potential for these models to expedite the creation of datasets. By doing so, they can significantly reduce both time and cost, as has been observed in other languages. We hope our work will inspire further exploration into the capabilities of LLMs for AGEC, thus aiding in the progress of this field.

**Advancing Second Language Learning through LLMs.** With increasing interest in second language learning, ensuring accuracy and effectiveness of written language has become significant for pedagogical tools. Nowadays, individuals treat LLMs as their own writing assistants. Therefore, LLMs in the context of educational applications and more specifically GEC is becoming increasingly important. As such, introducing works in the development of tools that aid assistance in writing can help bridge the gap between non-native speakers and fluent written communication, enhancing the efficacy of educational tools. Especially with Arabic, being a collection of a diverse array of languages and dialectal varieties, we hope this will inspire more work to ensure comprehensive coverage and improved support for all learners. However, it is crucial to emphasize the ethical implications of using AI-driven educational tools. It's essential that these tools remain unbiased, transparent, and considerate of individual learning differences, ensuring the trustworthiness and integrity of educational platforms for every learner.

**Data privacy.** In relation to the data used in this work, all datasets are publicly available. Therefore, we do not have privacy concerns.

## Acknowledgments

We acknowledge support from Canada Research Chairs (CRC), the Natural Sciences and Engineering Research Council of Canada (NSERC; RGPIN-2018-04267), the Social Sciences and Humanities Research Council of Canada (SSHRC; 435-2018-0576; 895-2020-1004; 895-2021-1008), Canadian Foundation for Innovation (CFI; 37771), Digital Research Alliance of Canada,[3] and UBC Advanced Research Computing-Sockeye.[4]

## References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Muhammad Abdul-Mageed, Chiyu Zhang, Houda Bouamor, and Nizar Habash. 2020. NADI 2020: The first nuanced Arabic dialect identification shared task. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 97–110, Barcelona, Spain (Online). Association for Computational Linguistics.

Abdullah Alfaifi and Eric Atwell. 2012. Arabic learner corpora (alc): A taxonomy of coding errors. In *The 8th International Computing Conference in Arabic*.

Bashar Alhafni, Go Inoue, Christian Khairallah, and Nizar Habash. 2023. Advancements in arabic grammatical error detection and correction: An empirical investigation. *arXiv preprint arXiv:2305.14734*.

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4260–4270, Hong Kong, China. Association for Computational Linguistics.

Riadh Belkebir and Nizar Habash. 2021. Automatic error type annotation for arabic. *arXiv preprint arXiv:2109.08068*.

[3] https://alliancecan.ca

[4] https://arc.ubc.ca/ubc-arc-sockeye

Houda Bouamor, Nizar Habash, Mohammad Salameh, Wajdi Zaghouani, Owen Rambow, Dana Abdulrahim, Ossama Obeid, Salam Khalifa, Fadhl Eryani, Alexander Erdmann, and Kemal Oflazer. 2018. The MADAR Arabic dialect corpus and lexicon. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe. 2022. Grammatical error correction: A survey of the state of the art. *arXiv preprint arXiv:2211.05166*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572, Montréal, Canada. Association for Computational Linguistics.

Moussa Kamal Eddine, Nadi Tomeh, Nizar Habash, Joseph Le Roux, and Michalis Vazirgiannis. 2022. Arabart: a pretrained arabic sequence-to-sequence model for abstractive summarization. *arXiv preprint arXiv:2203.10945*.

AbdelRahim Elmadany, El Moatez Billah Nagoudi, and Muhammad Abdul-Mageed. 2022. Orca: A challenging benchmark for arabic language understanding. *arXiv preprint arXiv:2212.10758*.

Tao Fang, Shu Yang, Kaixin Lan, Derek F Wong, Jinpeng Hu, Lidia S Chao, and Yue Zhang. 2023. Is chatgpt a highly fluent grammatical error correction system? a comprehensive evaluation. *arXiv preprint arXiv:2304.01746*.

Mariano Felice, Zheng Yuan, Øistein E. Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 15–24, Baltimore, Maryland. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Simon Flachs, Felix Stahlberg, and Shankar Kumar. 2021. Data strategies for low-resource grammatical error correction. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 117–122, Online. Association for Computational Linguistics.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. *arXiv preprint arXiv:1702.01806*.

Ulrich Germann. 2003. Greedy decoding for statistical machine translation in almost linear time. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 72–79.

Peiyuan Gong, Xuebo Liu, Heyan Huang, and Min Zhang. 2022. Revisiting grammatical error correction evaluation and beyond. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6891–6902, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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, Florence, Italy. Association for Computational Linguistics.

Nizar Habash and David Palfreyman. 2022. ZAEBUC: An annotated Arabic-English bilingual writer corpus. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 79–88, Marseille, France. European Language Resources Association.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Serena Jeblee, Houda Bouamor, Wajdi Zaghouani, and Kemal Oflazer. 2014. CMUQ@QALB-2014: An SMT-based system for automatic Arabic error correction. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 137–142, Doha, Qatar. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018a. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 595–606, New Orleans, Louisiana. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018b. Approaching neural grammatical error correction as a low-resource machine translation task. *arXiv preprint arXiv:1804.05940*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Haonan Li, Fajri Koto, Minghao Wu, Alham Fikri Aji, and Timothy Baldwin. 2023. Bactrian-x: A multilingual replicable instruction-following model. `https://github.com/MBZUAI-nlp/Bactrian-X`.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. *arXiv preprint arXiv:1909.01187*.

Behrang Mohit, Alla Rozovskaya, Nizar Habash, Wajdi Zaghouani, and Ossama Obeid. 2014. The first qalb shared task on automatic text correction for arabic. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 39–47.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2022. Crosslingual generalization through multitask finetuning.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, and Muhammad Abdul-Mageed. 2022. AraT5: Text-to-text transformers for Arabic language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 628–647, Dublin, Ireland. Association for Computational Linguistics.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, Muhammad Abdul-Mageed, Tariq Alhindi, and Hasan Cavusoglu. 2020. Machine generation and detection of arabic manipulated and fake news. *arXiv preprint arXiv:2011.03092*.

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, Baltimore, Maryland. Association for Computational Linguistics.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: tag, not rewrite. *arXiv preprint arXiv:2005.12592*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 702–707, Online. Association for Computational Linguistics.

Alla Rozovskaya, Houda Bouamor, Nizar Habash, Wajdi Zaghouani, Ossama Obeid, and Behrang Mohit. 2015a. The second qalb shared task on automatic text correction for arabic. In *Proceedings of the Second workshop on Arabic natural language processing*, pages 26–35.

Alla Rozovskaya, Houda Bouamor, Nizar Habash, Wajdi Zaghouani, Ossama Obeid, and Behrang Mohit. 2015b. The second QALB shared task on automatic text correction for Arabic. In *Proceedings of the Second Workshop on Arabic Natural Language Processing*, pages 26–35, Beijing, China. Association for Computational Linguistics.

Alla Rozovskaya, Nizar Habash, Ramy Eskander, Noura Farra, and Wael Salloum. 2014. The Columbia system in the QALB-2014 shared task on Arabic error correction. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 160–164, Doha, Qatar. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Aiman Solyman, Zhenyu Wang, Qian Tao, Arafat Abdulgader Mohammed Elhag, Rui Zhang, and Zeinab Mahmoud. 2022. Automatic arabic grammatical error correction based on expectation-maximization routing and target-bidirectional agreement. *Knowledge-Based Systems*, 241:108180.

Aiman Solyman, Marco Zappatore, Wang Zhenyu, Zeinab Mahmoud, Ali Alfatemi, Ashraf Osman Ibrahim, and Lubna Abdelkareim Gabralla. 2023. Optimizing the impact of data augmentation for low-resource grammatical error correction. *Journal of King Saud University - Computer and Information Sciences*, 35(6):101572.

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.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Maksym Tarnavskyi, Artem Chernodub, and Kostiantyn Omelianchuk. 2022. Ensembling and knowledge distilling of large sequence taggers for grammatical error correction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3842–3852, Dublin, Ireland. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Daniel Watson, Nasser Zalmout, and Nizar Habash. 2018. Utilizing character and word embeddings for text normalization with sequence-to-sequence models. *arXiv preprint arXiv:1809.01534*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Haoran Wu, Wenxuan Wang, Yuxuan Wan, Wenxiang Jiao, and Michael Lyu. 2023. Chatgpt or grammarly? evaluating chatgpt on grammatical error correction benchmark. *arXiv preprint arXiv:2303.13648*.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 619–628, New Orleans, Louisiana. Association for Computational Linguistics.

Benfeng Xu, An Yang, Junyang Lin, Quan Wang, Chang Zhou, Yongdong Zhang, and Zhendong Mao. 2023. Expertprompting: Instructing large language models to be distinguished experts. *arXiv preprint arXiv:2305.14688*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Zheng Yuan and Christopher Bryant. 2021. Document-level grammatical error correction. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 75–84, Online. Association for Computational Linguistics.

Wajdi Zaghouani, Behrang Mohit, Nizar Habash, Ossama Obeid, Nadi Tomeh, Alla Rozovskaya, Noura Farra, Sarah Alkuhlani, and Kemal Oflazer. 2014. Large scale Arabic error annotation: Guidelines and framework. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Imad Zeroual, Dirk Goldhahn, Thomas Eckart, and Abdelhak Lakhouaja. 2019. OSIAN: Open source international Arabic news corpus - preparation and integration into the CLARIN-infrastructure. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 175–182, Florence, Italy. Association for Computational Linguistics.

Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. 2023. Llama-adapter: Efficient fine-tuning of language models with zero-init attention.

## A Related Works

**Sequence to sequence approach.** Transformer-based Language Models (LMs) have been integral to advancements in GEC. These models have substantially transformed the perception of GEC, reframing it as a MT task. In this framework, erroneous sentences are considered as the source language, and the corrected versions as the target language. This perspective, which has led to SOTA results in the CONLL 2013 and 2014 shared tasks (Bryant et al., 2022; Ng et al., 2013, 2014), reinterprets GEC as a low-resource or mid-resource MT task. Building on this paradigm, Junczys-Dowmunt et al. (2018a) successfully adopted techniques from low-resource NMT and Statistical Machine Translation (SMT)-based GEC methods, leading to considerable improvements on both the CONLL and JFLEG datasets.

**Sequence tagging approach.** Sequence tagging methods, another successful route to GEC, are showcased by models like GECToR (Omelianchuk et al., 2020), LaserTagger (Malmi et al., 2019), and the Parallel Iterative Edit (PIE) model (Awasthi et al., 2019). By viewing GEC as a text editing task, these models make edits predictions instead of tokens, label sequences rather than generating them, and iteratively refine predictions to tackle dependencies. Employing a limited set of output tags, these models apply edit operations on the input sequence, reconstructing the output. This technique not only capably mirrors a significant chunk of the target training data, but it also diminishes the vocabulary size and establishes the output length as the source text's word count. Consequently, it curtails the number of training examples necessary for model accuracy, which is particularly beneficial in settings with sparse human-labeled data (Awasthi et al., 2019).

**Instruction fine-tuning.** LLMs have revolutionized NLP, their vast data-learning capability enabling diverse task generalizations. Key to their enhancement has been instructional finetuning, which fortifies the model's directive response and mitigates the need for few-shot examples (Ouyang et al., 2022; Wei et al., 2022b; Sanh et al., 2021). A novel approach, Chain of Thought (CoT), directs LLMs through a series of natural language reasoning, generating superior outputs. Proven beneficial in 'Let's think step by step' prompts (Wei et al., 2022b), CoT has harnessed LLMs for multi-task cognitive tasks (Kojima et al., 2022) and achieved SOTA results in complex system-2 tasks like arithmetic and symbolic reasoning.

**ChatGPT.** In the specific realm of GEC, LLMs have demonstrated its potential. Fang et al. (2023) applied zero-shot and few-shot CoT settings using in-context learning for ChatGPT (Brown et al., 2020) and evaluated its performance on three document-level English GEC test sets. Similarly, Wu et al. (2023) carried out an empirical study to assess the effectiveness of ChatGPT in GEC, in the CoNLL2014 benchmark dataset.

**Development in AGEC** Arabic consists of a collection of diverse languages and dialectal varieties with Modern Standard Arabic (MSA) being the current standard variety used in government and pan-arab media as well as education (Abdul-Mageed et al., 2020). The inherent ambiguity of Arabic at the orthographic, morphological, syntactic, and semantic levels makes AGEC particularly challenging. Optional use of diacritics further introduces orthographic ambiguity (Belkebir and Habash, 2021), making AGEC even harder.

Despite these hurdles, progress has been made in AGEC. For dataset development, the QALB corpus (Zaghouani et al., 2014) was utilized. During the QALB-2014 and 2015 shared tasks (Mohit et al., 2014; Rozovskaya et al., 2015b), the first AGEC datasets containing comments and documents from both native (L1) and Arabic learner (L2) speakers were released. Furthermore, the more recent ZAEBUC corpus (Habash and Palfreyman, 2022), which features essays from first-year university students at Zayed University in the UAE, has also been released. There has also been work on generating synthetic data. Solyman et al. (2021, 2023) apply Convolutional neural network (CNN) to generate synthetic parallel data using unsupervised noise injection techniques showing improvements in the QALB-2014 and 2015 benchmark datasets. In terms of model development, Watson et al. (2018) developed a character-level seq2seq model that achieved notable results on AGEC L1 data, marking prgoress from basic classifier models (Rozovskaya et al., 2014) and statistical machine translation models (Jeblee et al., 2014). More recently, Solyman et al. (2022, 2021) introduced novel design that incorporates dynamic linear combinations and the EM routing algorithm within a seq2seq Transformer framework.

## B Instruction Fine-tuning LLMs

### B.1 Instructions for LLMs

Instruction format used for training is provided in Table 9 and instructions used for training are shown in Table 10.

### B.2 Baseline and experimental setup for LLMs and ChatGPT

For LLMs, evaluation was only done on the QALB-2014 Test set, for two main reasons. First was due to the high cost in producing results using ChatGPT and we were able to observation of a similar trend in our preliminary experiment with ChatGPT-3.5 Turbo on the QALB-2015. Second, as instruction fine-tuned were predominantly compared against ChatGPT's performance, we also evaluate them only on the QALB-2014 Test set. These Results can be found in Table 11.

## C Sequence Tagging Approach

The training procedure detailed in the original GECToR paper (Omelianchuk et al., 2020) encompasses three stages:

1. Pre-training on synthetically generated sentences with errors.

2. Fine-tuning solely on sentences that contain errors.

3. Further fine-tuning on a mix of sentences, both with and without errors.

For our training process, we pre-train the model on the complete AGEC dataset (Solyman et al., 2021), use the reverseGold dataset for stage 2, and employ the gold training data in the third stage. Moreover, as some corrections in a sentence depend on others, applying edit sequences once may not be enough to correct the sentence fully. To address this issue, GECToR employs an iterative correction approach from Awasthi et al. (2019). However, in our experiments, we find that the iterative correction approach does not result in any tangible improvement. Therefore, we set our iterations to 3.

## D Normalization Methods

### D.1 Normalization examples

Examples of text under each normalization methods can be found in Table 12

### D.2 Arabic Learner Corpus error type taxonomy

The ALC error type taxonomy can be found in Table 13.

### D.3 Hyperparameters

The Hyperparameters used for training are shown in Table 14.

### D.4 Dev results

Results on the Dev set are presented in Table 15.

### D.5 ARETA results

Full results evaluated using ARETA are presented in Table 16.

| Fine-tune Instruction Example |
|---|
| فيما يلي أمر توجيه يصف مهمة مرتبطة بمدخل لتزويد النص بسياق اضافي. يرجى صياغة ردود مناسبة لتحقق الطلب بطريقة مناسبة و دقيقة.<br>### الأمر التوجيه :<br>قم بتصحيح كل الأخطاء الكتابية في النص التالي:<br>### المدخل :<br>الرجل يرب الفرس .<br>### الرد :<br>الرجل يركب الفرس . |

Table 9: Modified data format for the LLaMA instruction fine-tuning step.

| Translated in English | Instructions Samples |
|---|---|
| Correct all written errors in the following text except for a thousand, ya and punctuation: | قم بتصحيح كل الأخطاء الكتابية في النص التالي ماعدا المتعلقة بالألف والياء وعلامات الترقيم : |
| Please verify spelling, grammatical scrutiny, and correct all errors in the following sentence, except for punctuation: | الرجاء التدقيق الإملائي والتدقيق النحوي و تصحيح كل الأخطاء في الجملة التالية إلا الخاصة بعلامات الترقيم : |
| Explore the grammatical errors and repair them except for punctuation marks such as a comma, or a question marks, etc: | قم بإستكشاف أخطاء التدقيق الإملائي وإصلاحها ماعدا المتعلقة بعلامات الترقيم كالفاصلة أو علامة إستفهام ، إلخ: |
| Can you correct all errors in the following text except those related to punctuation such as commas, periods, etc: | هل تمكنك كل الأخطاء الموجودة في النص التالي ماعدا المتعلقة بعلامات الترقيم كالفاصلة ، النقطة ، إلخ : |
| Can you fix all spelling and grammatical errors, except for the mistakes of the "Alif" and "Ya": | هل تمكنك إصلاح كل الأخطاء الإملائية والنحوية ماعدا الأخطاء الخاصة بالألف والياء: |
| Please explore the grammatical spelling errors and repair them all, except for the mistakes related to the "Alif" and "Ya" | الرجاء إستكشاف أخطاء التدقيق الإملائي النحوي وإصلاحها كلها ماعدا الأخطاء المتعلقة بالألف والياء: |
| Correct all the written errors in the following text except for the "Alif" and "Ya": | قم بتصحيح كل الأخطاء الكتابية في النص التالي ماعدا المتعلقة بالألف والياء: |
| Please correct all errors in the following sentence: | الرجاء تصحيح كل الأخطاء الموجودة في الجملة التالية: |

Table 10: Different instructions used for instruction fine-tuning.

| Settings | Models | Exact Match | | | |
|---|---|---|---|---|---|
| | | P | R | $F_{1.0}$ | $F_{0.5}$ |
| + CoT | ChatGPT (3-shot) | 49.89 | 46.72 | 48.22 | 49.49 |
| | ChatGPT (5-shot) | 52.33 | 47.57 | 49.83 | 51.15 |

Table 11: Performance of ChatGPT-3.5 on QALB-2015 Test set.

| Normalisation Method | Example |
|---|---|
| **Normal** | نحن معشر العرب نعرف إلا الشماتة ، ولكن يجب أن ندرس هذه الحالة ونحن المخرج منها من الاقتصاد الإسلامي. |
| **No Alif/Ya** | نحن معشر العرب نعرف الا الشماتة ، ولكن يجب ان ندرس هذه الحالة ونحن المخرج منها من الاقتصاد الاسلامي. |
| **No Punct** | نحن معشر العرب نعرف إلا الشماتة ولكن يجب أن ندرس هذه الحالة ونحن المخرج منها من الاقتصاد الإسلامي |
| **No Alif/Ya & Punct** | نحن معشر العرب نعرف الا الشماتة ولكن يجب ان ندرس هذه الحالة ونحن المخرج منها من الاقتصاد الاسلامي |

Table 12: Examples of normalized text: with Alif/Ya errors removed, punctuation removed, and both Alif/Ya errors and punctuation removed.

| Class | Sub-class | Description |
|---|---|---|
| **Orthographic** | **OH** | **Hamza error** |
| | **OT** | Confusion in Ha and Ta Mutadarrifatin |
| | **OA** | **Confusuion in Alif and Ya Mutadarrifatin** |
| | **OW** | Confusion in Alif Fariqa |
| | **ON** | Confusion Between Nun and Tanwin |
| | **OS** | Shortening the long vowels |
| | **OG** | Lengthening the short vowels |
| | **OC** | Wrong order of word characters |
| | **OR** | Replacement in word character(s) |
| | **OD** | Additional character(s) |
| | **OM** | Missing character(s) |
| | **OO** | Other orthographic errors |
| **Morphological** | **MI** | Word inflection |
| | **MT** | Verb tense |
| | **MO** | Other morphological errors |
| | **XF** | Definiteness |
| | **XG** | Gender |
| | **XN** | Number |
| | **XT** | Unnecessary word |
| | **XM** | Missing word |
| | **XO** | Other syntactic errors |
| **Semantic** | **SW** | Word selection error |
| | **SF** | Fasl wa wasl (confusion in conjunction use/non-use) |
| | **SO** | Other semantic errors |
| **Punctuation** | **PC** | Punctuation confusion |
| | **PT** | Unnecessary punctuation |
| | **PM** | Missing punctuation |
| | **PO** | Other errors in punctuation |
| **Merge** | **MG** | Words are merged |
| **Split** | **SP** | Words are split |

Table 13: The ALC error type taxonomy extended with merge and split classes

| Hyperparameter | Seq2seq | Decoder Only (LLMs) | Seq2Edit Encoder Only |
|---|---|---|---|
| Learning Rate | $5 \times 10^{-5}$ | $2 \times 10^{-5}$ | $1 \times 10^{-5}$ |
| Train Batch Size | 4 | 8 | 8 |
| Eval Batch Size | 4 | 8 | 8 |
| Seed | 42 | 42 | 42 |
| Gradient Accumulation Steps | 8 | 8 | 8 |
| Total Train Batch Size | 32 | 64 | 64 |
| Optimizer | Adam (betas=(0.9,0.999), epsilon=$1 \times 10^{-8}$) | AdamW (betas=(0.9,0.999), epsilon=$1 \times 10^{-7}$) | AdamW (betas=(0.9,0.999), epsilon=$1 \times 10^{-8}$) |
| LR Scheduler Type | Cosine | Linear | Cosine |
| Num Epochs | 50 | 4 | 100 |

Table 14: Summary of hyperparameters used for model training.

| Settings | Models | Exact Match | | | | No Alif / Ya Errors | | | | No Punctuation | | | | No Puncation and Alif / Ya Errors | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ | P | R | $F_{1.0}$ | $F_{0.5}$ |
| **Seq2Edit** | ARBERTv2 | 73.30 | 47.85 | 57.90 | 66.25 | 65.60 | 44.20 | 52.81 | 59.81 | 72.38 | 48.75 | 58.26 | 65.98 | 57.40 | 33.90 | 42.63 | 50.41 |
| | $ARBERT_{v2}$ 3-stage | 74.65 | 46.70 | 57.46 | 66.67 | 65.00 | 41.20 | 50.43 | 58.27 | 75.50 | 44.50 | 56.00 | 66.27 | 55.70 | 27.50 | 36.82 | 46.22 |
| | $MARBERT_{v2}$ | 72.95 | 47.65 | 57.65 | 65.95 | 64.60 | 43.20 | 51.78 | 58.78 | 73.72 | 44.16 | 55.23 | 65.02 | 56.80 | 34.20 | 42.69 | 50.17 |
| | $MARBERT_{v2}$ 3-stage | 74.55 | 45.75 | 56.70 | 66.21 | 65.10 | 41.30 | 50.54 | 58.37 | 75.41 | 45.52 | 56.77 | 66.66 | 56.00 | 29.20 | 38.38 | 47.31 |
| **LLMs** | LLama-7B | 58.20 | 32.50 | 41.71 | 50.25 | 35.50 | 16.70 | 22.71 | 28.98 | 19.60 | 54.30 | 28.80 | 22.47 | 65.10 | 32.00 | 42.91 | 53.94 |
| | Alpaca-7B | 42.20 | 31.20 | 35.88 | 39.42 | 42.20 | 33.40 | 37.29 | 40.09 | 82.20 | 62.20 | 70.81 | 77.23 | 62.20 | 39.50 | 48.32 | 55.79 |
| | Vicuna-13B | 63.90 | 51.00 | 56.73 | 60.82 | 51.40 | 39.30 | 44.54 | 48.42 | 83.90 | 73.90 | 78.58 | 81.69 | 68.50 | 49.00 | 57.13 | 63.45 |
| | Bactrian-$X_{bloom}$-7B | 60.80 | 43.80 | 50.92 | 56.42 | 53.70 | 41.00 | 46.50 | 50.57 | 79.40 | 63.00 | 70.26 | 75.47 | 62.00 | 51.00 | 55.96 | 59.44 |
| | Bactrian-$X_{llama}$-7B | 58.60 | 41.40 | 48.52 | 54.10 | 51.00 | 38.10 | 43.62 | 47.77 | 77.00 | 59.20 | 66.94 | 72.63 | 58.60 | 48.10 | 52.83 | 56.15 |
| **Seq2Seq** | mT0 | 69.35 | 54.29 | 60.90 | 65.70 | 57.45 | 42.50 | 48.86 | 53.67 | 82.35 | 75.34 | 78.69 | 80.85 | 70.20 | 50.30 | 58.61 | 65.05 |
| | mT5 | 69.00 | 53.20 | 60.08 | 65.13 | 56.70 | 39.50 | 46.56 | 52.16 | 81.00 | 70.00 | 75.10 | 78.53 | 68.00 | 48.00 | 56.28 | 62.77 |
| | AraBART | 72.00 | 61.50 | 66.34 | 69.62 | 60.00 | 49.70 | 54.37 | 57.61 | 85.00 | 78.50 | 81.62 | 83.62 | 74.00 | 60.50 | 66.57 | 70.84 |
| | $AraT5_{v2}$ | 74.50 | 64.50 | 69.14 | 72.26 | 63.50 | 52.70 | 57.60 | 61.00 | 88.00 | 84.50 | 86.21 | 87.28 | 81.50 | 69.50 | 75.02 | 78.78 |
| | $AraT5_{v2}$ (5M) | 75.33 | 67.44 | 71.17 | 73.61 | 64.55 | 51.55 | 57.32 | 61.45 | 89.22 | 83.40 | 86.21 | 87.99 | 81.30 | 70.24 | 75.37 | 78.82 |
| | $AraT5_{v2}$ (10M) | 75.90 | 68.33 | 71.92 | 74.25 | 65.34 | 52.44 | 58.18 | 62.28 | 89.88 | 84.22 | 86.96 | 88.69 | 82.34 | 71.44 | 76.50 | 79.90 |
| | $AraT5_{v2}$ (11M) | 77.85 | 68.90 | 73.10 | 75.88 | 66.33 | 55.20 | 60.26 | 63.76 | 90.10 | 85.21 | 87.59 | 89.08 | 84.55 | 71.50 | 77.48 | 81.57 |

Table 15: Dev Set results on the QALB-2014 benchmark dataset.

| CLASS | GECToR_ARBERT | five-shot_2014_expertprompt | five-shot_2014-chatgpt4 | AraT5 (11M) | COUNT |
|---|---|---|---|---|---|
| OH | 73.73 | 89.80 | 92.91 | 87.34 | 4902 |
| OT | 76.59 | 94.12 | 95.58 | 90.84 | 708 |
| OA | 78.63 | 84.66 | 88.93 | 87.35 | 275 |
| OW | 38.57 | 80.79 | 86.96 | 83.70 | 107 |
| ON | 0.00 | 0.00 | 0.00 | 0.00 | 0 |
| OG | 48.00 | 55.74 | 63.64 | 90.32 | 34 |
| OC | 21.43 | 28.57 | 53.66 | 87.18 | 22 |
| OR | 38.24 | 53.02 | 65.96 | 77.10 | 528 |
| OD | 33.76 | 51.89 | 59.60 | 73.07 | 321 |
| OM | 41.80 | 44.53 | 57.35 | 86.44 | 393 |
| OO | 0.00 | 0.00 | 0.00 | 0.00 | 0 |
| MI | 11.02 | 13.25 | 20.53 | 75.00 | 83 |
| MT | 0.00 | 7.84 | 11.43 | 62.50 | 7 |
| XC | 32.95 | 46.10 | 50.78 | 88.35 | 526 |
| XF | 6.06 | 17.98 | 23.81 | 76.92 | 29 |
| XG | 37.10 | 19.57 | 31.35 | 89.47 | 79 |
| XN | 25.19 | 25.79 | 31.25 | 88.12 | 108 |
| XT | 3.95 | 3.78 | 5.48 | 2.48 | 66 |
| XM | 2.04 | 4.14 | 6.38 | 1.07 | 26 |
| XO | 0.00 | 0.00 | 0.00 | 0.00 | 0 |
| SW | 50.51 | 21.25 | 33.38 | 8.29 | 219 |
| SF | 0.00 | 6.67 | 3.45 | 57.14 | 3 |
| PC | 60.89 | 56.25 | 47.59 | 74.98 | 713 |
| PT | 29.62 | 29.58 | 21.40 | 57.42 | 480 |
| PM | 55.24 | 54.21 | 52.09 | 67.08 | 5599 |
| MG | 25.05 | 75.96 | 79.70 | 64.80 | 434 |
| SP | 42.27 | 90.93 | 91.61 | 86.70 | 805 |
| **micro avg** | 55.67 | 60.05 | 64.51 | 57.28 | 16467 |
| **macro avg** | 30.84 | 39.13 | 43.51 | 61.62 | 16467 |
| **weighted avg** | 56.98 | 66.96 | 68.24 | 76.35 | 16467 |

Table 16: Analysis of Error Type performances on the QALB-2014 Test set.