Prompting open-source and commercial language models for grammatical error correction of English learner text

Christopher Davis 🍇 **Andrew Caines** 🍇 **Øistein Andersen** 🍇
Shiva Taslimipoor 🍇 **Helen Yannakoudakis** 🌺 **Zheng Yuan** 🌺
Christopher Bryant 🌻 **Marek Rei** 🌴 **Paula Buttery** 🍇
🍇 ALTA Institute, Computer Laboratory, University of Cambridge, U.K.
🌻 Writer Inc., San Francisco, California, U.S.A.
🌺 King's College London, U.K.
🌴 Imperial College London, U.K.
🍇 firstname.secondname@cl.cam.ac.uk 🌻 cjb255@cam.ac.uk
🌺 firstname.secondname@kcl.ac.uk 🌴 marek.rei@imperial.ac.uk

Abstract

Thanks to recent advances in generative AI, we are able to prompt large language models (LLMs) to produce texts which are fluent and grammatical. In addition, it has been shown that we can elicit attempts at grammatical error correction (GEC) from LLMs when prompted with ungrammatical input sentences. We evaluate how well LLMs can perform at GEC by measuring their performance on established benchmark datasets. We go beyond previous studies, which only examined GPT∗ models on a selection of English GEC datasets, by evaluating seven open-source and three commercial LLMs on four established GEC benchmarks. We investigate model performance and report results against individual error types. Our results indicate that LLMs do not always outperform supervised English GEC models except in specific contexts – namely commercial LLMs on benchmarks annotated with fluency corrections as opposed to minimal edits. We find that several open-source models outperform commercial ones on minimal edit benchmarks, and that in some settings zero-shot prompting is just as competitive as few-shot prompting.

1 Introduction

Grammatical error correction (GEC) of second language learner English text is an important task in Educational AI. Its main applications include: i) enabling learners to receive instant feedback on their written work, ii) providing features for automarking, and iii) profiling learners' grammatical knowledge in such a way as to facilitate personalised learning (Andersen et al., 2013; Yannakoudakis et al., 2018; Zaidi et al., 2019).

There is a long history of GEC research in the field of computational linguistics, developing from rule-based to statistical approaches to neural network models, as has happened with other tasks in natural language processing (Bryant et al., 2023). Given the recent emergence of large language models (LLMs), such as OpenAI's GPT∗ and Meta's Llama LLMs, it is natural to ask how well they can perform at GEC and how they compare to existing state-of-the-art supervised approaches (Caines et al., 2023).

To answer this question, we aim to elicit minimal edit style corrections from LLMs through zero-shot and few-shot prompting. Minimal edit correction of text aims to resolve any grammatical errors in a text while staying as close as possible to the phrasing and lexical choices of the original. This is sometimes held up as a distinct alternative to fluency correction – where texts are rewritten for naturalness – though in reality the two annotation methods are not completely separable from each other (Bryant et al., 2023). The tendency so far has been to annotate GEC datasets with minimal edits only, and so that is the way past systems have been trained. This is an important point to note, as LLMs by default will output a transformative fluency correction of ungrammatical text (Coyne et al., 2023; Fang et al., 2023; Loem et al., 2023). In this paper, we attempt to prompt LLMs to perform minimal edit correction rather than fluency correction, so that the outputs are comparable to previous systems. Minimal edit corrections are additionally valuable within an educational setting, for example displaying grammatical errors to a learner (Yannakoudakis et al., 2018).

We evaluate three commercial and seven open source LLMs on four publicly available GEC benchmarks: CoNLL 2014 (Ng et al., 2014), the FCE Corpus (Yannakoudakis et al., 2011), JFLEG

Findings of the Association for Computational Linguistics: ACL 2024, pages 11952–11967
August 11-16, 2024 ©2024 Association for Computational Linguistics

(Napoles et al., 2017), and Write&Improve + LOCNESS (W&I) from the BEA-2019 shared task (Bryant et al., 2019). This builds on previous work, which involved GPT∗ models only and at most three GEC benchmarks (Coyne et al., 2023; Fang et al., 2023; Loem et al., 2023). We find that some of our chosen open-source models outperform GPT-3.5 Turbo on English GEC benchmarks which have been annotated in a minimal edit fashion. In contrast, GPT-3.5 performs best on a test set annotated with fluency corrections.

We evaluate several zero-shot and few-shot prompts, but find that different models require different styles of prompting. Some models appear to respond better to few-shot prompting than others, and certain prompt templates work best with a specific LLM and dataset rather than universally across the board. We provide strong empirical evidence that LLMs do not always outperform existing state-of-the-art, supervised GEC models – though the search space over LLMs, prompt templates, and few-shot learning is so great that our results can only be considered a building block in the full picture of LLM evaluation on English GEC. Our investigations can serve as a comparison point for future work on GEC with LLMs. We make our code, prompt templates, few-shot examples, and model predictions publicly available.[1]

2 Related work

Grammatical error correction (GEC). GEC is the task of returning an edited version of an input text such that any errors are corrected. It is a long-standing task in research on NLP for educational applications.

Thanks to large-scale annotation projects and the public release of labelled data, GEC systems can be built aiming at *general* correction of all error types. Machine translation GEC systems pioneered this general purpose approach, at first with statistical and later neural models (Brockett et al., 2006; Junczys-Dowmunt et al., 2018; Yuan and Bryant, 2021). More recently, edit-based approaches have been proposed in which corrections are applied on a sequence labelling (Omelianchuk et al., 2020) or sequence-to-sequence basis (Stahlberg and Kumar, 2020). Bryant et al. (2023) offers a comprehensive survey of the history and current state of GEC.

GEC with large language models (LLMs). The term 'large language model' is currently used to refer to a variety of neural networks developed by a number of organisations and businesses. These models reached the mainstream media through GPT-3 and ChatGPT, and as a result there is now a widespread awareness of 'generative AI' – in particular relating to text generation – amongst the general public. OpenAI's GPT∗ models feature in this paper, alongside others for comparison. We have selected ten open-source and proprietary LLMs for reasons described in Section 4.

Recent studies indicate that LLMs from OpenAI can be prompted to generate corrected versions of ungrammatical inputs. Wu et al. (2023) compares ChatGPT to Grammarly and GECToR (Omelianchuk et al., 2020) on a sample of 100 sentences from the CoNLL-14 test set. Coyne et al. (2023) compares GPT-3.5 and GPT-4[2] to two GEC systems on English benchmarks (Yasunaga et al., 2021; Liu et al., 2021), whilst Fang et al. (2023) compare ChatGPT with multiple baselines including TagGEC (Stahlberg and Kumar, 2021) and T5 (Rothe et al., 2021). Finally Loem et al. (2023) compare ChatGPT and GPT-3.5 (`text-davinci-003`) with models trained on synthetic data (Grundkiewicz and Junczys-Dowmunt, 2019; Grundkiewicz et al., 2019).

Coyne et al. (2023); Fang et al. (2023); Loem et al. (2023) all perform evaluation of English GEC on some combination of the JFLEG, CoNLL-14 and W&I+LOCNESS test sets. They find that the GPT∗ models set new state-of-the-art (SOTA) performance on the JFLEG dataset in which the annotators were permitted to carry out naturalistic fluency rewrites (Napoles et al., 2017). However, they perform worse than SOTA on CoNLL-14 and W&I+LOCNESS, which are much larger, more popular datasets that were annotated on the basis of minimal edit corrections. Coyne et al. (2023) and Fang et al. (2023) found through further investigation that the GPT∗ models have a tendency to over-correct and make extraneous fluency edits. This explains why it is that they can score so highly on JFLEG but not on minimal edit data. Loem et al. (2023) meanwhile investigated the possibility of prompting for minimal edits rather than fluency rewrites, and obtained promising improvements which motivate further work.

[1] `https://github.com/chrisdavis90/gec-prompting-public`

[2] Specifically the `text-davinci-003` GPT-3.5 model, and a GPT-4 model `gpt-4-0314`.

Dataset	Split	# Tokens	# Sentences
W&I+LOCNESS	Dev	86,973	4,384
CoNLL-14	Test	30,144	1,312
JFLEG	Dev	14,010	754
	Test	14,096	747
FCE	Dev	34,748	2,191
	Test	41,932	2,695
W&I+LOCNESS	Train	628,719	34,308
	Sampled	18,386	1,000
FCE	Train	454,736	28,350
	Sampled	16,112	1,000

Table 1: Grammatical error correction datasets.

Both Wu et al. (2023) and Coyne et al. (2023) additionally carried out human evaluation to rate the output from each system and found a preference amongst human raters for the GPT∗ outputs because they were considered to be more fluent. They also found instances of *under*-correction in the reference sentences derived from human annotators: in other words the LLMs were able to catch and correct errors which had not been corrected by the original annotators. These human evaluations are tentative only, since they involve only small samples of 100 sentences at a time from each test set.

While more fluent corrections may be preferred by human evaluators, they may not aid language learners if they diverge too greatly from the original text. Existing annotation guidelines for error correction state that edits should be as minimal as possible so that the learner can be helped to express what they are trying to say, rather than told how to express it differently (which may otherwise discourage them); i.e. how to amend an error rather than avoid it (Nicholls, 2003). Consequently, although both minimal and fluent corrections may be valuable to different user groups, we focus on minimal corrections for educational applications in this paper.

3 Datasets

We compare model performance on four publicly available and well-known English language GEC datasets: CoNLL-14 Test (Ng et al., 2014), JFLEG Dev and Test (Napoles et al., 2017), FCE Dev and Test (Yannakoudakis et al., 2011), and W&I+LOCNESS Dev[3] (Bryant et al., 2019). We additionally sample 2,000 sentences uniformly from FCE train and W&I+LOCNESS train to construct a development set in order to filter the set of prompt templates. Table 1 presents the number of sentences and tokens per dataset.

The CoNLL-14 test set contains 50 essays written by undergraduate students at the National University of Singapore on one of two topics. It has featured in multiple GEC studies, and new SOTA performance was reported by Zhou et al. (2023) in a recent paper describing decoding interventions.

JFLEG dev and test contain approximately 1.5k sentences randomly sampled from essays by learners of English of unknown proficiency levels, and corrected by crowdworkers. Annotators were permitted to make fluency corrections to the sentences: not just minimal edits for grammaticality. Stahlberg and Kumar (2021) achieved the current SOTA performance on JFLEG for a single system with their guided approach to synthetic generation of training data based on error type distributions found in annotated corpora.

FCE dev and test feature essays written by intermediate learners of English (CEFR levels B1 and B2). It is a subset of the Cambridge Learner Corpus and has also been used in multiple GEC studies. Current SOTA was established by Yuan and Bryant (2021) with a multi-encoder model which encodes a given sentence and the preceding one separately, integrating them in the decoder.

W&I+LOCNESS is a hybrid dataset made up of native speaker essays written by undergraduate students (LOCNESS; Granger (1998)) and essays submitted to the Write&Improve learning platform by learners of English at varying levels of proficiency (W&I). It was prepared for the BEA 2019 Shared Task on GEC (Bryant et al., 2019), and SOTA was achieved by Qorib et al. (2022) with system combination across multiple GEC models.

Each dataset was processed with ERRANT (Felice et al., 2016; Bryant et al., 2017), an automatic error annotation tool, in order to be standardised into a common format. Consequently the datasets are in tokenised M2 format, and we first need to detokenise them as LLMs expect untokenised inputs. To carry out this task, we use the Moses detokeniser[4] and a rule-based heuristic to combine negative contractions which are not fully handled by the detokeniser[5].

[3]The test set for W&I+LOCNESS is not publicly available.

[4]https://github.com/luismsgomes/mosestokenizer

[5]The detokeniser transforms token sequences such as "couldn 't" to "couldn't" in a satisfactory manner but sequences such as "could n't" are missed.

Name	Prompt
MIN	Make minimal changes to the following text such that it is grammatically correct. {text}
ELT†	You are an English language teacher. A student has sent you the following text. \n{text}\nProvide a grammatical correction for the text, making only necessary changes. Do not provide any additional comments or explanations. If the input text is already correct, return it unchanged.
TOOL*†	You are a grammatical error correction tool. Your task is to correct the grammaticality and spelling in the input sentence. Make the smallest possible change in order to make the sentence grammatically correct. Change as few words as possible. Do not rephrase parts of the sentence that are already grammatical. Do not change the meaning of the sentence by adding or removing information. If the sentence is already grammatically correct, you should output the original sentence without changing anything. \n\nInput sentence: {text}\nOutput sentence:
DN	Please correct the following text. Do not attempt to rewrite it into perfect English or to interpret the text. Often, things could be expressed better by paraphrase, but the task is to make minimal changes to correct the text. Do not change anything that is correct. Please make no changes if there are no errors.
CYN†	Reply with a corrected version of the input sentence with all grammatical and spelling errors fixed. If there are no errors, reply with a copy of the original sentence.\n\nInput sentence: {text}\nCorrected sentence:
CON	This sentence is ungrammatical: {text}. I would correct the sentence with as few changes as possible like this:

Table 2: The set of prompts used in zero- and few-shot settings. *There are two versions of the TOOL prompt: with and without quotations around the {text}. † indicates prompts used in few-shot evaluation.

4 Models

We evaluate three commercial and seven open-source LLMs. We include more open-source than commercial models as we assume that the latter will have a performance advantage and wish to investigate whether open-source models can perform in comparable ways. If so, this would be positive news from an open-science perspective. For the commercial LLMs, we include OpenAI's GPT-3.5-turbo and GPT-4 models (OpenAI, 2023), and Cohere's Command model.[6] Many more are available but due to budget constraints we work only with these three and we only evaluate GPT-4 in the zero-shot setting. We choose not to work with ChatGPT as it has been engineered to function as a chatbot.

For the open-source models, we select instruction-tuned models because the majority of our prompt templates contain instructions, and we evaluate the largest model from each model type that fits on a server with two A100 80GB NVIDIA GPUs. Our upper bound on model size relates to the computing resources available to us at the time of writing.

The open-source models are: OPT-IML-Max-30B (Iyer et al., 2022), Llama-2-70B-chat (Touvron et al., 2023), Stable Beluga 2 (Mahan et al.), Falcon-40B-Instruct (Almazrouei et al., 2023), Flan-T5-XXL (Chung et al., 2022), BLOOMZ-7B1 (Muennighoff et al., 2022), InstructPalmyra-20B (Writer Inc., 2023). This is a representative sample of the models available, with a range of sizes and architectures. Approximate model sizes are given in Table 7 in the Appendix. We use HuggingFace (Wolf et al., 2020) to run the models and load them with float16 precision.[7]

4.1 Prompting LLMs for grammatical error correction

Prior work has shown that prompt format and wording can have a significant impact on task performance (Jiang et al., 2020; Shin et al., 2020; Schick and Schütze, 2021). We therefore evaluate and compare models across a selection of prompt templates (hereinafter prompts). In order to constrain the scope of experiments, we carry out two filtering and evaluation steps to construct and evaluate a set of zero- and few-shot prompts as follows.

We collect eleven zero-shot prompts based on a survey of NLP colleagues and related work.[8] We first evaluate the zero-shot prompts with each model on a development set of 2,000 sentences sampled uniformly from the FCE and W&I+LOCNESS training sets. From these results we exclude four prompts with the lowest maximum scores, leaving seven prompts to evaluate in the zero-shot setting on the three development datasets: FCE, JFLEG, and W&I+LOCNESS.[9]

[6] Both GPT-3.5-turbo and GPT-4 are the 0613 versions. Cohere's Command is "v1".

[7] We use bfloat16 for Falcon-40B-instruct.

[8] We considered a wide set of prompts used in related work but ultimately decided against their inclusion due to the estimated difficulty in replication and time/budget constraints.

[9] Details are provided in Appendix D.

We then select the 3 best-performing zero-shot prompts and create few-shot versions using 1, 2, 3, and 4 examples – 12 few-shot prompts in total. While related work samples few-shot examples, we make the decision to use a fixed set and order of examples to control the experimental parameters. A dynamic set of few-shot examples would require multiple samples per sentence in order to obtain a clear view of few-shot performance for each model. In addition, we evaluate models using the best performing few-shot prompt from Coyne et al. (2023).

Table 2 lists the prompts we evaluate in zero- and few-shot settings (see Appendix B for the complete set). Briefly, prompts MIN and DN contain general instructions to make minimal corrections, prompt ELT uses an "English language teacher" expert, TOOL uses a "grammatical error correction tool" expert, CYN is the prompt from Coyne et al. (2023), and CON frames the GEC instruction as a continuation. The set of few-shot examples are listed in Appendix Table 9.

Generation hyper-parameters We use the following settings for all models – we set temperature to 0.1, top-K to 50, and top-P to 1.0. Preliminary work has shown that lower temperature values result in better GEC performance (Coyne et al., 2023), and importantly, we want the model to make minimal edits and stay as close as possible to the original sentence. For some models the lowest temperature is 0.1, and so we set the parameter to this value to be constant across all models.

Evaluation As per the recommendations in Bryant et al. (2023), we evaluate the FCE and W&I corpus in terms of $F_{0.5}$ using ERRANT (Bryant et al., 2017), the CoNLL-2014 test set in terms of $F_{0.5}$ using the M^2 scorer (Dahlmeier and Ng, 2012), and the JFLEG corpus using GLEU (Napoles et al., 2015).

Along with the open search space in prompt design, a practical question arises as to how much time and effort to dedicate to implement a model- or prompt-specific post-processing step to extract the generated hypothesis sentence from the model output. Due to the number and variety of models and prompts, it's possible that each model–prompt combination will generate a different output format, and clearly, the quality of the post-processing step will impact evaluation measures. For all models, we replace all new line tokens with blank spaces, replace sequences of multiple spaces with a single space, and remove all trailing quotation marks. We also remove strings from the start and end of sentences based on keyword matching – for example, we remove "Output sentence: ", "Corrected sentence: ", and "Input sentence: " from the start of sentences. The output from Llama-2-chat was particularly noisy and required more rules – we detail our processing steps in Appendix C.

5 Results

Table 3 presents the top-1 results for each model on the development sets for the FCE, JFLEG and W&I+LOCNESS. From the models we test, the results show GPT-4 scores highest on every development dataset, though Stable Beluga 2 and GPT-3.5 Turbo obtain comparable performance to GPT-4 on JFLEG. Amongst the open-source models, Falcon-40B-Instruct and Stable Beluga 2 have relatively high performance across the board, whilst Flan-T5 scores highly on FCE dev specifically.

Contrary to expectations set by previous work, adding few-shot examples to the three zero-shot prompts does not always lead to an improvement in performance. Indeed for FCE dev, zero-shot prompts perform best for most models. The picture is mixed for JFLEG dev, whilst the majority of models benefit from few-shot learning for W&I dev. It remains a matter for future work to investigate whether more dynamic approaches to data sampling (as opposed to a fixed selection of examples) will aid with few-shot GEC prompting.

Table 5 shows the performance of each model (except GPT-4) on the three test sets in our study: FCE, JFLEG and CoNLL-14. We compare with previous work on GEC with LLMs, and SOTA results from GEC-specific systems in the literature. For FCE and JFLEG, we use the prompt template that resulted in the best performance on the corresponding development set. For example, for GPT-3.5 Turbo on the FCE test set, we use the ELT zero-shot prompt because it resulted in the best performance on FCE dev. For CoNLL-14, we do the same based on model performance for W&I+LOCNESS dev.

Our LLM results are well short of SOTA performance, established by task-specific supervised models, for FCE test and CoNLL-14 test – the corpora annotated in minimal edit fashion – whereas the performance of GPT 3.5 Turbo is much closer to the SOTA on JFLEG test. These findings reinforce initial experiments by Coyne et al. (2023),

Model	FCE$_{dev}$			JFLEG$_{dev}$			W&I$_{dev}$		
	$F_{0.5}$	N	Prompt	GLEU	N	Prompt	$F_{0.5}$	N	Prompt
BLOOMZ	0.349	3	CYN	0.456	2	CYN†	0.347	3	CYN
FLAN-T5	0.447	1	TOOL	0.463	1	TOOL	0.423	3	TOOL
InstructPalmyra	0.341	2	CYN	0.517	0	TOOL	0.374	2	CYN
OPT-IML	0.395	0	TOOL	0.506	2	CYN†	0.400	3	ELT
Falcon-40B-Instruct	0.425	2	TOOL	0.548	4	CYN	0.454	4	TOOL
Llama 2	0.323	0	TOOL	0.500	0	TOOL	0.359	0	TOOL
Stable Beluga 2	0.403	0	TOOL	0.563	0	CYN	0.447	0	TOOL
Command	0.353	0	TOOL	0.543	2	CYN†	0.391	0	TOOL
GPT-3.5 Turbo 0613	0.416	0	ELT	0.577	4	TOOL	0.439	1	TOOL
GPT-4 0613*	**0.474**	0	ELT	**0.582**	0	TOOL	**0.510**	0	TOOL
C: GPT 3.5 text-davinci-003	–	–	–	0.582	0	–	–	–	–
C: GPT 3.5 text-davinci-003	–	–	–	0.590	2	–	–	–	–
C: GPT-4 0314	–	–	–	**0.601**	0	–	–	–	–
C: GPT-4 0314	–	–	–	0.600	2	–	–	–	–

Table 3: Results on the FCE, JFLEG, and W&I dev sets, using the best prompt per model. “N” refers to the number of few-shot examples. “Prompt” refers to the type of prompt instruction: TOOL is the GEC tool expert, ELT the English Language Teacher expert, CYN refers to the prompt from Coyne et al. (2023) with our few-shot examples, and CYN† indicates the template with their few-shot examples. Performance reported in previous work is shown in the lower part of the table. C: refers to Coyne et al. (2023). GPT-4* was only evaluated in a zero-shot setting.

Model	P	R	$F_{0.5}$
BLOOMZ	0.475	0.169	0.349
FLAN-T5	0.615	0.213	0.447
InstructPalmyra	0.357	0.287	0.341
OPT-IML	0.559	0.182	0.395
Falcon-40B-Instruct	0.438	0.381	0.425
Llama 2	0.304	0.428	0.323
Stable Beluga 2	0.396	0.432	0.403
Command	0.356	0.342	0.353
GPT-3.5 Turbo 0613	0.398	0.504	0.416
GPT-4 0613	0.473	0.477	0.474

Table 4: Performance for models on the FCE development set, using their best prompts – models ordered by increasing size.

Fang et al. (2023) and Loem et al. (2023). It is apparent that supervised GEC systems, trained on each corpus, are best for minimal edit style corrections, whereas LLMs generate SOTA fluency corrections more similar to the style found in JFLEG.

We find that the four smallest models are biased towards precision over recall, while the larger models are more balanced (Table 4). The GPT∗ models have the best recall, which is a finding that deserves further investigation in future work.

5.1 Error type analysis

We use ERRANT to obtain the grammatical error types found in the W&I+LOCNESS development set – the largest development set we evaluate. ERRANT can identify 55 error classes. Table 6 presents $F_{0.5}$ scores for the 18 most frequent error types for the three best performing models: Falcon-40B-Instruct, Stable Beluga 2, and GPT-3.5 Turbo. Performance for each error type is comparable across the models, though GPT-3.5 Turbo is notably better at replacement punctuation errors.

Generally, the LLMs excel at spelling, missing determiners, replacement subject–verb agreement, replacement noun number, and orthography errors, while struggling on the open-class replacement of nouns and verbs, and the catch-all “other” error type. It seems that the LLMs perform better on morphological or character-based corrections which are not too distant from the original form, whereas lexical or phrasal replacement within the minimal edit paradigm are much more challenging.

6 Discussion

We set out to investigate the performance levels of LLMs on the task of English GEC. Previous work has shown that GPT∗ models could perform GEC with mixed success: outdoing existing SOTA models on the JFLEG dataset, which contains fluency corrections, whilst performing poorly on benchmarks annotated with minimal edit corrections – namely CoNLL-14, the FCE and W&I+LOCNESS (Fang et al., 2023; Loem et al., 2023; Coyne et al., 2023). We aimed to elicit minimal edit corrections through exploration of different prompting strategies, and evaluated models other than GPT∗ – including more open-source than commercial LLMs.

Our findings echo those in previous papers: our chosen LLMs perform well on JFLEG test – above

Model	FCE$_{test}$ F$_{0.5}$	N	Prompt	JFLEG$_{test}$ GLEU	N	Prompt	CoNLL-14$_{test}$ F$_{0.5}$	N	Prompt
BLOOMZ	0.358	3	CYN	0.498	2	CYN†	0.405	3	CYN
FLAN-T5	**0.463**	1	TOOL	0.508	1	TOOL	0.397	3	TOOL
InstructPalmyra	0.396	2	CYN	0.572	0	TOOL	0.499	2	CYN
OPT-IML	0.400	0	TOOL	0.521	2	CYN†	0.396	3	ELT
Falcon-40b-Instruct	0.456	2	TOOL	0.602	4	CYN	0.560	4	TOOL
Llama 2	0.374	0	TOOL	0.560	0	TOOL	0.517	0	TOOL
Stable Beluga 2	0.454	0	TOOL	0.613	0	CYN	**0.572**	0	TOOL
Command	0.408	0	TOOL	0.592	2	CYN†	0.538	0	TOOL
GPT 3.5 Turbo 0613	0.442	0	ELT	**0.625**	4	TOOL	**0.572**	1	TOOL
F: GPT-3.5 Turbo	–	–	–	0.614	0	–	0.517	0	–
F: GPT-3.5 Turbo	–	–	–	0.597	1	–	0.531	1	–
F: GPT-3.5 Turbo	–	–	–	0.635	3	–	0.532	3	–
F: GPT-3.5 Turbo	–	–	–	0.625	5	–	0.528	5	–
L: GPT-3.5 text-davinci-003	–	–	–	0.670	16	–	**0.570**	16	–
L: GPT-3.5 text-davinci-003	–	–	–	**0.693**	64	–	–	–	–
C: GPT-3.5 text-davinci-003	–	–	–	0.634	2	–	–	–	–
C: GPT-4 0314	–	–	–	0.650	2	–	–	–	–
Stahlberg and Kumar (2021)	–			**0.647**			0.666		
Yuan and Bryant (2021)	**0.626**			–			0.629		
Zhou et al. (2023)	–			–			**0.696**		

Table 5: Results on the FCE, JFLEG, and CoNLL-14 test sets. For each model and test set, we use the prompt that results in the best performance on the corresponding dev set. CYN refers to the prompt from Coyne et al. (2023) with our few-shot examples listed in Table 9, while CYN† indicates the prompt from Coyne et al. (2023) with their few-shot examples. Performance reported for GPT∗ in previous work is shown in the middle part of the table, with the number of few-shot examples where applicable. F: refers to Fang et al. (2023), L: to Loem et al. (2023), C: to Coyne et al. (2023). The final section of the table shows SOTA performance by a single non-ensemble system for each test set in the literature. The best scores in each table section are in bold.

all Falcon-40B-Instruct, Stable Beluga 2 and GPT-3.5 – though not outdoing SOTA, possibly because our prompts were designed to discourage fluency style corrections. Based on experiments with JFLEG dev, GPT-4 might perform best on JFLEG test, but full investigation of this question requires additional funding as GPT-4 is currently an order of magnitude more expensive than GPT-3.5 Turbo.

In contrast, the LLMs perform poorly on the FCE and CoNLL-14 test sets, lagging far behind SOTA in both cases. For these datasets, open-source models outperform or compete with the commercial models: the best performing model is FLAN-T5 on the FCE, and Stable Beluga 2 matches GPT 3.5 Turbo in the case of CoNLL-14. Again, performance on the FCE and W&I+LOCNESS dev sets suggests that GPT-4 could outperform the other LLMs on the test sets.

We narrowed down our initial 11 zero-shot prompts to the 7 which performed best on a sample of sentences from the FCE and W&I+LOCNESS training sets. We created few-shot prompts from the 3 best performing zero-shot prompts and varied the number of examples from 1 to 4. The results for zero-shot versus few-shot learning do not clearly show a best method for prompting. The open-source models which perform best on the test sets are FLAN-T5, Falcon-40B-Instruct, and Stable Beluga 2: of these, FLAN-T5 and Falcon-40B-Instruct work best with few-shot learning, whereas Stable Beluga 2 is best with a zero-shot prompt. For GPT-3.5 Turbo, zero-shot is best for FCE test, few-shot is best for JFLEG and CoNLL-14 test.

In terms of prompt wording, the TOOL and CYN prompts are best for the three best open-source models: FLAN-T5, Falcon-40B-Instruct, and Stable Beluga 2. For GPT-3.5 Turbo, the ELT prompt is best for the FCE test set and the TOOL one is best for JFLEG and CoNLL-14 test. Note that two of the three best performing prompts are those in which a role is clearly specified to the LLM – either as an English language teacher or a grammatical error correction tool (Table 2).

The fact that the other best performing prompt, the one from Coyne et al. (2023), replicates the strong results from that paper is evidence for convergence around optimal prompt crafting. Further exploration of the huge prompt search space is possible, but we show that the CYN prompt holds up

Error	Falcon	GPT-3.5	StableB2
M:DET	0.643	0.620	0.638
M:OTHER	0.155	0.175	0.221
M:PREP	0.447	0.403	0.422
M:PUNCT	0.570	0.475	0.470
R:DET	0.375	0.353	0.362
R:MORPH	0.444	0.395	0.399
R:NOUN	0.291	0.261	0.284
R:NOUN:NUM	0.633	0.570	0.593
R:ORTH	0.597	0.609	0.589
R:OTHER	0.281	0.300	0.296
R:PREP	0.490	0.488	0.466
R:PUNCT	0.365	0.503	0.315
R:SPELL	0.781	0.769	0.761
R:VERB	0.219	0.253	0.258
R:VERB:FORM	0.552	0.486	0.454
R:VERB:SVA	0.641	0.571	0.611
R:VERB:TENSE	0.499	0.471	0.516
U:DET	0.530	0.555	0.554

Table 6: $F_{0.5}$ for the 18 most frequent error types in the W&I+LOCNESS development set, for the 3 best performing models: Falcon-40B-Instruct, GPT-3.5 Turbo, and Stable Beluga 2.

well against a set of alternatives, and can therefore be considered a strong baseline for future GEC experiments.

Another provision we make for replication in future studies is to supply the list of examples we used in few-shot learning (Table 9). This allows others to use them for their own novel prompts, while holding constant the nature of the examples. Furthermore we believe that alternative methods for sourcing few-shot examples could be explored in future work, as discussed below.

Finally, we note that the comparison between commercial and open-source LLMs is not entirely even, as the former sit behind APIs and a black box processing pipeline. We recognise that GPT-3.5 Turbo shows great promise for English GEC, at least for fluency corrections, but we also find that several open-source models perform relatively well – in fact better than GPT-3.5 on benchmarks annotated with minimal edits. This is a boon for open science, because models which researchers can obtain and work with directly lead to greater transparency in GEC and beyond.

7 Conclusion

We have shown that LLMs do not always outperform existing SOTA models for English GEC: for minimal edit style datasets such as the FCE, CoNLL-14 and W&I+LOCNESS, their performance is far below that of supervised GEC systems. We attempted to elicit minimal edit corrections from LLMs through prompt crafting, but it may be that LLMs are still biased towards fluency rewrites as has been shown in previous work (Coyne et al., 2023; Fang et al., 2023; Loem et al., 2023). This is consistent with our finding, echoing that of others, that LLMs perform best on JFLEG, which was annotated with a fluency correction style.

We arrive at the following conclusions: (i) Supervised models are still best for English GEC with minimal edit corrections; (ii) Further explorations of prompt crafting, few-shot learning, and dynamic sampling are justified, as is work with open-source models as opposed to commercial ones; (iii) Methods for improving LLM performance on specific error types could be explored.

Other potential areas for future work include document-level GEC and human evaluation of proposed corrections. We worked with sentence-level GEC, but this deviates from the greater amount of essay context given to annotators. Document-level GEC has been proposed and recommended in previous work (Yuan and Bryant, 2021; Coyne et al., 2023). Exploratory work by Fang et al. (2023) showed that ChatGPT could not perform document-level GEC well, and speculated that it may not be able to handle long inputs requiring "high levels of coherence and consistency between sentences". We notice that LLMs are better at GEC on beginner and intermediate texts, rather than advanced or native-speaker ones (Appendix Table 13): further investigation is needed on this matter.

Initial human evaluation studies suggest a preference for the corrections generated by LLMs over the reference corrections contained in GEC corpora (Coyne et al., 2023; Fang et al., 2023). It may be that human raters prefer to read the more fluent LLM-derived corrections but minimal edit corrections are actually more helpful for language learning since they are more faithful to the original intended meaning of the writer. Investigating learning benefits from receiving minimal edit grammatical feedback as opposed to fluency rewrites is a matter for future work which will involve longitudinal data collection, a focus on different feedback styles, and tracking how learners respond.

Acknowledgements

This paper reports on research supported by Cambridge University Press & Assessment. Some of this work was performed using resources provided by the Cambridge Service for Data Driven Discovery (CSD3) operated by the University of Cambridge Research Computing Service, provided by Dell EMC and Intel using Tier-2 funding from the Engineering and Physical Sciences Research Council (capital grant EP/T022159/1), and DiRAC funding from the Science and Technology Facilities Council. We acknowledge the Cohere For AI Research Grant that supported our evaluation of the Cohere Command model. We thank Yulong Lin for his exploratory work on GEC with LLMs.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. Falcon-40B: an open large language model with state-of-the-art performance.

Øistein E. Andersen, Helen Yannakoudakis, Fiona Barker, and Tim Parish. 2013. Developing and testing a self-assessment and tutoring system. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 32–41, Atlanta, Georgia. Association for Computational Linguistics.

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 249–256, Sydney, Australia. Association for Computational Linguistics.

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, 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.

Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe. 2023. Grammatical Error Correction: A Survey of the State of the Art. *Computational Linguistics*, pages 1–59.

Andrew Caines, Luca Benedetto, Shiva Taslimipoor, Christopher Davis, Yuan Gao, Øistein Andersen, Zheng Yuan, Mark Elliott, Russell Moore, Christopher Bryant, Marek Rei, Helen Yannakoudakis, Andrew Mullooly, Diane Nicholls, and Paula Buttery. 2023. On the application of large language models for language teaching and assessment technology. In *Proceedings of the Empowering Education with LLMs – the Next-Gen Interface and Content Generation Workshop at AIED*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Steven Coyne, Keisuke Sakaguchi, Diana Galvan-Sosa, Michael Zock, and Kentaro Inui. 2023. Analyzing the performance of GPT-3.5 and GPT-4 in Grammatical Error Correction. arXiv:2303.14342.

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.

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:2304.01746.

Mariano Felice, Christopher Bryant, and Ted Briscoe. 2016. Automatic extraction of learner errors in ESL sentences using linguistically enhanced alignments. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 825–835, Osaka, Japan. The COLING 2016 Organizing Committee.

Sylviane Granger. 1998. The computer learner corpus: A versatile new source of data for SLA research. In Sylviane Granger, editor, *Learner English on Computer*. London & New York: Addison Wesley Longman.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2019. Minimally-augmented grammatical error correction. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 357–363, Hong Kong, China. 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.

Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, et al. 2022. Opt-iml: Scaling language model instruction meta learning through the lens of generalization. arXiv:2212.12017.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. 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.

Zhenghao Liu, Xiaoyuan Yi, Maosong Sun, Liner Yang, and Tat-Seng Chua. 2021. Neural quality estimation with multiple hypotheses for grammatical error correction. arXiv:2105.04443.

Mengsay Loem, Masahiro Kaneko, Sho Takase, and Naoaki Okazaki. 2023. Exploring effectiveness of GPT-3 in grammatical error correction: A study on performance and controllability in prompt-based methods. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 205–219, Toronto, Canada. Association for Computational Linguistics.

Dakota Mahan, Ryan Carlow, Louis Castricato, Nathan Cooper, and Christian Laforte. Stable Beluga models.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. 2022. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*.

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.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

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.

Diane Nicholls. 2003. The Cambridge Learner Corpus – error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 conference; UCREL technical paper number 16*.

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

OpenAI. 2023. GPT-4 technical report. *arXiv*, arXiv:2303.08774.

Muhammad Reza Qorib, Seung-Hoon Na, and Hwee Tou Ng. 2022. Frustratingly easy system combination for grammatical error correction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1964–1974, Seattle, United States. Association for Computational Linguistics.

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.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Felix Stahlberg and Shankar Kumar. 2020. Seq2Edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on*

Empirical Methods in Natural Language Processing (EMNLP), pages 5147–5159, Online. Association for Computational Linguistics.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47, Online. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Writer Inc. 2023. InstructPalmyra-20b: Instruct tuned Palmyra-Large model.

Haoran Wu, Wenxuan Wang, Yuxuan Wan, Wenxiang Jiao, and Michael Lyu. 2023. ChatGPT or Grammarly? Evaluating ChatGPT on grammatical error correction benchmark. arXiv:2303.13648.

Helen Yannakoudakis, Øistein E Andersen, Ardeshir Geranpayeh, Ted Briscoe, and Diane Nicholls. 2018. Developing an automated writing placement system for ESL learners. *Applied Measurement in Education*, 31(3):251–267.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2021. LM-Critic: language models for unsupervised grammatical error correction. arXiv:2109.06822.

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.

Ahmed H Zaidi, Andrew Caines, Christopher Davis, Russell Moore, Paula Buttery, and Andrew Rice. 2019. Accurate modelling of language learning tasks and students using representations of grammatical proficiency. In *Proceedings of The 12th International Conference on Educational Data Mining (EDM)*.

Houquan Zhou, Yumeng Liu, Zhenghua Li, Min Zhang, Bo Zhang, Chen Li, Ji Zhang, and Fei Huang. 2023. Improving Seq2Seq grammatical error correction via decoding interventions. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore.

A Model sizes

Model	Size
bloomz-7b1	7B
flan-t5-xxl	11B
InstructPalmyra-20b	20B
opt-iml-max-30b	30B
falcon-40b-instruct	40B
StableBeluga2	70B
Llama-2-70b-chat-hf	70B
Cohere Command	–
OpenAI gpt-3.5-turbo-0613	–
OpenAI gpt-4-0613	–

Table 7: List of models and their approximate sizes.

B Prompt Templates

Table 8 includes the complete list of zero-shot prompt templates used to perform GEC with LLMs. While the majority of the models use these templates, four models recommend a predefined prompt format – we describe model-specific prompts below. Table 9 contains the list of examples used in the few-shot prompts.

B.1 OpenAI GPT-∗

We use the OpenAI ChatCompletion endpoint that formats prompts with separate System and User messages. We adapt our prompts and put the instruction in the System message, and the learner sentence with any "Input" tags in the User message. For few-shot prompts, we format each example using separate User and Assistant messages, to mimic a chat-history as context – see Table 10 for an example.

B.2 Llama-2-chat

Llama-2-chat is trained with the following structure for the first turn in chat applications:[10]

```
<s>[INST] <<SYS>>
{system_prompt}
<</SYS>>

{input} [/INST]
```

We insert the entire GEC instruction into the `system_prompt`, and the learner sentence into the `input`. Where a prompt template uses "Input:"/"Output:" tags, we append the output tags after the final `[\INST]`.

For the few-shot prompts, we follow the conversational setup and include examples as:

```
{input 1} [/INST] {hypothesis 1}
</s><s>[INST] {input 2} [/INST]
```

B.3 Stable Beluga 2

Stable Beluga 2 recommends structuring prompts with `System`, `User`, and `Response` tags:

```
### System:
This is a system prompt, please behave
and help the user.

### User:
{input}

### Assistant:
{The output of Stable Beluga 2}
```

B.4 InstructPalmyra-20B

InstructPalmyra recommends the following prompt format, including a preamble followed by Instruction, Input, and Response tags:

```
Below is an instruction that describes a
task, paired with an input that provides
further context. Write a response that
appropriately completes the request.
\n\n### Instruction:\n {instruction}
\n\n### Input:\n {input}\n\n### Response:
```

C Post-processing model output

For each model, we process the output with the following rules:

1. Remove "Output sentence: ", "Corrected sentence: ", and "Input sentence: " from the start of sentences.
2. Strip double-quotes.
3. If there is an odd number of quotations, we remove trailing quotations.
4. For Llama 2, we search for and remove strings from a keyword list (included in Table 11).
5. For Llama 2, we split model generations based on the keyword list in Table 11.
6. For Falcon-40B-Instruct, we split model generations based on "Input sentence:" – this mainly impacts the few-shot setting, where the model tends to continue the few-shot pattern and generate a novel learner sentence after the correction.

D Filtering zero-shot prompts with a sampled development set

We evaluated a long-list of eleven zero-shot prompts with each model on a development set of 2,000 sentences sampled uniformly from the

[10] https://huggingface.co/blog/llama2#how-to-prompt-llama-2

Index	Shorthand	Prompt
1	.	Correct the errors. Do not paraphrase.
2	.	Grammar.
3	MIN	Make minimal changes to the following text such that it is grammatically correct.
4	.	You are an English language teacher. A student has sent you the following essay. \n{text}\nCorrect the errors in the essay that will best help the student to learn from their mistakes.
5	ELT	You are an English language teacher. A student has sent you the following text. \n{text}\nProvide a grammatical correction for the text, making only necessary changes. Do not provide any additional comments or explanations. If the input text is already correct, return it unchanged.
6	TOOL	You are a grammatical error correction tool. Your task is to correct the grammaticality and spelling in the input sentence. Make the smallest possible change in order to make the sentence grammatically correct. Change as few words as possible. Do not rephrase parts of the sentence that are already grammatical. Do not change the meaning of the sentence by adding or removing information. If the sentence is already grammatically correct, you should output the original sentence without changing anything. \n\nInput sentence: {text}\nOutput sentence:
7	TOOL	You are a grammatical error correction tool. Your task is to correct the grammaticality and spelling in the input sentence. Make the smallest possible change in order to make the sentence grammatically correct. Change as few words as possible. Do not rephrase parts of the sentence that are already grammatical. Do not change the meaning of the sentence by adding or removing information. If the sentence is already grammatically correct, you should output the original sentence without changing anything. \n\nInput sentence: “{text}”\nOutput sentence: “
8	DN	Please correct the following text. Do not attempt to rewrite it into perfect English or to interpret the text. Often, things could be expressed better by paraphrase, but the task is to make minimal changes to correct the text. Do not change anything that is correct. Please make no changes if there are no errors.
9	.	Correct this to standard English:
10	CYN	Reply with a corrected version of the input sentence with all grammatical and spelling errors fixed. If there are no errors, reply with a copy of the original sentence.\n\nInput sentence: {text}\nCorrected sentence:
11	CON	This sentence is ungrammatical: {text}. I would correct the sentence with as few changes as possible like this:

Table 8: Zero-shot prompts. The prompts without a shorthand were removed after the first evaluation phase on 2,000 trial sentences (Appendix D).

FCE and W&I+LOCNESS training sets. We report $F_{0.5}$ scores as calculated using the automatic scorer in ERRANT. Table 12 presents the score for the top-1 performing prompt for each model and prompt-type.

We find Stable Beluga 2 and GPT-3.5 Turbo perform the best and obtain comparable performance using different prompts: the former using the "GEC tool" expert and the latter using the "English language teacher". Indeed, we observe that the two expert prompts and the prompt from Coyne et al. (2023) result in the best performance across the models.

Figure 1a and 1b illustrate $F_{0.5}$ scores for models using the zero-shot prompts, evaluated on the sampled development set. In the former, we can see that Dolly-v2-12B stands out with particularly low performance across all prompts. While in the latter, we can see that prompts 2, 4, and 9 have the lowest maximum scores. Additionally, prompts 4 and 5 are paired: they both use the "English language teacher" expert template, but prompt 5 contains more detailed instructions. It is clear from the plot that the more detailed instructions tend to result in higher performance.

From these results, we exclude zero-shot prompts 1, 2, 4, and 9 from the final evaluation due to their relatively low performance with every model. We additionally exclude Dolly-v2-12B due to its low performance across every prompt.

Index		Prompt
1	Input	I love this sport. I look forward to the weakened, to go out with my bike and my group of friends.
	Output	I love this sport. I look forward to the weekend to go out with my bike and my group of friends.
2	Input	Lucy Keyes was the last thriller I've seen.
	Output	Lucy Keyes was the last thriller I saw.
3	Input	In the biggest cities around the world the traffic nonstop and increase every day.
	Output	In the biggest cities around the world, the traffic is nonstop and increasing every day.
4	Input	Also, the satisfaction of the customers pushes me to work harder and be better at my job.
	Output	Also, the satisfaction of the customers pushes me to work harder and be better at my job.

Table 9: The list of examples used in few-shot prompts. For example, 3-shot prompts include examples, in order, 1, 2, and 3.

Type	Message
System	Reply with a corrected version of the input sentence with all grammatical and spelling errors fixed. If there are no errors, reply with a copy of the original sentence.
User	Input sentence: I think smoke should to be ban in all restarants.
Assistant	Corrected sentence: I think smoking should be banned at all restaurants.
User	Input sentence: We discussed about the issu.
Assistant	Corrected sentence: We discussed the issue.
User	Input sentence: text

Table 10: Example formatting for a few-shot prompt template with OpenAI's Chat Completion endpoint.

E Results on the development sets

Table 4 shows precision, recall and $F_{0.5}$ on the FCE development set. We find that the four smallest models have a bias towards precision over recall, while the larger models are more balanced.

Figure 2 presents the scores for each model on the FCE, JFLEG and W&I+LOCNESS development sets with our seven zero–shot prompts. We observe that InstructPalmyra and Stable Beluga 2 have much smaller variance in both zero- and few-shot settings. On the other hand we observe high variability with different prompts for OPT-IML and Falcon-40B-Instruct. For most models, we observe more consistent performance in the few-shot settings.

Figure 3 presents the scores for each model on the FCE development set with the prompts CYN, ELT, TOOL in zero- and few-shot settings. BLOOMZ, OPT-IML, and Falcon-40B-Instruct stand out as particularly sensitive to the choice of prompt – in particular, OPT-IML scores ∼0 $F_{0.5}$ using the MIN, ELT, and DN prompts on each development set.[11]

[11]OPT-IML generates empty hypotheses for the majority of sentences with prompts MIN, ELT, and DN.

Start of sentence keyword list
"Sure! Here"
"Sure! The sentence"
"Here is a"
"Here's a"

Truncation keyword list
"(No changes"
"Explanation:"
"(The corrections"
"(No correction"
"Corrections:"
"Is there anything"
"Here's a list of"
"Here is a list of"
"The original sentence"
"(The original sentence"
"In the original sentence"
"(The sentence"
"(The only error in"
"(Changes made:"
"(The change made"
"(Note: "
"The main issue"
"The only change I made"
"I changed"
"I made.*changes"

Table 11: List of keywords used to clean generations from LLama-2-chat.

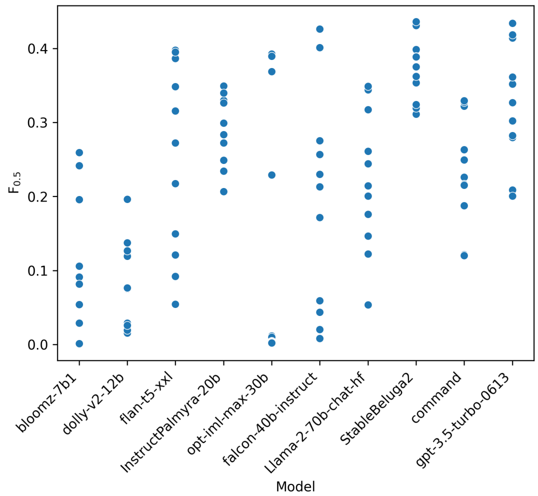
E.1 Proficiency level analysis

We report performance on the W&I+LOCNESS development set grouped by CEFR level in Table 13. The majority of models perform relatively well on A-level learner text (beginners), followed by intermediate B-level text, English text written by native speakers, and finally advanced learner C-level text.

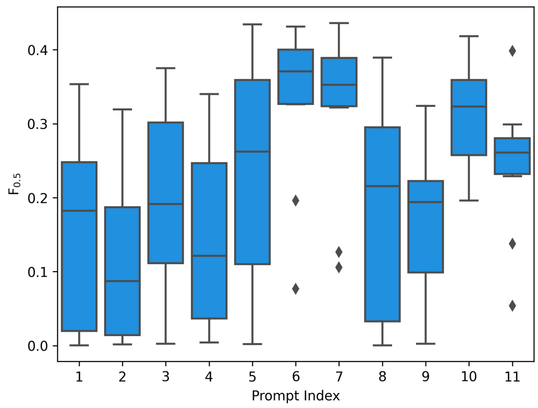
Interestingly, BLOOMZ, FLAN-T5, and OPT-IML perform best on native speaker text. A closer inspection of the precision and recall results show all of these models have a bias towards high precision and low recall.

Model	Prompt	$F_{0.5}$
BLOOMZ	CYN	0.259
FLAN-T5	TOOL	0.398
InstructPalmyra	ELT	0.349
OPT-IML	TOOL	0.393
Falcon-40B-Instruct	TOOL†	0.426
Llama 2	TOOL	0.349
Stable Beluga 2	TOOL†	**0.436**
Command	CYN	0.330
GPT-3.5 Turbo 0613	ELT	0.434

Table 12: Top-1 performing zero-shot prompt for each model on the sampled development set. Refer to Table 8 for the prompts. † indicates the prompt with quotations around the input sentence.



(a) Performance by model.



(b) Performance by prompt.

Figure 1: Performance of models using zero-shot prompts on 2,000 sentences sampled uniformly from the FCE and W&I training sets (1,000 sentences each).

Group	A	B	C	NS
BLOOMZ	0.349	0.328	0.328	0.396
Flan-T5	0.428	0.386	0.353	0.532
InstructPalmyra	0.408	0.375	0.280	0.388
OPT-IML	0.421	0.359	0.325	0.486
Falcon-40B-instruct†	0.487	0.465	0.373	0.434
Llama-2	0.412	0.380	0.273	0.315
StableBeluga2†	0.490	0.462	0.344	0.434
Command	0.440	0.400	0.284	0.376
GPT-3.5-turbo†	0.488	0.457	0.344	0.401
GPT-4	0.547	0.516	0.427	0.495

Table 13: $F_{0.5}$ for for each proficiency level in the W&I+LOCNESS development set. † indicates the top 3 performing models for the dataset: Falcon-40B-Instruct, GPT-3.5 Turbo, and Stable Beluga 2. A = beginner learner, B = intermediate, C = advanced, NS = native speaker of English.

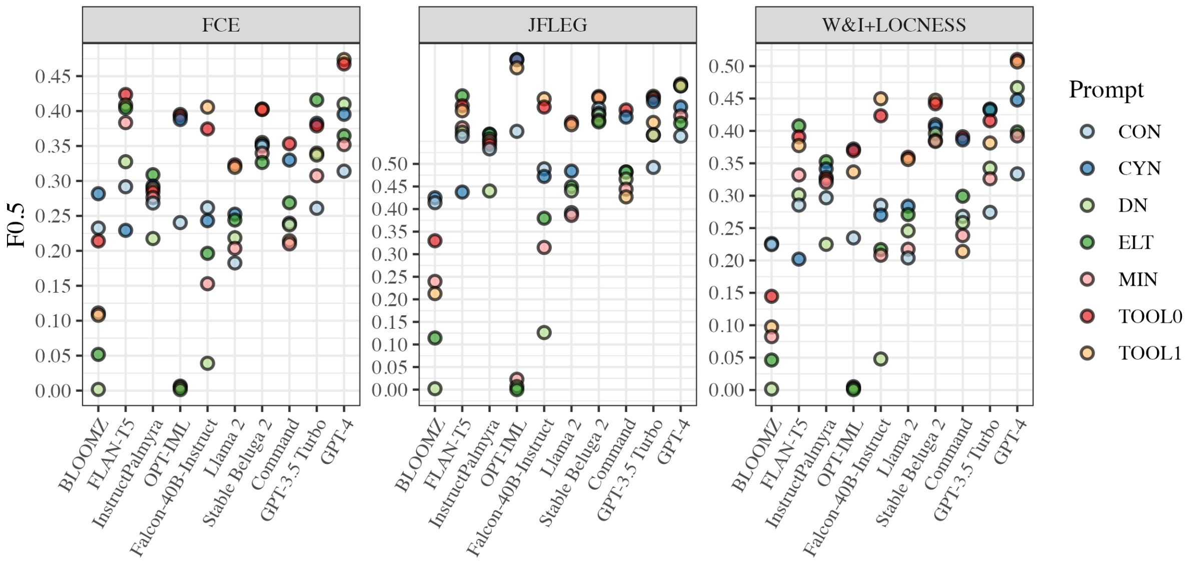


Figure 2: Performance per model and prompt on the FCE development set: $F_{0.5}$ for each model with our seven zero-shot prompts on the FCE, JFLEG and W&I+LOCNESS development sets. TOOL0 is prompt 6 in Table 8 (without quote marks); TOOL1 is prompt 7 (with quote marks).

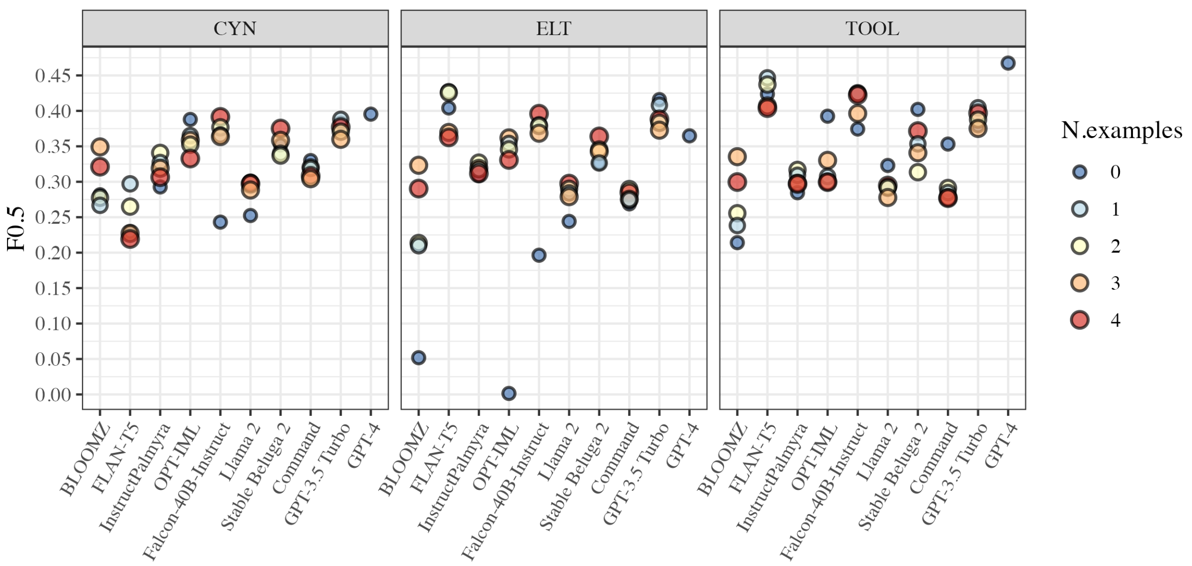


Figure 3: Performance per model and prompt on the FCE development set: $F_{0.5}$ for each model with the prompts CYN, ELT, TOOL in zero- and few-shot settings. GPT-4 was only evaluated with zero-shot prompts due to budget constraints.