

GrammarGenius: A Generative AI Chat Model with Enhanced Grammatical Error Correction Capability

by

Sadia Ahmed Bushra (201014005)
Razaan Reza (201014045)
Apon Ghosh (201014069)

*Capstone Project (CSE499) Report submitted in partial fulfillment of the
requirements for the degree of*

Bachelor of Science in Computer Science and Engineering

Under the supervision of

Mohammed Ashikur Rahman, PhD



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF LIBERAL ARTS BANGLADESH

SUMMER 2024

© *Sadia Ahmed Bushra, Razaan Reza, and Apon Ghosh*
All rights reserved

DECLARATION

Project Title GrammaGenius: A Generative AI Chat Model with Enhanced Grammatical Error Correction Capability
Authors *Sadia Ahmed Bushra, Razaan Reza, and Apon Ghosh*
Student IDs 201014005, 201014045, and 201014069
Supervisor Mohammed Ashikur Rahman, PhD

We declare that this capstone project report entitled *GrammaGenius: A Generative AI Chat Model with Enhanced Grammatical Error Correction Capability* is the result of our own work except as cited in the references. The capstone project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Sadia Ahmed Bushra
201014005

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Razaan Reza
201014045

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Apon Ghosh
201014069

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Date: November 3, 2024



Department of Computer Science and Engineering
University of Liberal Arts Bangladesh
Mohammadpur, Dhaka - 1207

CERTIFICATE

This is to certify that the capstone project report entitled **GammaGenius: A Generative AI Chat Model with Enhanced Grammatical Error Correction Capability**, submitted by **Sadia Ahmed Bushra** (Student ID: 201014005), **Razaan Reza** (Student ID: 201014045), **Apon Ghosh** (Student ID: 201014069) are undergraduate students of the **Department of Computer Science and Engineering** has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Computer Science and Engineering*.

Mohammed Ashikur Rahman, Assistant Professor
Dept. of CSE, University of Liberal Arts Bangladesh

Prof. Muhammad Golam Kibria, PhD, SMIEEE
Dept. of CSE, University of Liberal Arts Bangladesh

Raihan Kibria, Senior Lecturer
Dept. of CSE, University of Liberal Arts Bangladesh

Sovon Chakraborty, Lecturer
Dept. of CSE, University of Liberal Arts Bangladesh

Place: Dhaka

Date: November 3, 2024

ACKNOWLEDGEMENTS

We would like to express our deep and sincere gratitude to our research supervisor, *Mohammed Ashikur Rahman, PhD*, for giving us the opportunity to conduct research and providing invaluable guidance throughout this work. His dynamism, vision, sincerity and motivation have deeply inspired us. He has taught us the methodology to carry out the work and to present the works as clearly as possible. It was a great privilege and honor to work and study under his guidance.

We are greatly indebted to our honorable teachers of the Department of Computer Science and Engineering at the University of Liberal Arts Bangladesh who taught us during the course of our study. Without any doubt, their teaching and guidance have completely transformed us to the persons that we are today.

We are extremely thankful to our parents for their unconditional love, endless prayers and caring, and immense sacrifices for educating and preparing us for our future. We would like to say thanks to our friends and relatives for their kind support and care.

Finally, we would like to thank all the people who have supported us to complete the project work directly or indirectly.

Sadia Ahmed Bushra, Razaan Reza, and Apon Ghosh

University of Liberal Arts Bangladesh

Date: November 3, 2024

Dedicated to
the best Mother
my inspirational sister
Abbu, Alvee,
and my best friends Razaan
and Apon

– *Sadia Ahmed Bushra*

Dedicated to
my beloved parents
the source of my inspiration
and my group members

– *Razaan Reza*

Dedicated to
my beloved parents
and my group members
good souls.

– *Apon Ghhosh*

ABSTRACT

This report describes the process of improving the Llama-2-7B chat model to correct grammatical errors using the seq2seq (sequence-to-sequence) method. For this, we used the JFLEG dataset, which contains sentences with mistakes and their corrected versions. The goal was to teach the model how to turn sentences with grammar issues into correctly structured sentences. We fine-tuned the model on the sentence pairs present in the JFLEG dataset to train the model for identification and error correction in real-time text. This report involves the data preparation of the project, training of the model, and testing based on tensorboard logs associated with grammatical correction. Our fine-tuned model shows promising results and indicates that our model performs better in grammatical correction, which therefore can be useful in the improvement of language fluency and accuracy.

Keywords: GEC, LLM, GEN AI, Llama, seq2seq

Contents

1	Introduction	1
1.1	Problem statement	2
1.2	Aims, Objectives, Motivation, and Significance	3
1.2.1	Aims	3
1.2.2	Objectives	3
1.2.3	Motivation	3
1.2.4	Significance	4
1.3	Project Specification	5
1.4	Contribution of The Project	5
1.5	Report Layout	6
2	Literature Review	8
2.1	Preliminaries	8
2.1.1	Generative AI	8
2.1.2	LLM	8
2.1.3	Chatbot	9
2.1.4	ChatGPT	9
2.1.5	Llama 2	9
2.1.6	Fine-Tuning	10
2.1.7	Grammatical Error Correction Task	10
2.1.8	Grammatical Error Correction Dataset	12
2.1.8.1	JFLEG	12
2.1.9	Sequence-To-Sequence	12
2.2	Systematic Literature Review	14
2.2.1	Introduction	14
2.2.1.1	Systematic Review	14

2.2.1.2	Rationale	16
2.2.1.3	Objectives	16
2.2.1.4	Research Questions	17
2.2.2	Methods	17
2.2.2.1	Eligibility Criteria	17
2.2.2.2	Information Sources	19
2.2.2.3	Search Strategy	19
2.2.2.4	Selection Process	20
2.2.2.5	Data Collection Process	20
2.2.3	Results	21
2.2.3.1	Study Selection	21
2.2.3.2	Study Characteristics	23
2.2.3.3	Result of Individual Studies	23
2.2.4	Findings	29
3	Methodology	35
3.1	System Development Model	36
3.1.1	Scrum	36
3.1.2	Application of Agile Scrum Framework	37
3.2	System Analysis	38
3.2.1	Problem Identification	38
3.2.2	Requirement Analysis	39
3.2.3	System Design	39
3.2.3.1	Use Case Diagram	40
3.2.3.2	Activity Diagram	41
3.2.3.3	Sequence Diagram	43
3.2.3.4	Class Diagram	45
3.2.3.5	State Diagram	48
3.2.4	Development	50
3.2.4.1	Python	50
3.2.4.2	Python Libraries	51
3.2.5	Testing	52
3.3	Software Requirement for Development	52
3.3.1	Google Colab	52

3.3.2	Anaconda	52
3.3.3	Jupyter Notebook	53
3.3.4	Mendeley	53
3.3.5	LaTeX	53
3.3.6	Lucid chart	54
3.3.7	Visual Paradigm	54
3.4	Dataset Preparation	55
3.4.1	Data Collection	55
3.4.2	Dataset Pre-processing	55
4	Design and Development	57
4.1	Project Management	57
4.1.1	Work Breakdown Structure	58
4.2	Proposed Model Deployment	60
4.2.1	Fine Tuning Llama-2 7B Chat Model	60
4.2.1.1	Saving and Loading Fine-Tuned Llama-2 7B Chat Model	60
4.3	Implementation	61
4.3.1	Front-end using Gradio Library	61
4.3.2	Back-end using Python	62
4.3.3	Algorithm: How Sequence-To-Sequence Is Implemented? . .	62
4.4	Model Output Snapshots	64
5	Result Analysis	67
5.1	Performance Analysis	67
5.1.1	Training Epochs	68
5.1.2	Learning Rate Decay	69
5.1.3	Training Loss Reduction	70
5.1.4	Testing Loss Graph	72
5.2	Discussion	73
6	Conclusions	74
6.1	Social, Ethical, and Environmental Issues and Impacts	74
6.1.1	Social	75
6.1.2	Ethical	75

6.1.3	Environmental	76
6.2	Brief Summary	76
6.3	Future Works	77
References		78

List of Figures

2.1	Sequence-to-sequence Block Diagram	13
2.2	A flow diagram summary of the result of the search and selection process	22
3.1	System Development Model	37
3.2	Use Case Diagram for GrammaGenius	40
3.3	Activity Diagram for GrammaGenius	42
3.4	Sequence Diagram for GrammaGenius	44
3.5	Class Diagram of GrammaGenius	46
3.6	State Diagram of GrammaGenius	49
4.1	WBS of GrammaGenius	59
4.2	Output of Our GrammaGenius Chat Model successfully correcting tense, subject-verb agreement, morphological, punctuation, preposition, syntax, determiner errors in the inputted sentences	64
4.3	Output of our GrammaGenius Chat Model successfully correcting tense, subject-verb agreement, morphological errors in an input paragraph	65
4.4	Performing GEC Task on The Model Fine-tuned with GEC Dataset .	66
5.1	Train/Epoch graph of the model GrammaGenius	68
5.2	Train/learning rate of the model GrammaGenius	69
5.3	Train/loss graph of the model GrammaGenius	70
5.4	Eval/loss graph of the model GrammaGenius	72

List of Tables

2.1	Examples of Different Grammatical Error Types and Their Corrections.	11
2.2	Summary of performance strengths of ChatGPT in English GEC tasks.	30
2.3	Summary of performance weaknesses of ChatGPT in English GEC tasks.	31
2.4	Techniques do ChatGPT and Llama models use for grammatical error correction.	32
2.5	Techniques do ChatGPT and Llama models use for grammatical error correction.	33
2.6	Evaluation Metrics for GEC tasks	34

Chapter 1

Introduction

The widespread adoption of Gen AI technologies, such as ChatGPT, BARD, LLAMA, and Gemini, has begun to fundamentally alter how we operate and interact. Generative AI systems increasingly will be used to aid people as intelligent question-answering systems [Omar et al. \(2023\)](#), as well as for brainstorming, content generation, creative writing, summarizing [Yang et al. \(2023\)](#), paraphrasing, grammatical error correction [Fang et al. \(2023\)](#); [Wu et al. \(2023\)](#) etc.

Grammatical Error Correction (GEC) is a prominent NLP task that aims to fix grammar-related mistakes in written text automatically. The core objective of GEC is a combination of the two. The first is determining and efficaciously repairing as many faults as possible. The second one is preserving the original right words without including any new words. Subconsciously, GEC may be thought of as a machine translation task, where the wrong text is taken as the source language, and the corrected text can be taken as the intended outcome. Grammatical mistakes can be classified into several forms, spanning from prepositions, articles or determiners, subject-verb agreement, punctuation, noun form, verb form, word choice, pronouns, sentence structure, capitalization, etc. Determiners and prepositions are among the most common errors made by English language learners.

Current GEC algorithms perform poorly for many errors, making this task challenging and requiring more study. Furthermore, addressing this issue has profound implications since it is predicted that hundreds of millions of individuals worldwide are learning English and gain straightaway from an automatic grammar checker [Ng et al. \(n.d.\)](#). There is a significant possibility for such a system to be employed in a variety of fields, including document proofreading and editing, writing assis-

tance, language acquisition training, and text post-processing for automated speech recognition [Leng et al. \(n.d.\)](#).

1.1 Problem statement

The present research landscape reveals numerous constraints in conversational AI's performance on GEC tasks. The challenges include under-correction issues such as prepositions, subject-verb agreement, articles or determiners, punctuation, and sentence structure, as well as over-correction cases like problems while handling lengthier sentences and changes in meaning and structure.

The sequence-to-sequence (Seq2Seq) [Malmi et al. \(2019\)](#); [Zhao et al. \(2022\)](#) technique has lately gained popularity in GEC and demonstrates favorable outcomes. Recent studies demonstrate that the Seq2Seq strategy consistently performs better than the Seq2Edit approach across a wide range of languages and datasets, particularly when dealing with more complicated problems. The Seq2Seq strategy considers GEC as a monolingual text translation process. However, the Seq2Seq-based method has demonstrated state-of-the-art (SOTA) performance on a variety of benchmarks [Chen et al. \(n.d.\)](#); [Malmi et al. \(2019\)](#); [Zhao et al. \(2022\)](#).

1.2 Aims, Objectives, Motivation, and Significance

1.2.1 Aims

The proposed system aims to develop a conversational AI system that has improved GEC capabilities. The system would use the LLaMA 2 chat model as the base model and incorporate seq2seq techniques with GEC benchmark datasets like JFLEG for better GEC performance. The model will assure precise grammatical correction during real-time conversations. This feature seeks to increase the efficiency of GEC capability.

1.2.2 Objectives

- To develop an interactive conversational AI with sophisticated GEC competency
- To fine-tune with the benchmark GEC dataset
- To implement a modern GEC technique such as seq2seq GEC to enhance performance

1.2.3 Motivation

Artificial intelligence breakthroughs have had a tremendous impact on education. These advantages encompass better learning experiences, active and self-directed learning promotion, and increased accessibility [Yoon et al. \(n.d.\)](#). The GEC task has received increasing attention in recent years [Mani et al. \(2020\)](#). Chatbot-based solutions have captured the curiosity of language learners because they allow them to interact with them in the language they are learning, and they have the potential to develop social interaction and communication competencies [Behzad & Washington \(2024\)](#); [Tyen et al. \(2022\)](#). EnglishBot [Ruan et al. \(2021\)](#) had a greater influence on student fluency than typical listen-and-repeat systems. These chatbots have the potential to improve English speaking and grammatical abilities while also reducing anxiety about learning English.

Generative LLMs provide language assistance to non-native speakers [J. Li et al. \(2022\)](#). Second language learners make up a substantial and growing proportion of the global population. There is an increasing demand for technologies that ease

language learning and instruction. The creation of large language models (LLMs) has had a tremendous influence on the areas of natural language processing, and these improvements have great potential in the field of educational technology. While the potential for big language models in education appears to be promising, further research into their impact and limits is required.

GEC has real-world applications in not only language acquisition, but also text proofreading [Maity et al. \(n.d.\)](#). In the field of writing assistance, and text post-processing for automated speech recognition the necessity of GEC system is unneglectable [Leng et al. \(n.d.\)](#). Such systems have the ability to assist hundreds of millions of people globally [Ng et al. \(n.d.\)](#).

1.2.4 Significance

Generative LLMs have the ability to provide essential linguistic assistance to non-native speakers and have tremendous potential in educational technology. A chatbot-based GEC system would not only be useful language learning, but also in proofreading, content editing, post-processing activities, writing assistance, and automated speech recognition. The significance of this project stems from its capability to improve language assistance by incorporating advanced GEC technique into a conversational AI system. Using LLaMA 2 and benchmark GEC datasets such as JFLEG, the system will give real-time, grammar corrections, boosting correct grammatical sense for users. This merging of GEC approach into a conversational LLM places the project at the crossroads of educational technology where AI-powered language assistance provides increasing grammar efficiency.

1.3 Project Specification

This project is about a conversational AI model with enhanced grammatical error correction capability. The model is developed by employing the Llama-2-chat model as the base model, with which the seq2seq GEC approach is implemented. Therefore, the model is able to solve grammatical error correction tasks and correct several types of grammatical errors inputted with the prompt efficiently. Types of under-correction errors such as subject-verb-agreement, tense, preposition, article, and punctuation errors can be repaired by the model. Users can input texts (sentences/paragraphs) and prompts to fix grammatical errors. The model will then generate output text (sentence/paragraph) that will have all the grammatical errors fixed.

1.4 Contribution of The Project

The project's contribution is to include a GEC approach in a conversational AI model. The conversational AI model utilized in this research is LLAMA-2-7B chat, and the GEC approach integrated as the contribution of the project is sequence-to-sequence approach.

1.5 Report Layout

Chapter 1: Introduction

This chapter covers the problem statement, objectives, aims, and motivation for developing a conversational AI system with improved GEC. It establishes the context for the remainder of the report by describing the project's relevance and overall contribution to the area of generative AI.

Chapter 2: Literature Review

This chapter dives into the reasons for selecting the project and the motivations for its creation. It emphasizes the existing limitations and challenges in GEC tasks, particularly in prior research and projects, highlighting the need for more advancement and creativity.

Chapter 3: Methodology

This chapter discusses the intricacies of system development model and system design and modeling, as well as implementation requirements. The focus is on the activities and elements required for the system's effective implementation. This section visualizes system interactions and architecture using diagrams such as use case, sequence, state, and class diagrams.

Chapter 4: Design and Development

This chapter examines the development processes for both the front-end and back-end designs. It also discusses how the Gradio library is used for the front end and Python for the back end, focusing on the system's desired functional and aesthetic layout.

Chapter 5: Result Analysis

This section covers monitoring of the GEC model performance in TensorBoard, based on a few key performance indicators: such as the progress of the model at which epoch, how the learning rate has changed, and loss over time. The extensive training of the model at 550 epochs is explored in depth, discussing the decaying learning rate and the constant decrease in training loss over time, hence model improvement.

Chapter 6: Conclusion

This chapter covers the project's larger societal implications, ethical, and environmental problems. The long-term sustainability of the system is investigated, and the project's overall contributions are summarized. The chapter finishes with future development goals and potential improvements to the conversational AI and GEC system.

Chapter 2

Literature Review

2.1 Preliminaries

In this section, we have discussed the following topics related with the context of making a chatbot based grammatical error correction system.

2.1.1 Generative AI

Generative artificial intelligence (AI) is now recognized as a key research area, which is influencing disciplines such as natural language processing (NLP), computer vision, and even artistic endeavors [Bandi et al. \(2023\)](#). A generative AI model is a form of machine learning architecture that uses AI algorithms to generate unique data instances based on the patterns and correlations discovered in training data. These types of models are usually fragmental and require additional fine-tuning to specific tasks [Feuerriegel et al. \(2024\)](#). The potential to produce data that closely resembles real-world properties has the promise of helping solve a variety of problems, including data augmentation, anomaly detection, and creative content development.

2.1.2 LLM

Large language models (LLMs) are foundation models trained on enormous quantities of data, which enable them to comprehend and compose natural language and other types of content for a range of applications [IBM \(n.d.-a\)](#). LLMs can translate between languages, learn from context, provide logical and contextually appropriate remarks, summarize content, reply to questions, and even assist with

literary creation or programming activities. LLMs are a significant improvement in NLP and artificial intelligence, and they are easily accessible to the general public using interfaces such as Chat GPT-3 and GPT-4 by Open AI. Other examples are Llama models by Meta and bidirectional encoder representations of transformers (BERT/RobERTa) and PaLM models by Google.

2.1.3 Chatbot

The topic of chatbots has always been prominent since people began to wonder, "Can computers think?" [Ömer AYDIN & KARAARSLAN \(2023\)](#). A computer program that employs conversational AI approaches, like NLP, to understand queries from humans and provide answers automatically is known as a modern AI-equipped chatbot. Where conversational AI chatbots are capable of processing user questions and producing responses that seem human, generative AI chatbots can produce exclusive content as their output. Chatbot interfaces using generative AI may identify, condense, translate, forecast, and produce content in response to a user's inquiry [IBM \(n.d.-b\)](#).

2.1.4 ChatGPT

Open AI has trained a conversational model called ChatGPT utilizing Reinforcement Learning from Human Feedback (RLHF), which has the ability to respond to questions, concede mistakes, rebut wrong preconceived notions, and refuse unsuited inquiries due to its dialogue nature. It is the sister model of InstructGPT, and it is trained to respond to instructions comprehensively and to comply with prompts [Introducing ChatGPT — OpenAI \(n.d.\)](#). ChatGPT is fine-tuned using a GPT-3.5 series model that concluded training in the beginning of 2022.

2.1.5 Llama 2

Llama 2 base models are pre-trained foundation models designed to be adjusted for certain use cases. The Llama 2 model family was released in 2022 under a noncommercial license that gives access to research institutes only on an individual basis. It is available as both foundational base models and refined "chat" models.

The majority of cutting-edge LLMs, such as BARD, Claude, and GPT, are massively closed-source models. As a result, access to public AI research has been

severely restricted. Aside from freely sharing its code alongside model weights, the Llama project has been concentrating on bettering the efficiency of smaller models. Llama 2 models are available with 7 billion (7B), 13 billion (13B), or 70 billion parameters (70B). This lets Llama-based models created by the AI community to be locally deployed by smaller businesses, such as startups and research community members, without requiring prohibitively expensive computing time or infrastructure investments. Although Llama 2 has a specific API of its own, it is available from a number of sources. Watsonx offers Llama-2-13B-chat and Llama-2-70B-chat, which are made possible by IBM's collaboration with Hugging Face [IBM \(n.d.-c\)](#).

2.1.6 Fine-Tuning

Fine-tuning is a strategy that involves training a pre-trained model on a much reduced dataset using task-specific information in order to adapt to a new purpose. While a large general dataset is used for training at the beginning, fine-tuning makes the model specialize in some particular task using highly focused data. That way, the process generally requires less computational resources and time than training a model from the model that is already tuned to recognize general patterns and structures in the data [DataCamp \(n.d.\)](#).

2.1.7 Grammatical Error Correction Task

Grammatical error correction, or GEC, is an approach of automatically detecting and repairing textual errors. The task includes correcting orthographic and semantic issues such as mistakes in spelling and word selection errors, as well as mistakes in grammar such as omitting prepositions and inconsistent subject-verb agreement [Bryant et al. \(2023\)](#).

Some common types of errors are discussed here.

Tense/Aspect Errors: In this type of error, the verb form fails to accurately represent the time period of the activity or situation described. For example, using the past tense form to explain continuing acts might cause misunderstanding.

Agreement Errors: In this type of error, the constituents of a sentence do not correspond in number or person. For example, pairing a singular subject with a plural verb ("They was") results in a grammatical mismatch that may confuse readers about the subject's purpose.

Multiple Errors: In this type of error, more than one grammatical error is present in a sentence, adding to the confusion and making it difficult to interpret the intended meaning.

Morphological Errors: Incorrect spelling, plural form abuse, removal of third-person singular ending, and punctuation and capitalization problems are the most common morphological errors.

Punctuation Errors: The erroneous use or absence of punctuation marks, which can impair a sentence's readability and meaning. For example, failure to employ commas might result in run-on sentences or misinterpretations of content.

Preposition Errors: Often a preposition is mistakenly utilized or omitted, changing the connection between words in a phrase. These mistakes might result in poor language and ambiguity about the intended meaning.

Syntax Errors: Syntax is the arranging of words and phrases to produce well-formed sentences. Syntax errors can result in phrases that are hard to comprehend or do not adhere to normal English rules, such as wrong word order.

Determiner Errors: The improper use or removal of determiners, which include articles ("a," "an," "the") or demonstrative words ("this," "that"). These mistakes might result in ambiguous or imprecise allusions, making it hard for readers to comprehend the exact nouns under discussion.

Some types of errors and their corrections are given as examples:

Type	Error	Correction
Tense/ Aspect	I did not went there yesterday.	I did not go there yesterday.
Morphology	He runned away.	He ran away.
Agreement	She don't like you.	She doesn't like you.
Multiple	I reading a book named big bang.	I am reading a book named Big Bang.
Punctuation	Yes I have done that.	Yes, I have done that.
Preposition	I arrived the airport on time.	I arrived at the airport on time.
Syntax	I love play the piano.	I love to play the piano.
Determiner	Is this new dress you bought this week?	Is this the new dress you bought this week?

Table 2.1: Examples of Different Grammatical Error Types and Their Corrections.

2.1.8 Grammatical Error Correction Dataset

A limited number of grammatical error correction benchmark English datasets are popularized due to shared tasks. Some of them are L FCE (First English Certificate) consisting approximately 531 thousand words, NUCLE/CoNLL, Lang8, JFLEG etc.

2.1.8.1 JFLEG

The Johns Hopkins Fluency-Extended GUG corpus (JFLEG) consists of 1,501 sentences (about 28.1k words) divided evenly between development and test sets. The sentences were picked at random from essays authored by L2 English learners of an unknown skill level and revised by crowdsourced annotators on Amazon Mechanical Turk. Every sentence was annotated four times, yielding four sets of concurrent reference annotations; however, modifications were not clearly specified or categorized. The primary novelty of JFLEG is that sentences are adjusted to be fluid instead of minimum grammatical. The primary critiques leveled with JFLEG are that it is significantly smaller than other test sets, that the sentences are included out of context, and it was not revised by expert annotators [Bryant et al. \(2023\)](#).

2.1.9 Sequence-To-Sequence

Recently, sequence-to-sequence models have become one of the cornerstones of GEC. It has an encoder and a decoder as its core components. The former transforms the input sentence into a series of hidden states representing its semantic meaning; therefore, together with the attention mechanism, it can handle long input sentences. The decoder in turn predicts the tokens of the output sequence, based on the hidden states and previously generated tokens. Sequence-to-sequence models have matched very high performance on a wide range of languages and datasets and have therefore become popularly one of the most favorite choices for GEC tasks [J. Li et al. \(2022\)](#).

The integration of seq2seq models into GEC involves the combination of several techniques to make these models even more effective. For instance, recent works propose data synthesis methods to synthesize the training datasets with diverse examples of grammatical errors, along with improved decoding strategies to alleviate problems associated with slow inference speeds. Further, some of them have integrated multi-task learning frameworks that could segregate the detection and

correction phases in order to improve the efficiency of their processing. These are innovations toward balancing the need for accurate corrections with minimizing the risk of generating incorrect outputs. Overall, seq2seq models continue to evolve by leveraging large language models and novel architectures to improve performance in correcting grammatical errors, at the same time addressing inherent limitations such as over-correction and inference speed [W. Li & Wang \(2024\)](#).

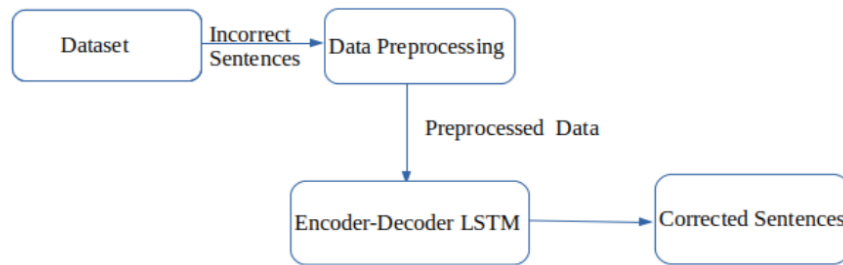


Figure 2.1: Sequence-to-sequence Block Diagram

In this project, for grammatical error correction, we fine-tuned the pre-trained model of Llama-2-7B. It was originally trained on general tasks related to language, and we fine-tuned this model based on a dataset related to out-of-the-box correction of grammatical errors. Fine-tuning the model leveraged its current language capabilities and taught it how to manage the new task efficiently. This approach allowed us to reach high accuracy in sentence correction without having to retrain the model for a long period.

2.2 Systematic Literature Review

2.2.1 Introduction

2.2.1.1 Systematic Review

Systematic reviews are a kind of research-based literature review that stick to rigorous guidelines, similar to primary research, and provide the reader with a clear, understandable goal. They are conducted and reported differently from typical literature reviews and are utilised in research and policymaking to promote evidence-based decisions. Systematic reviews are trustworthy, repeatable, and easily updated. They also adopt transparent and unambiguous procedures and stick to a well-defined, step-by-step methodology. To make sure the evaluation is accurate and beneficial, they also solicit feedback from other people. A systematic review, for instance, needs to have a precisely stated research question and an explanation of the authors' methodology. To answer the review question, the techniques for finding, evaluating, and integrating research are explained; usually, multiple people are involved to ensure objectivity and trustworthiness. Peer-reviewed, published, or registered research methods for systematic reviews can help to reduce duplication and allow for comparisons between the planned and finished studies [Gough et al. \(2020\)](#).

A comprehensive literature review allows scholars to fully interact with and appreciate the selected body of knowledge. It allows them to summarise and critically evaluate existing studies while eliminating bias when selecting research to include. Researchers may eliminate subjectivity and create a review that others can replicate by employing a systematic method for reading and categorizing relevant material, as well as an unbiased review approach. It also helps to maintain an orderly database of the included studies for future reference, identify hidden research gaps using objective data, and write a systematic literature review piece. Furthermore, the findings can be used as instructional aids. Unlike traditional narrative reviews, a systematic review allows researchers to detect gaps in the literature and back up their results with numerical data, making it a form of research in its own right [University of Leeds \(n.d.\)](#).

The abbreviation PRISMA stands for Preferred Reporting Items for Systematic Reviews and Meta-analyses. It is a small set of items based on evidence that must

be disclosed in systematic reviews and meta-analyses. The PRISMA statement includes a four-phase diagram of flow and a 27-item checklist. It can serve as a framework for releasing systematic reviews with a variety of objectives. PRISMA is primarily concerned with the documentation of reviews that assess the impact of interventions. The PRISMA 2020 checklist provides detailed instructions for researchers undertaking systematic reviews. For this aim, the 2009 statement is superseded by the PRISMA 2020 statement, containing new reporting criteria that reflect advances in research discovery, selection, evaluation, and synthesis. To facilitate implementation, the items' layout and look have been modified. The checklist covers all the necessary elements for openly disclosing systematic review procedures and is organized into categories such as Abstract, Introduction, Methods, Results, Discussion, and Other Information. It underlines how crucial it is to provide details about the goals, reasoning, and methodology—including eligibility requirements, search tactics, and data-gathering procedures. To provide a standard and transparent process, the checklist also emphasizes the evaluation of study quality, synthesis techniques, and results presentation. In general, systematic reviews in the research literature are more clear, repeatable and reliable when they follow the PRISMA 2020 criteria [Page et al. \(2021\)](#).

Because of ChatGPT's enormous user base and general popularity, we have decided to use it as our main research approach. Thanks to its capacity to produce text that is both logical and contextually relevant, ChatGPT—which is powered by OpenAI's GPT models—has attracted a lot of attention in a number of fields, including education, content development, customer support, and personal help. Numerous applications that use the model's natural language processing capabilities for tasks like writing aid, conversational bots, and even code generation demonstrate how widely used it is [Rasul et al. \(2023\)](#). Through examining ChatGPT's grammatical error correction (GEC) capabilities, we can make significant contributions to enhancing the functionality of language models that are currently widely employed in practical applications. Furthermore, the study is timely and relevant since enhancing GEC capabilities could greatly enhance the efficiency of widely used AI-driven tools, particularly in educational and professional settings [Farr \(n.d.\)](#).

2.2.1.2 Rationale

The question of the future prospects of the large language models has become one of the hottest topics, especially after the public release of one of the most extensively used Large Language Models, chatGPT [Ray \(2023\)](#). It remains a matter of debate how these models can be utilized effectively as well as accurately in a diverse set of NLP tasks ranging from content generation to virtual assistants. Among varied NLP tasks, English essay writing, English language learning, GEC tasks, summarization tasks can be considered common and most demanding tasks [Zeng et al. \(2024\)](#). Therefore, a systematic literature review on ChatGPT’s GEC capabilities is warranted, considering the great popularity and requirements for the English Learning task. Nevertheless, when these models are being used more and more extensively, it is essential to evaluate their advantages and shortcomings. Existing study indicates that ChatGPT and similar models are incredibly good at generating language, but they can have trouble correcting grammatically erroneous text at any time. As ChatGPT models are increasingly used in real world applications where precise language generation is critical, it is essential to identify and resolve GEC issues in order to improve their reliability and performance. By methodically examining and combining the existing literature on GEC concerns of the popular LLMs of today, especially ChatGPT, this review seeks to bridge this gap. Understanding the intricate details of ChatGPT’s performance in GEC brings out the necessity of this review.

2.2.1.3 Objectives

The aim of this systematic literature review is to present a broad outline of the academic literature on the current problems with ChatGPT’s grammatical error correction while carrying out GEC activities. Our goal is to review existing research on the specific issues and limitations related to the performance of ChatGPT’s GEC. We aim to provide an in-depth review of the current GEC status of ChatGPT and to gain valuable insights for researchers, developers and practitioners in the NLP and AI fields.

2.2.1.4 Research Questions

This systematic literature review will attempt to investigate the current GEC complications of ChatGPT/ Gen AI LLMs and to identify the methods that such models employ for the GEC job. The GEC methodologies that are now in use as well as the metrics that are being used to assess how well the GEC tasks are doing will be covered in this overview. The systematic review is shaped by the subsequent series of research questions:

1. What is the current performance of ChatGPT/ Gen AI LLMs in English GEC tasks
2. What techniques do ChatGPT and Llama models use for grammatical error correction?
3. What are the current GEC approaches and which is the best one?
4. What evaluation metrics are used to check the GEC capabilities of the ChatGPT model?

2.2.2 Methods

The literature review was done following the above research questions (discussed in the introduction section) and the protocols mentioned in The Prisma 2020 statement.

2.2.2.1 Eligibility Criteria

According to the PRISMA 2020 checklist eligibility criteria are the crucial rules that specify which research gets included or excluded in systematic reviews. Researchers guarantee the relevance and uniformity of the included studies by precisely defining these factors, which improves the accuracy and openness of the systematic review procedure. With this clear explanation, readers may evaluate if the studies are acceptable for answering the study question or goal.

The eligibility criteria for our literature review is given below:

Inclusion criteria:

- Published papers that examined the efficiency of ChatGPT or other Gen AI LLMs in the area of English GEC tasks.
- Publications that are released between 2020 and 2024 to guarantee the most recent findings.
- Studies comparing the performance of ChatGPT or other Gen AI LLMs with other AI writing assistants or traditional writing assistant tools for GEC tasks.
- Published papers discussing the techniques used by ChatGPT, Llama, or other Gen AI LLMs for GEC tasks.
- Published papers discussing and comparing the modern techniques for GEC.

Exclusion criteria:

- Papers that are not published in renowned academic publications, peer-reviewed journals, or conference proceedings.
- Studies that do not address ChatGPT's or other Gen AI LLMs capabilities while performing GEC tasks.
- Articles released prior to 2020 should be kept current and emphasize the latest advancements of ChatGPT or other Gen AI LLMs in writing support.
- Studies discussing the grammatical error corrections of ChatGPT or other Gen AI LLMs in languages except English.
- Research that does not mention the strengths and weaknesses of modern GEC techniques
- Papers that lack thorough evaluation or clarity
- Studies without a detailed report on results or outcome

2.2.2.2 Information Sources

The significance of openly disclosing information sources in systematic reviews is stated in the PRISMA 2020 checklist. Readers can assess the thoroughness of the literature review if the information sources are made clear to them. The records for our literature review were identified from six databases: nature, arXiv, Science Direct, Core, Semantic Scholar, and Springer. The last source was searched on 15/05/24. We have limited the search from 2020-2024 in order to select the papers containing the current information on the topic.

2.2.2.3 Search Strategy

The search was led on three databases namely: Nature, Science Direct, Springer, ArXiv, CORE, and Semantic Scholar. In the beginning, we decided to search for the current limitations of ChatGPT using the keywords “ChatGPT AND limitations”, and “Future works on ChatGPT” on the above-mentioned databases. Boolean operators (AND, OR) were applied to connect the keywords, resulting in a more focused study of the research environment. After gaining knowledge on several limitations of ChatGPT we decided that we will work on the English grammatical error correction issues of ChatGPT or other Gen AI LLMs. To find all the relevant papers regarding the GEC issues of ChatGPT or other Gen AI LLMs we used the following keywords: “GEC issues of ChatGPT AND Grammarly”, and “English grammar in ChatGPT OR Gen AI LLMs” on the same databases used before. After retrieving all the relevant papers we could find, we moved on to search the current strategies used by ChatGPT and Llama models for their GEC tasks using the keywords “Methods of ChatGPT for GEC tasks AND Llama”, and “Methods of Llama for GEC tasks”. After finding out the relevant answers we searched for the current GEC techniques used and which one gives better results. For this, we used the keywords “Current GEC approaches”, “Algorithms for Grammatical Error Correction”, “Sequence-to-sequence”, “Sequence-sequence”, and “Unsupervised GEC methods”. Synonymous terms such as “Modern GEC methods” were also used to search for better results. Publication dates from 2020 to 2024 are applied as filters for these searches. The search was limited to peer-reviewed and published papers.

2.2.2.4 Selection Process

At first, we decided on six widely used databases: Nature, Science Direct, Springer, ArXiv, Core, and Semantic Scholar to search for our papers. We then developed a set of inclusion-exclusion standards in accordance with our study questions. According to the Prisma flow chart given in the figure, initially, we found 280 papers which were then screened by reading the abstract and titles according to our research questions. After screening we were left with 147 papers. Each paper was extensively studied to verify that it fulfilled the review's inclusion and exclusion criteria. In the end, 16 papers were chosen for our literature review. Three reviewers were involved in the study selection process. The reviewers worked independently in selecting the published papers according to the inclusion criteria. Each record was screened by every member of the group for eligibility. No automation tools were used to select the papers.

2.2.2.5 Data Collection Process

Each report was read by three independent reviewers, who worked simultaneously to improve the dependability of the data collection process. All reviewers collected data independently from the reports, and any inconsistencies or conflicts were handled by discussion or contact with the supervisor, as needed. To ensure an in-depth and detailed comprehension of the information inside the reports, the whole data collection procedure was carried out manually, without the use of automated tools.

2.2.3 Results

2.2.3.1 Study Selection

The below figure summarises the flow diagram that depicts how we chose the evidence sources. During the identification stage, our search returned a total of 280 records within the databases we had chosen, and no more records could be found. We manually eliminated the duplicates. After removing duplicates, 225 research articles were left. After screening, 110 of these entries were discarded, and 145 journal articles were left for the eligibility evaluation. A total of 130 publications were eliminated for various reasons, including research that mainly concentrated on models other than ChatGPT and the lack of full text availability. Ultimately, 15 research articles were found to be eligible for inclusion.

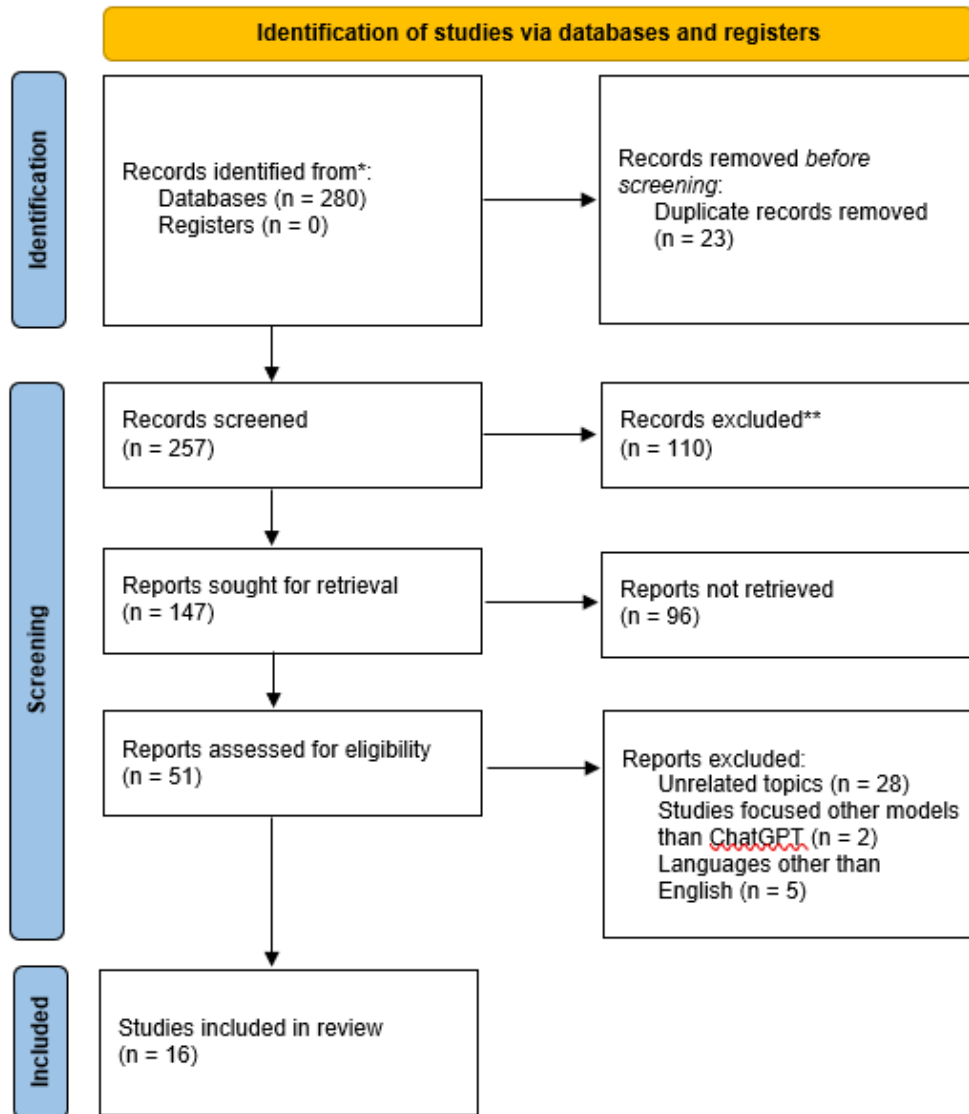


Figure 2.2: A flow diagram summary of the result of the search and selection process

2.2.3.2 Study Characteristics

Up to the time of writing on October 15, 2024, the search was limited to records published between 2020 and 2024. They were all gathered from reputable journals.

2.2.3.3 Result of Individual Studies

According to the study [Wu et al. \(2023\)](#), ChatGPT overcorrects sentences and performs poorly on long sentences. ChatGPT's capacity to fix grammatical errors was assessed using the CONLL-2014 benchmark dataset. After further examination, Grammarly gets the greatest F0.5 score and strikes a better balance between recall and precision than ChatGPT and GECToR. According to these findings, ChatGPT tries to fix errors as much as possible, which could cause it to alter sentence structures and cause overcorrection problems. One of the paper's shortcomings is that it only evaluated 100 sentences from a single dataset, and its evaluation measures were limited to F0.5, precision, and recall.

The authors of the article [Coyne et al. \(n.d.\)](#) have conducted tests on major GEC benchmarks, including the BEA-2019 and JFLEG benchmark datasets, to assess the capabilities of a GPT-3.5 model and a GPT-4 model. The findings indicate instances of over-editing and under-editing in both GPT-3.5 and GPT-4. In addition, punctuation problems are displayed by the GPT models. GLEU and F0.5 scores from the JFLEG and BEA-2019 datasets were used for performance evaluation. The drawbacks of the paper include a narrow focus on prompts without investigating strategies such as chain-of-thought prompting, and time and financial limits that impact both metric scores and human annotation studies.

Studies conducted on renowned GEC test sets, such as CoNLL14 and BEA19, show that ChatGPT frequently exhibits tendencies towards both overcorrection and under-correction [Fang et al. \(2023\)](#). Additionally, it demonstrates good fluency and error detection, but in document-level Grammatical Error Correction (GEC), it struggles with tense errors, coreference, agreement, and also cross-sentence boundary errors. The evaluation metrics in the study include precision, F0.5 scores, fluency, and human evaluations to assess ChatGPT's performance in Grammatical Error Correction. The paper points out a drawback of concentrating only on ChatGPT without looking at other popular language models and emphasizes the model's

usefulness in real-world contexts like language instruction. It also makes recommendations for future research directions on minimal editing corrections.

The paper [Younis et al. \(2023\)](#) compares ChatGPT's capabilities to writing tools like Grammarly and Quillbot, with a focus on grammar, style correction, plagiarism detection, and general writing enhancement. The process includes a thorough analysis of ten unique situations, covering a range of writing-related topics. The findings demonstrate ChatGPT's effectiveness in accomplishing writing objectives, especially in content creation, but they also point out ChatGPT's lack of accuracy and comprehensiveness of grammar and style advice compared to Grammarly.

The authors [Bryant et al. \(2023\)](#) examine a number of methods, such as edit-based methods, SMT, NMT, and rule-based classifiers. They call attention to the shortcomings of local context dependency and the requirement for large amounts of parallel data, as well as the limits of classifiers and SMT. It is detailed how NMT came to be, emphasizing its cutting-edge performance and ease of use. The study explores edit-based methods, highlighting its openness and quicker rate of inference while admitting the necessity of human engineering. Additionally, the authors present Language Model (LM)-based methods. With the use of the pre-trained models such as GPT-2, GPT-3, OPT, and PaLM, which have been trained on enormous datasets and refined for zero-shot or few-shot correction tasks, recent Language Model (LM)-based approaches in Grammatical Error Correction (GEC) allow the advancement of multilingual GEC systems without the need for explicit language training. The paper also points out that these Large Language Models fail on datasets intended for minimum corrections because they overcorrect for fluency, even if they are capable of multilingual GEC. Evaluation procedures are explained, with a focus on metric alignment with particular datasets, and the significance of model ensembling for enhanced performance is mentioned. In conclusion, future difficulties are discussed, such as contextual GEC, customized systems, domain generalization, and the possibility of integrating several methodologies for improved corrective capabilities.

The authors [Petiška \(2023\)](#) conducted an elaborate analysis of significant preprints to evaluate the many applications and problems of ChatGPT. The process involved dividing up the preprints according to Altmetric attention scores, applying a relevance filter, and asking ChatGPT (Model: GPT-4) to produce an organized review article. The findings offered promising information, pointing to ChatGPT's possible

use in producing review articles. The research papers that were examined addressed a variety of subjects, including the financial effects of artificial intelligence (AI) language modeling, public opinion forecasting, grammatical error correction, translating texts, and producing philosophical writings. The results emphasized the potential advantages of ChatGPT across a range of fields, but they also highlighted its drawbacks and challenges, such as overcorrection problems with grammatical error correction, differences in translation performance, and the requirement for more research and development.

The study [Katinskaia & Yangarber \(2024\)](#) examines how well GPT-3.5 performs in GEC for various languages, including its flaws, constraints, datasets, and methods. GPT-3.5 encounters several difficulties in GEC, including overcorrection, changes in sentence semantics, and challenges with punctuation, tense, syntactic dependencies, and lexical compatibility issues, especially in Spanish, Russian, Czech, Ukrainian, and German. While GPT-3.5 shows good memory and fluency in English, it still needs help with articles, prepositions, punctuation, and verb tense issues. Using OpenAI's GPT-3.5-turbo models via API, the evaluation technique comprises three experimental settings: zero-shot GEC, fine-tuning, and re-ranking hypotheses produced by other GEC models. English prompts were used throughout sentence-level experiments to maintain uniformity across all languages. SweLL-gold was one of the datasets used for assessment and fine-tuning; owing to GDPR protection, 60% of the phrases were sampled and made anonymous. The study points out some limitations, such as the inability to conduct fine-tuning experiments due to budgetary restrictions, the reliance on a restricted amount of prompt messages, and the requirement for new assessment measures that effectively represent the model's performance. Though it has great promise, GPT-3.5's effectiveness varies significantly by language and frequently modifies input words more freely, making performance measurement with standard reference-based measures more difficult. According to manual evaluations, many errors continued, especially in the English and Russian sentences corrected by GPT-3.5 regarding using hyphens, prepositions, articles, tense, and lexical compatibility and punctuation, respectively. However, 86.5% of English and 80% of Russian writings were admissible. Higher semantic similarity was attained by reranking techniques while preserving good recall, indicating the possibility of integrating them with more compact, refined GEC models

According to the research done in [Zhou et al. \(2023b\)](#), modern GEC systems follow one of the three main approaches. First, the sequence-to-sequence (seq2seq) method uses encoder-decoder models like Transformer to perform corrections, treating GEC similarly to a monolingual machine translation assignment. Secondly, source tokens or erroneous sentences are corrected by applying the sequence-to-edit (seq2edit) technique, which predicts a series of edits to be applied to a sentence to make it correct. Finally, by prompting or fine-tuning, LLMs such as ChatGPT and Llama use their capabilities for GEC. Llama is fine-tuned for Grammatical Error Correction (GEC) activities and ChatGPT uses a zero-shot prompting strategy which implies that it can generate responses in GEC scenarios without explicit GEC data training.

The study [Algaraady & Mahyoob \(2023\)](#) compares the efficiency of ChatGPT with human instructors in detecting grammatical faults in the writing of English as a Foreign Language (EFL) learners. LLMs like ChatGPT are good at identifying surface-level faults, but they are less efficient at identifying more subtle problems relating to the deep structural and practical components of writing. The study offers insightful information about the possible advantages and drawbacks of using ChatGPT in EFL writing instruction. The tool was unable to correctly identify writing tenses in some cases. A notable F-score of 1.5 and a p-value of 0.23 are revealed by the statistical analysis, demonstrating the validity of the data that was gathered. Further analysis concluded that ChatGPT is good at deciphering basic structures, but it has trouble understanding the deeper meaning of mistakes made.

According to [Sutter & Carvalho \(2023\)](#) the prompt-based GEC strategy that uses LLMs has grown in popularity because of its simplicity of application, which relies on calls to the API without the requirement for training, and no huge amount of data is needed which is less costly. However, this technique has limitations, for example, ChatGPT overcorrects and provides corrections that are not consistent with the test dataset. The study also discusses supervised GEC approaches, such as sequence-sequence and sequence-tagging, and how they perform better than unsupervised methods.

As explained in the paper [J. Li et al. \(2022\)](#), GEC can be seen as some sort of machine translation, where the source language consists of erroneous sentences and the target language consists of corrected sentences. Recently, solutions to the GEC task have been proposed through NMT-based approaches, which have attracted

attention. These are models similar to those proposed by Bahdanau, Cho, and Bengio in 2015. Such models are a sequence-to-sequence structure whose inputs are sentences with grammatical mistakes and whose outputs are the correct sentences. Of these, the Transformer model, first introduced by Vaswani in 2017 and then adapted for GEC by Bryant in 2019, has become one of the leading choices of NMT-based GEC approaches.

In other studies made by [Zhou et al. \(2023a\)](#), grammatical error correction can be a very useful task in NLP, as it has practical applications in document proofreading, text completion in writing, language learning, and enhancement of text generated by automatic speech recognition systems. There are two general directions in which GEC can proceed: Seq2Seq approaches, such as those by [Sun et al. \(2021\)](#); [Rothe et al. \(2021\)](#), and Seq2Edit approaches, such as those by Awasthi and Omelianchuk. While Seq2Seq treats GEC as a genre of translation within the same language, Seq2Edit considers GEC as labeling the changes to the text. For instance, it has been shown through experiments that Seq2Seq mostly outperforms Seq2Edit, especially in the correction of more complicated errors involving word order. In this regard, the Seq2Seq technique will be quite effective in handling a wider range of grammatical errors [Zhang & Goyal \(2022\)](#).

According to research by [Qorib et al. \(2022\)](#), GEC is the task of finding and correcting grammar errors in an input text. Due to the development of Seq2Seq models, great improvement has been achieved in GEC starting from the end of the 2000s. Competitions such as CoNLL-2013, CoNLL-2014, and BEA-2019 have also livened up attention to this task. The Transformer model of Vaswani et al. has achieved such astonishing success in Seq2Seq tasks that in the current top-performing GEC systems, it has made the Transformer architecture the one most in use. Most of these systems are based on Transformer-based models, but all of them differ in how to address the task and in how they pre-train their models on various data. This work turned out to be very effective for GEC, especially the Seq2Seq method with Transformer models because it handles different types of grammatical errors in a very structured way.

The study [Sottana et al. \(2023\)](#) deals with the inconsistency in the assessment of LLMs by concentrating on enhancing understanding of LLMs' performance, notably ChatGPT, on Grammatical Error Correction (GEC). It uses both automatic and human evaluation to find the results. Using the BEA-2019 dataset and F0.5

scores it came to the conclusion that ChatGPT tends to overcorrect a lot by adding new information to a phrase that was not initially present.

In the study [Caines et al. \(2023\)](#) it is talked about how essential it is to identify and fix grammatical errors in language learning applications. Traditional methods include training computer models with particular datasets. New prompting approaches such as zero-shot and few-shot in ChatGPT show promising results but also have limitations. For example, ChatGPT occasionally overcorrects, modifying phrases more than necessary. Additionally, they discovered under-correction in a few places in the reference phrase. The dataset used for the evaluations includes CoNLL-2014, JFLEG, and BEA-2019.

The study [Davis et al. \(2024\)](#) looks at the efficacy of LLMs like ChatGPT in fixing grammatical problems in English texts. The study concludes that supervised models continue to perform better than LLMs for some datasets that emphasize minimum modifications. LLMs have a preference for rewrites, making it difficult to encourage minimum editing. The findings demonstrate that supervised models continue to be the better options for minimal edit fixes. F0.5 and GLEU were used to evaluate the GEC issues of ChatGPT. The performance of their model was compared on four renowned GEC datasets: CoNLL-14, JFLEG Dev, FCE Dev, and W&I+LOCNESS Dev.

2.2.4 Findings

R.Q.1: Performance of ChatGPT in English GEC tasks Grammar is considered one of the most pivotal branches of language learning. It is the foundation for speaking, understanding, writing, and overall communication. Correct grammar makes the communication clear, precise, and effective. For any piece of writing, whether it is an essay, an article, a research paper, or an assignment, accurate grammar is essential to interpret the exact meaning without distorting it. That is why our second question attempts to investigate whether the super-hyped chatbot can effectively detect and correct grammatical mistakes in writing or not. With the aim of improving students' grammar, the use of AI based tools and computer technologies has been commonly practiced in EFL literature [Alzahrani & Alotaibi \(2024\)](#). According to [Božić & Poola \(2023\)](#), AI chatbots, such as ChatGPT, may mimic real-world interactions offering students immediate feedback on their vocabulary, grammar, and pronunciation. Table 2.1 notes the strengths of ChatGPT in Grammatical Error Correction tasks based on the collected research articles.

Table 2.2 A summary of performance strengths of ChatGPT in English GEC tasks

Domain	Description	Source
Fluency corrections	ChatGPT demonstrates good fluency.	Katinskaia & Yangarber (2024), Coyne et al. (n.d.), Fang et al. (2023), Davis et al. (2024)
Error detection	ChatGPT corrects as many errors as possible and is good in error detection.	Wu et al. (2023), Fang et al. (2023), Davis et al. (2024))
Comparison with other GEC systems	ChatGPT has the least number of under-corrections when compared with Grammarly and GECToR. Fewer mis-corrections were found compared with GECToR.	Wu et al. (2023)

Table 2.2: Summary of performance strengths of ChatGPT in English GEC tasks.

In a word, ChatGPT has a strong sense of fluency, according to papers Katinskaia & Yangarber (2024), Coyne et al. (n.d.), Fang et al. (2023), Davis et al. (2024) excels at detecting and fixing most errors, and Wu et al. (2023), Fang et al. (2023), Davis et al. (2024). Wu et al. (2023) compares ChatGPT with Grammarly and GECToR, claiming ChatGPT has the fewest under-corrections, highlighting its efficacy in GEC tasks.

Apart from fluency and reasonable error detection rates, a range of performance concerns have been discussed in many publications. Table 2.2 lists ChatGPT's performance weaknesses in GEC tasks based on the gathered research publications.

Table 2.3 A summary of performance weaknesses of ChatGPT in English GEC tasks

Domain	Description	Source
Over-correction issues	ChatGPT tries to fix errors as much as possible, which could cause it to alter sentence structures and cause overcorrection problems.	Wu et al. (2023), Coyne et al. (n.d.), Fang et al. (2023), Bryant et al. (2023), Petiška (2023), Sottana et al. (2023), Caines et al. (2023)
Under-editing/Mis-correction issues	ChatGPT frequently exhibits tendencies towards under-correction and miscorrection issues.	Coyne et al. (n.d.), Fang et al. (2023), Caines et al. (2023)
Punctuation issues	Punctuation problems are displayed by the GPT models.	Katinskaia & Yangarber (2024), Coyne et al. (n.d.)
Subject-verb agreement issues	In document-level Grammatical Error Correction (GEC), ChatGPT struggles with tense, coreference, agreement, and cross-sentence boundary errors.	Katinskaia & Yangarber (2024), Fang et al. (2023)
Tense errors	ChatGPT was unable to correctly identify writing tenses in some cases.	Katinskaia & Yangarber (2024), Coyne et al. (n.d.), Algaraady & Mahyoob (2023)
Preposition errors	While GPT-3.5 shows good memory and fluency in English, it still has trouble with articles, prepositions, punctuation, and verb tense issues.	Katinskaia & Yangarber (2024)
Comparison with other GEC systems	ChatGPT produces more over-corrections than other GEC systems such as Grammarly and GECToR.	Wu et al. (2023)

Table 2.3: Summary of performance weaknesses of ChatGPT in English GEC tasks.

Overcorrection issues of ChatGPT are evident in 7 papers Wu et al. (2023), Coyne et al. (n.d.), Fang et al. (2023), Bryant et al. (2023), Petiška (2023), Sottana et al. (2023), Caines et al. (2023), which also leads to changes in sentence structure. Under-correction and mis-correction issues are also prominent in papers Coyne et al. (n.d.), Fang et al. (2023), Caines et al. (2023), whereas only two papers Katinskaia & Yangarber (2024), Coyne et al. (n.d.) discuss punctuation errors, and Katinskaia & Yangarber (2024), Fang et al. (2023) discuss subject-verb agreement errors. Tense errors are mentioned in 3 papers Katinskaia & Yangarber (2024), Coyne et al. (n.d.), Algaraady

& Mahyoob (2023), and the result of comparison with Grammarly and GECToR shows the poorer performance of ChatGPT.

R.Q.2: What techniques do ChatGPT and Llama models use for grammatical error correction?

From the papers we nominated for our research, only five papers Bryant et al. (2023), Zhang & Goyal (2022), Sutter & Carvalho (2023), Sottana et al. (2023), and Davis et al. (2024) noted the answer to our second research question. As we are trying to identify the methods and approaches chatGPT and Llama model have implemented dedicatedly for grammatical error correction tasks, we observed not too many papers headlined or specially presented the currently implemented approaches.

Table 2.4 Techniques do ChatGPT and Llama models use for grammatical error correction

Domain	Description	Source
Unsupervised technique	ChatGPT uses unsupervised/ zero-shot or few-shot strategies for GEC tasks.	Bryant et al. (2023), Zhang & Goyal (2022), Sutter & Carvalho (2023), Sottana et al. (2023), Davis et al. (2024)
Fine-tuned technique	Llama 2 is fine-tuned for GEC tasks.	Zhang & Goyal (2022)

Table 2.4: Techniques do ChatGPT and Llama models use for grammatical error correction.

The above table indicates that ChatGPT and Llama models use unsupervised methods for GEC tasks where ChatGPT uses zero-shot/few-shot strategies and Llama model is fine-tuned for GEC tasks. These are discussed in some of the papers mentioned above.

The unsupervised/ zero-shot or few-shot strategies adapted by the ChatGPT have been publicized in three papers whereas the fine-tuning method applied in Llama 2 is articulated in only one paper from our search. These LLMs use unsupervised methods since gathering datasets is a costly operation, and several languages have vast volumes of raw data available.

R.Q.3: What are the current GEC approaches and which is the best one? What are the GEC approaches used in most used GEC systems?

Five publications from the papers we chose for our research have the answer to our third research question highlighted. At this stage, our goal is to identify which GEC approaches are in use right now and which one is the best overall. Sequence-to-sequence Sequence-Edit, Sequence-tagging, Prompting, and Fine-Tuning are noteworthy in this instance and are reported significantly in the mentioned publications as well.

Table 2.5: Current Used GEC approaches

Domain	Description	Source
Sequence-Sequence (Supervised method)	Sequence-to-sequence is one of the most popular GEC techniques in recent times.	Zhang et al. (2023), Sutter & Carvalho (2023), J. Li et al. (2022), Zhou et al. (2023b), Zhang & Goyal (2022), Qorib et al. (2022)
Sequence tagging (Supervised method)	Sequence tagging is gaining popularity.	Sutter & Carvalho (2023)
Prompting (Unsupervised method)	Many models use prompting for training a model with GEC capabilities.	Bryant et al. (2023), Zhang et al. (2023), Sutter & Carvalho (2023)
Performance of supervised methods	Supervised methods perform better than unsupervised methods.	Sutter & Carvalho (2023)

Table 2.5: Techniques do ChatGPT and Llama models use for grammatical error correction.

According to the above studies there are three most renowned approaches for GEC. These are sequence-sequence, sequence tagging, and LLM-based methods which include prompting or fine-tuning. According to some papers sequence-sequence method is the preferred method for GEC tasks as it provides promising results. Whereas sequence-sequence models generate correct sentences from scratch.

R.Q.4: What evaluation metrics are used to check the GEC capabilities of the ChatGPT model?

The evaluation metrics that are used in terms of testing the GEC task performance of chatGPT is our fourth research question. In search of the answer to our question, four studies were handpicked and those papers listed metrics including F0.5 score, precision, recall, GLEU, and human evaluation.

Table 2.6 Evaluation metrics used to check the GEC capabilities of the ChatGPT model

Domain	Description	Source
F0.5	F0.5 is one of the most popular evaluation metrics for testing GEC capabilities.	Wu et al. (2023), Fang et al. (2023), Algaraady & Mahyoob (2023), Sottana et al. (2023), Davis et al. (2024)
Precision	Precision measures the accuracy of positive predictions.	Wu et al. (2023), Fang et al. (2023), Algaraady & Mahyoob (2023) Recall
Recall measures the ability to identify all relevant positives.	Wu et al. (2023)	
GLEU	GLEU measures n-gram overlap between generated text and references.	Coyne et al. (n.d.), Davis et al. (2024)
Human evaluation	Human evaluation uses people to judge text quality..	Fang et al. (2023), Sottana et al. (2023)

Table 2.6: Evaluation Metrics for GEC tasks

The observation from the table can be summarized by stating that most of the publications used F0.5 score and precision. Amongst four publications, three of them used these two metrics. Other three metrics were used in only one paper. Therefore, the F0.5 score and precision can be considered as the most popular evaluation metrics for GEC tasks performance evaluation.

Chapter 3

Methodology

Methodology is a systematic framework that governs how research or a project is carried out, assuring consistency, precision, and integrity. It provides a defined structure for data collection, analysis, and interpretation, resulting in reliable and consistent outcomes. Researchers can avoid biases and errors by adhering to a well-defined approach, resulting in accurate and reliable findings. It also increases efficiency by identifying specific processes that ensure resources are spent effectively and the project stays on plan. Furthermore, a robust methodology promotes transparency and accountability through facilitating others to understand, analyze, and reproduce the study or initiative. Eventually, it is necessary for decision-making, problem-solving, and determining the validity of findings.

Our project, named GrammarGenius, is a conversation model that corrects grammatical errors by employing the sequence-to-sequence technique utilizing the llama-2-7b-chat model, fine-tuned on the JFLEG dataset. Besides, QLORA is used for efficient training. The sequence-to-sequence technique ensures the generation of grammatically corrected output sentences of the inputted erroneous sentence.

3.1 System Development Model

Generally speaking, system development is the broad process that covers the design, construction, and implementation of a system to meet certain needs in both hardware and software [Grady \(2016\)](#). Normally, this happens along a structured framework often known as the Systems Development Life Cycle (SDLC) with its stages of requirement analysis, system design, programming, testing, and maintenance. Some examples of SDLC include Agile, Rad, Waterfall, etc. Each one of these has sought improvements in both efficiency and responsiveness for building systems that will effectively solve complex problems and respond to real users' needs. We also have some frameworks of Agile including Scrum, and Xd. For the purposes of our project, we will be using the framework Scrum. It is used to manage and complete complicated projects.

3.1.1 Scrum

Scrum is a subset of the Agile project management approach in which teams create projects over small project cycles. These small project cycles are also known as sprints. Basically, the project's life cycle is divided into smaller cycles of particular time periods, which need to be completed separately. This helps meet deadlines and swiftly develop the project. Client suggestions is collected at the conclusion of each sprint, and their ideas are implemented before moving forward with the development process.

The Scrum process employs an iterative and incremental strategy to project management. As the full project is not addressed in one go, it is simpler to make modifications. Scrum aims to make test-driven development techniques quicker, simpler, and more flexible, making it easier to design complicated products. Scrum project management assigns distinct responsibilities, such as product owner, scrum master, and scrum development team. The product owner is in charge of understanding the project requirements and making final choices on what has to be included or removed from the project. The scrum master serves as the project manager, aiding the team in optimizing transparency and delivery flow. Promoting community and harmony is also the role of a scrum master. Finally, there are teams known as scrum development teams that are responsible for the development of the project [What Is Agile Scrum? With Proven Agile Scrum Techniques \(n.d.\)](#).

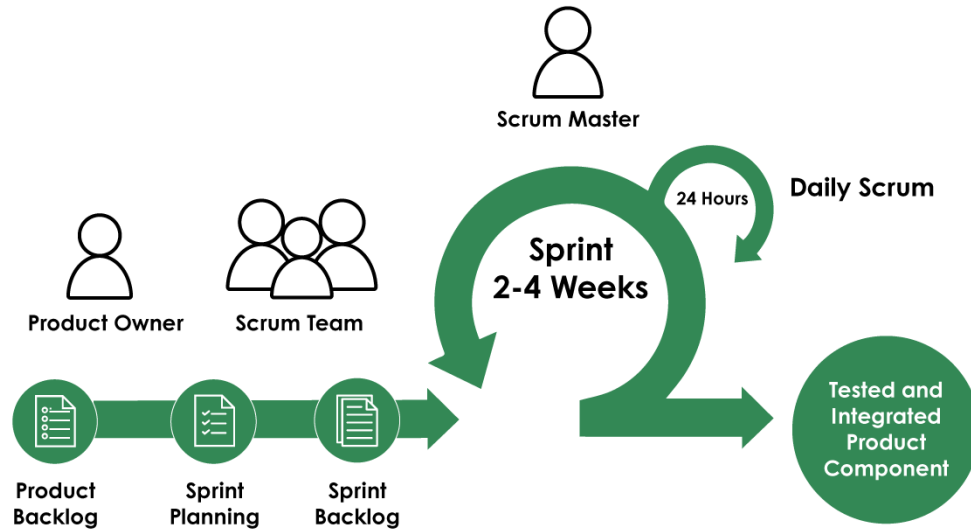


Figure 3.1: System Development Model

3.1.2 Application of Agile Scrum Framework

The project was broken into shorter iteratives, or sprints, using the scrum approach. Every sprint has a distinct objective to achieve. To ensure that the team is on track with the project's objectives, our supervisor assumed the position of product owner. To ensure that the work is performed in line with the sprint plan, the team leaders take on the role of scrum master. Members of the team were developers.

Before each sprint began, its purpose and objectives, as well as the tasks it intended to achieve, were established. Brief but frequent meetings were arranged to talk about the accomplishments, difficulties, and future plans. At the conclusion of each sprint, the progress made was assessed.

Scrum offers a precise structure for decomposing difficult projects. We were able to concentrate on certain tasks throughout each sprint and adapt when new problems or ideas came up. Scrum's iterative nature makes it perfect for projects like ours, where we need to tweak our strategy in response to findings or enhance model performance after testing. Additionally, as our capstone project requires collaboration, Scrum places a strong emphasis on communication and teamwork.

The goal and objective of each sprint and the tasks to accomplish in that sprint was defined before the starting of each sprint. Short but regular meetings were held to discuss progress, challenges, and next steps. The progress of each sprint was evaluated at the end of each sprint.

Scrum provides a clear framework for breaking down the complex tasks. Each sprint allowed us to focus on specific tasks and make adjustments as new issues or ideas arise. The iterative nature of Scrum is ideal for projects like ours, where we need to improve model performance after testing or refine our approach based on results. Furthermore, Scrum emphasizes communication and teamwork, crucial as our capstone project is a group effort. By using Scrum, we could ensure that your project progresses in a structured yet flexible way, allowing us to adapt to challenges while keeping track of your goals.

3.2 System Analysis

System analysis is a problem-solving approach that disassembles a system into its constituent parts to analyze how effectively those components function and interact to achieve their objectives. It is an explicitly formal inquiry undertaken to help decision makers in determining the most appropriate course of action and making more informed decisions [Gharajedaghi \(2011\)](#). The key steps of system analysis are: problem identification, requirement analysis, design, development, testing, deployment, and maintenance.

3.2.1 Problem Identification

The first stage of system analysis is problem identification, where any existing problem or areas of improvement are searched and identified. To do so, a thorough analysis of existing systems or operations is done. The analytics try to find incompetence or shortcomings of a system that blocks the best-case performance of the system.

For this project's problem identification, we have performed research over the internet regarding different issues of generative AI chatbots that are available at the present time. Then, we listed the major problems and selected one to continue with. We identified that the grammatical error correction capabilities of the present generative AI system have significant areas of improvement. To validate the problem, we

performed a systematic literature review and gathered the necessary information on this topic.

3.2.2 Requirement Analysis

After identifying the problem comes requirement analysis. To solve the identified problem, a thorough analysis must be done on what is required to solve the problem. In this stage, stakeholders are identified, and their requirements are noted. How the system is currently being used, what the features should be, how the features should work, etc, are analyzed in this stage.

For this project, the stakeholders are the users of generative AI systems. Millions of users worldwide use chatGPT, LLAMA, and other generative AI tools as their writing assistants for tasks like brainstorming, content generation, paraphrasing, grammar checking and correction, etc. The functional requirement for the project is basically improving a conversational AI chatbot model's GEC capabilities so that it can revise the grammatically wrong sentences prompted by the user and provide corrected sentences. The model should be able to handle a range of grammatical errors such as tense, prepositions, punctuation, article, subject-verb agreement, etc.

There are some technical requirements for carrying out the project. At first, an open-source base chat model is required. In this case, the llama-2-7b-chat model is best suited for use in huggingface. For the coding part, Python and several libraries such as Transformers, PEFT, Torch, and TRL are required to load and work with the llama-2-7b-chat model. Google Colab and Jupyter Notebook should be used to work on the project. For datasets, the JFLEG dataset can be used which contains pairs of incorrect and corrected sentences to train the model on a GEC corpus.

3.2.3 System Design

At this stage the project team designs the system. Different data on the existing system is collected and analyzed, and then a blueprint is crafted containing all the necessary changes and updates that are required to make. A draft of the solution is created and for visual representation different diagrams are designed for clear understanding of the requirements and how the system should work.

3.2.3.1 Use Case Diagram

Use-case diagrams are illustrations of a system's scope and high-level functionality. The major components and procedures that make up the system are shown in use case diagrams. In a use case diagram, the use cases and actors specify whatever the system performs and the way the actors communicate with it, but not how it operates inside. There are four components of the use-case diagram. They include the actors with whom the system interacts, the system by itself, the use cases or features that the system is designed for doing, and the lines that connect these components. Connections can be made between the use cases and between the actors and use cases [Use-Case Diagrams - IBM Documentation](#) (n.d.-a).

Use Case Diagram for GrammaGenius

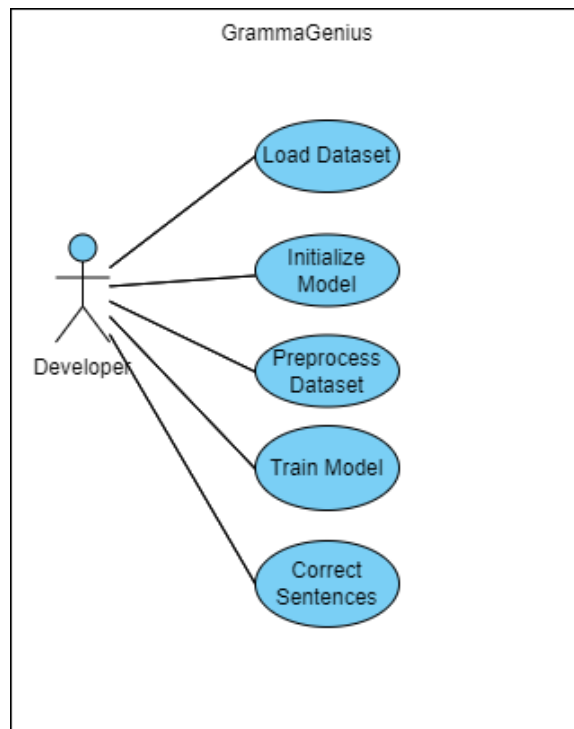


Figure 3.2: Use Case Diagram for GrammaGenius

The diagram represents the main actor interacting with the system designed for grammatical error correction. The developer's role in using the system is to manage the entire workflow, starting with the load dataset use case, where they upload the required dataset for training or evaluation. Next, the developer enables the model by setting up the Llama-2-7B-chat model to be fine-tuned or deployed. The pre-processed dataset use case comes before training, ensuring that the data is properly formatted and cleaned according to the model's requirements. This is followed by the train model use case, where the model is fine-tuned using the pre-processed data. Finally, the correct sentence use case involves applying the trained model to correct grammatical errors in sentences. Overall, this use case diagram outlines the key stages of preparation and fine-tuning for GEC tasks.

3.2.3.2 Activity Diagram

An activity diagram is a type of flowchart that shows how one action in a system or procedure leads to another. They are often used to depict even the most complex systems. They depict each stage of an activity, along with the order whereby they are presented. Activity diagrams describe the process from start to completion. An activity diagram has an activity node, decision node, control flows, start node, and end node to illustrate any system action [MindManager \(n.d.\)](#).

Activity Diagram for GrammaGenius

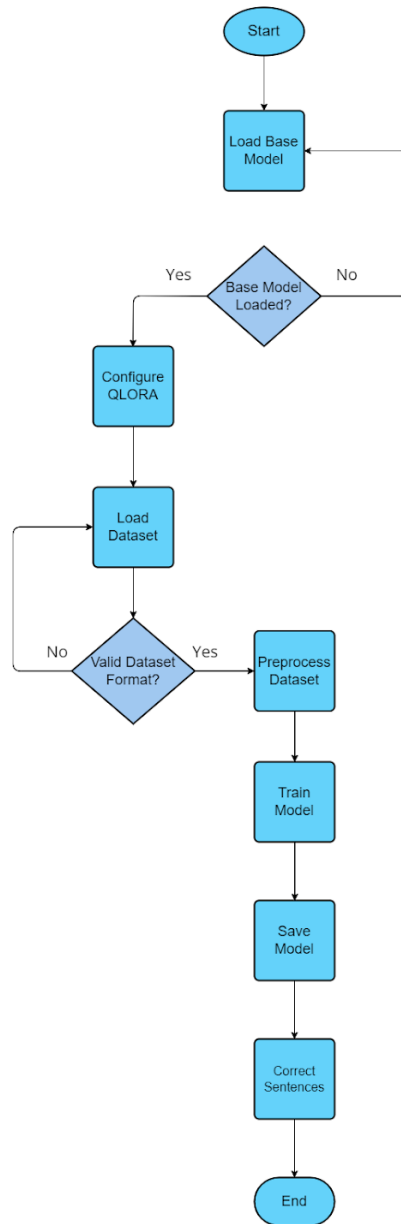


Figure 3.3: Activity Diagram for GrammaGenius

In the above diagram the activity flow of our model is shown. First, we load a pre-trained language model: llama-2-7b-chat. This is the base from which the finetuning takes place. Then, QLORA is initialized with some specific settings, such as how many low-rank adaptation layers, the rank per layer, and how much quantization should be performed. These are the parameters that determine the efficiency and accuracy of the fine-tuning. Then, we load a dataset representative of examples of the target task. In our case, the JFLEG dataset is loaded. Further, verification of the structure and compatibility format is performed to ensure re-use of it. If its format is invalid, the process may be stopped and the dataset needed to be fixed. Next is the dataset preprocessing, which will prepare the data for training. This pre-processing includes the sequence-to-sequence task to perform GEC capabilities in our model. After that, the QLORA algorithm is used to train the model on the preprocessed dataset. In this training phase, the model parameters would be tuned to accomplish better performance for the target task. Then we save the trained model in a huggingface repository for future use. Later on, this saved model can be deployed easily for the performance of GEC. Finally the fine-tuned model is used for sentence grammatical error correction.

3.2.3.3 Sequence Diagram

Sequence diagrams show the interactions between items in chronological sequence. One can use the order of events to determine which objects and actions are required to accomplish the functionality. They are useful tools for starting a project given that they offer the user a step-by-step description of what has to be done. Furthermore, sequence diagrams are important for interacting with clients, illustrating what is going on, and extracting needs [Roubtsova \(2015\)](#). The fundamental goal of a sequence diagram is to show item interactions in the order they occur. Sequence diagrams let an organization's technical personnel record the expected behavior of an approaching system. Architects and developers may use the diagram to drive out item connections in the system, which helps flesh out the overall system architecture throughout the design process [IBM Developer \(n.d.\)](#).

Sequence Diagram for GrammaGenius

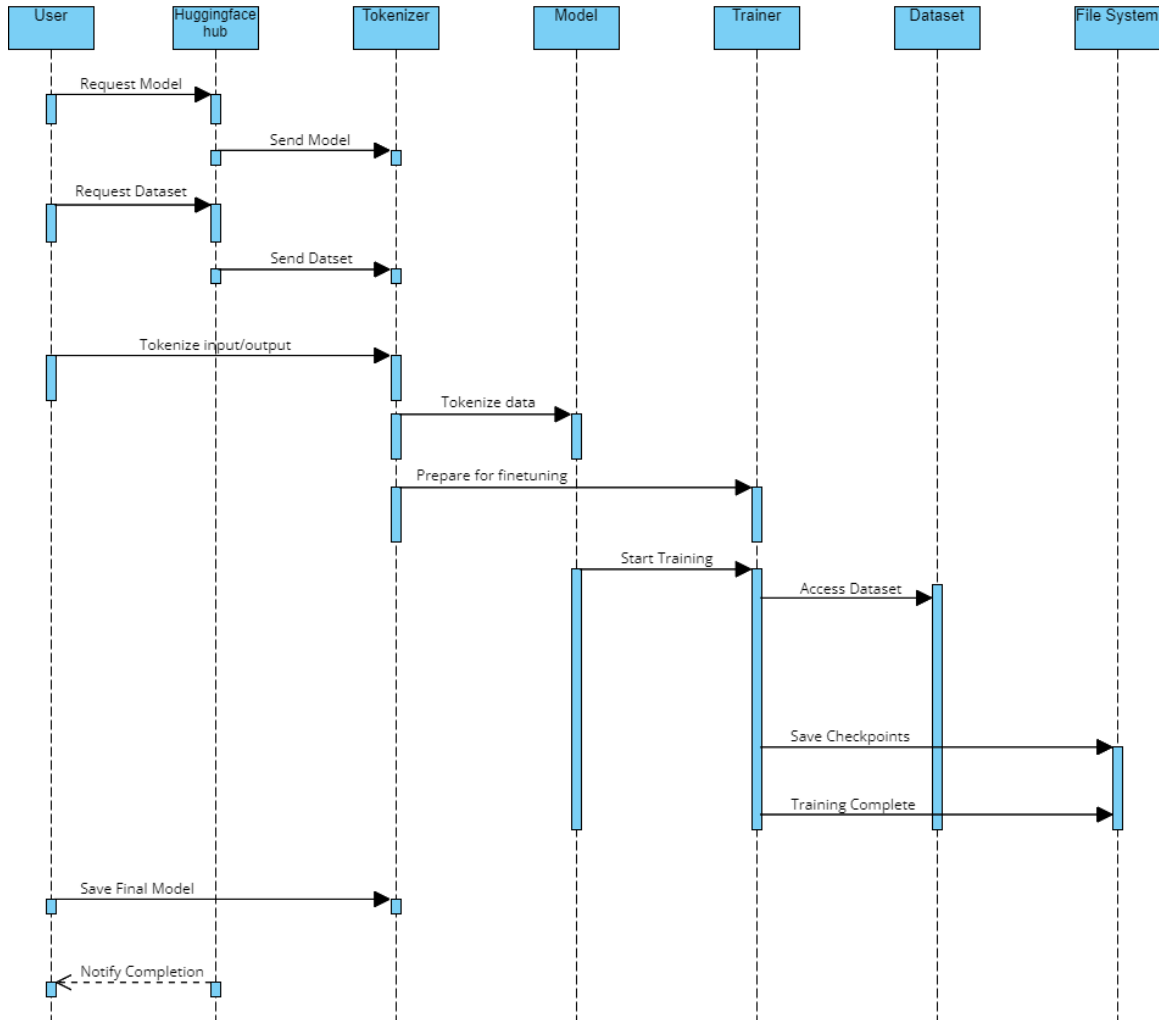


Figure 3.4: Sequence Diagram for GrammaGenius

This is the sequence diagram for the process of fine-tuning a language model on the Hugging Face platform. The user enters a request to get a pre-trained model and a dataset from the Hugging Face Hub. Once the model is retrieved, it is passed on to the tokenizer, where the text is divided into smaller units called tokens. The tokens are very important in that they give the model the ability to comprehend and process the text. Finally, the input data and output are tokenized and prepared for fine-tuning. Training starts whereby the model's parameters are modified in such a way that best realizes performance on the task at hand. The trainer draws upon the dataset for learning from the training examples. Periodically, check-points may be saved so that one keeps track of how far the model has gone toward convergence and may also support early stopping if this is necessary. The last step is to push the final model onto Hugging Face Hub once the training is completed. It then notifies the user that fine-tuning has been completed.

3.2.3.4 Class Diagram

Class diagrams are one of the most useful types of diagrams for precisely depicting the structure of a system by modeling its classes, characteristics, actions, and object relationships. It shows the system's classifiers' static structure. The picture depicts the fundamental notation used in various structural diagrams [The UML 2 Class Diagram - IBM Developer \(n.d.\)](#). The real class shape is a rectangle with three rows. The class name appears in the top row, the class characteristics in the middle row, and the methods or operations that the class may use in the bottom row. The arrangement of classes and subclasses demonstrates the static relationship between each item.

Class Diagram for GrammarGenius

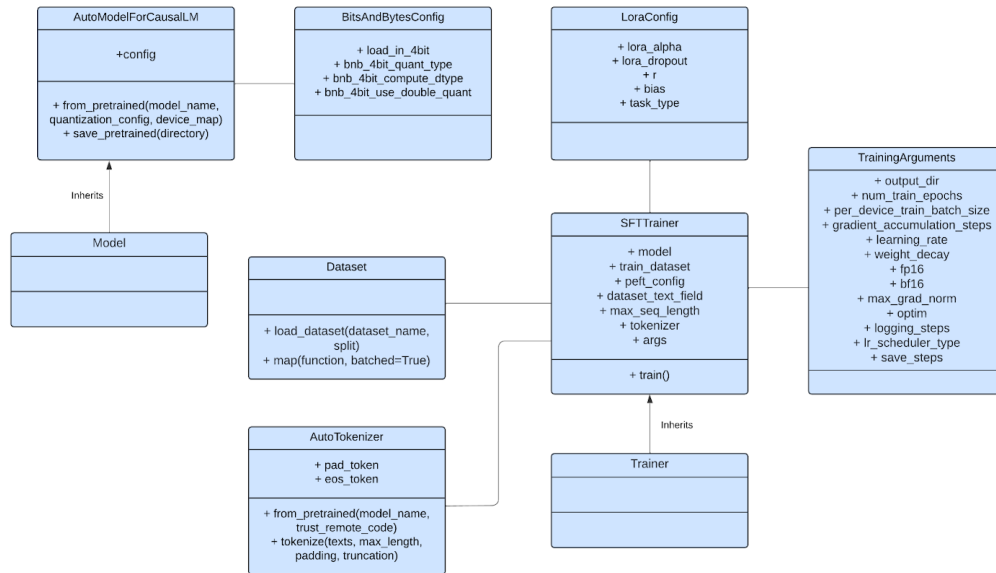


Figure 3.5: Class Diagram of GrammarGenius

The above diagram is the class diagram of our model. The `AutoModelForCausalLM` is the class from which the pre-trained LLaMA 2-7B Chat model will be instantiated. It is a causal model, meaning that text is generated one token at a time. The `BitsAndBytesConfig` class is used to set the quantization settings of the model. Quantization is a process in which the precision of model weights is reduced with the purpose of saving memory and computational resources. The `LoraConfig` class sets the low-rank adaptation settings of the model. Low-rank adaptation is a method that involves adding a small, low-rank matrix to the parameters of the model; therefore, it is efficient for fine-tuning. In the `TrainingArguments` class, the hyperparameters of the training are defined, including but not limited to the output directory, learning rate, number of training epochs, batch size, and evaluation steps. The `SFTTrainer` class is responsible for training the model using the training loop, pre-fetching data, and for this purpose, running the model forward pass, backward propagation, and optimization. The `Dataset` class is an object class representing examples of text with grammatical errors and the proper versions of the JFLEG dataset. The `AutoTokenizer` class is in charge of tokenization or the division of texts into sub-units that a model can understand. Lastly, the `Model` class represents the fine-tuned model that will, in fact, be used in carrying out the GEC task. That is, this is the base LLaMA 2-7B Chat model whose parameters have gone through updates in the fine-tuning process.

3.2.3.5 State Diagram

A state diagram includes states, transitions, events, and activities. State diagrams the dynamic picture of a system. They are useful for representing the behavior of an interface, class, or cooperation. A state diagram highlights an object's event-ordered behavior. In a state diagram, a state is a circumstance or scenario that an item experiences over its lifetime in which it meets a condition, performs an action, or awaits an event. An event is a noteworthy occurrence with a definite place in time and space. A transition is a connection between two states that indicates that an object in the first state will undertake specific activities before entering the second state when a defined event happens and certain criteria are met. An activity is a non-atomic execution that occurs continuously within a state machine. An action is an executable atomic calculation that changes the model's state or returns a value. Graphically, a state is represented as a rectangle with rounded corners. A transition is shown as a solid, directed line [All You Need to Know about State Diagrams \(n.d.\)](#).

State Diagram of GrammarGenius

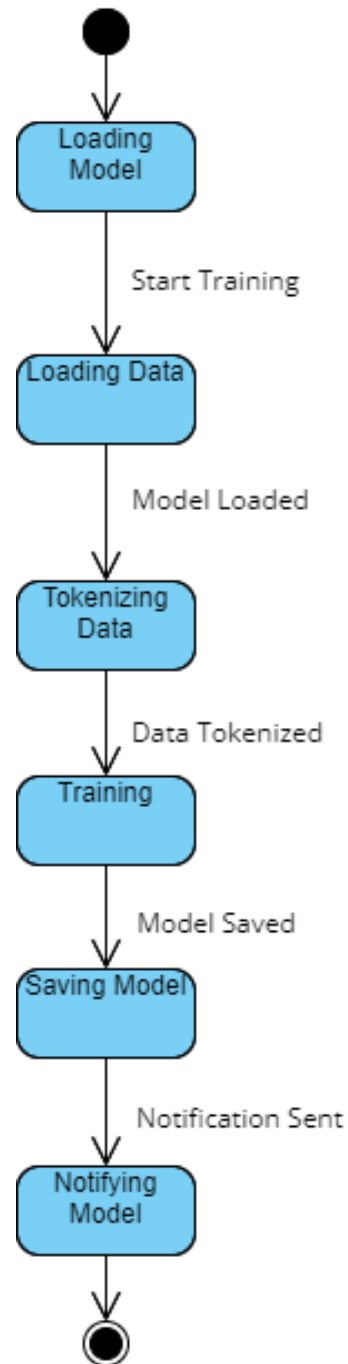


Figure 3.6: State Diagram of GrammaGenius

The fine-tuning of our model proceeds with the steps in the diagram above. First, it loads the pre-trained language model from which to base the fine-tuning. After that, it loads the training data, which includes examples of the particular task the model will be trained on, in our case, the JFLEG dataset. Following the loading of data, the data gets tokenized, meaning the text gets broken down into units called tokens. These tokens are essential in understanding and processing the text by the model. Further, the model is trained on the tokenized data. In training, parameters of the model are tuned in a way that best improves its performance for the target task. Once the training is finished, the fine-tuned model is then saved to be used in the future. Finally, a notification is given where fine-tuning is successful. This figure illustrates schematically some of the main steps involved in the process of adapting a language model to a particular task.

3.2.4 Development

The development process of our project is discussed in the following subsections. Moreover, the following subsections discuss the specific language and libraries used to develop the project and what contributions they made to make our project a success.

3.2.4.1 Python

Python has broad applications including machine learning, data science, web development, automation of processes, and testing of software. Since it is open-source, that automatically means it is free to use, having a very extensive community of developers who improve it all the time. Because Python features intuitive syntax, it is especially appealing for beginners, while at the same time, Python's strong potential makes it suitable for professionals when solving very complicated problems. The ease of use combined with robust functionality has established Python's status as one of the most widely used programming languages in the world [GeeksforGeeks \(n.d.\)](#). Python's strong ecosystem and ease of use meant it played an instrumental role in the project development process. Large libraries like Transformers, PEFT, TRL, Torch, and BitsAndBytes make rather complex processes loading a pre-trained model, applying quantization, and fine-tuning large language models like LLaMA-2

easier. The reason behind this is Python's flexibility, which allowed for easy integration of advanced techniques for efficient model optimization, such as LoRA. Python uses the Datasets library to handle large datasets in a very straightforward way that enables the loading, preprocessing, and formatting of large datasets required for effective model training. These powerful tools combined with ease of implementation were key to successfully developing and deploying the system.

3.2.4.2 Python Libraries

- The Python libraries used in this project are:
- Transformers: We used this library to load our pre-trained model which is the llama-2-7b chat model and then used this library to fine-tune our model with further configurations.
- PEFT: By using this library we are ensuring that we can perform fine-tuning in a parameter-efficient way by using methods such as LoRA. We have used this library to provide an easy way to add lightweight layers to our llama-2-7b chat model so that we can tailor our model for grammatical error correction at a low computational cost.
- Torch: This is the main library used in our model which deals with deep learning by providing tensor computation and GPU facilities. The model training of our project is based on the torch library.
- Datasets: We have prepared our dataset and uploaded it to our huggingface account. To import our dataset from huggingface we have used the library named 'datasets' in our project. In addition to that, it has enough capability to handle large datasets and prepare them in formats suitable for machine learning tasks.
- BitsAndBytes: This library is used to optimize our model in training through quantization methods, in the case of our project, 4-bit precision. It handles the memory efficiently and speeds up GPU training with minimal performance sacrifice.

- TRL: This is the library that is used for fine-tuning transformer models in a supervised way and performing reinforcement learning. The library 'trl', which efficiently trains models-such as LLaMA-2-is integrated with many tools, among them SFTTrainer for the least-complex fine-tuning of models.

3.2.5 Testing

After developing the solution, it is necessary to check if the system works properly without any errors and fulfills all the requirements. All the bugs and defects must be checked and addressed. All the functionalities and features are checked in this phase if all of them are working as per the plan [TechTarget \(n.d.\)](#).

3.3 Software Requirement for Development

3.3.1 Google Colab

Google Colaboratory, best known as Google Colab, is a cloud-based Jupyter Notebook. From running general Python codes to working on different machine learning algorithms to data science projects, Google Colab is considered to be the best solution as it has numerous Python libraries pre-installed, such as NumPy, Pandas, Scikit-learn, TensorFlow, and PyTorch, as well as visualization tools like Matplotlib, Seaborn, and Plotly, which make it simple to construct graphs, charts, and visualisations to analyze and present data. Setting up the environment is much easier with Google Colab, and specific libraries that are not pre-installed can be easily installed from a notebook using the pip command. It also provides free access to graphics processing units (GPUs) and tensor processing units (TPUs). Enabling this accelerates the model training process and reduces the time required to produce results. Notebooks created in Google Colab are automatically backed up to Google Drive, allowing easy storage and sharing with other developers and enabling real-time collaboration. Developers can access the code, update it in real-time and comment on it [Google Colab: the power of the cloud for machine learning \(n.d.\)](#).

3.3.2 Anaconda

Anaconda is a platform that is open-source, meaning anyone can use it free of cost. It utilizes the languages Python and R which helps in simplifying package

handling and interface. In order to give the users a smooth experience and avoid any inconvenience while installing the software, anaconda uses a package management system named Conda to handle package models and assess the present conditions for usage. The mentioned platform includes approximately 250 items that are automatically installed. PyPI allows you to install more than 7500 additional open-source collections, including Conda [Domino Data Lab \(n.d.\)](#).

3.3.3 Jupyter Notebook

Jupyter Notebook is a very well-known application that can be used by anyone free of cost. It is a server-client application that helps users edit and run pages in Jupyter Notebook. With the help of this application, data scientists can make new documents, share them with others, and avail themselves of the features of live coding. Data scientists can use it for many different tasks such as cleaning data, data visualization, machine learning, EDA and deep learning. Additionally, to use the Jupyter Notebook one does not need to have internet access [Databricks \(n.d.\)](#).

3.3.4 Mendeley

Mendeley is a reference management program that allows the organization of citations to articles, books, government documents, book chapters, and websites. Citing articles found in multiple library databases can be done through a few clicks using Mendeley. It allows effortless inclusion of these references and automatic production of bibliographies into Word documents. It also offers saving, taking notes, and sharing features. A web version is also available to maintain a research network of partners [Adams \(n.d.\)](#).

3.3.5 LaTeX

LaTeX is a document preparation system that produces high-quality typesetting. It is primarily used for technical or scientific publications. It provides control over huge documents that include sectioning, cross-references, tables and figures, sophisticated mathematical formula typesetting, and advanced mathematics typesetting using AMS-LaTeX. It also includes automatic bibliographies and indexes, multilingual typesetting, artwork, process or spot color, and the ability to use PostScript or Metafont fonts [Introduction to LaTeX \(n.d.\)](#).

3.3.6 Lucid chart

Lucidchart is a web-based visualizing program that allows individuals to visually communicate while creating, editing, and sharing charts and diagrams, as well as enhancing processes, systems, and organizational structures. It is the top intelligent diagramming application, allowing teams to reduce complex challenges, unite on ideas, and complete projects more quickly. Real-time updates allow teams to function concurrently [Workshops \(n.d.\)](#).

3.3.7 Visual Paradigm

Visual Paradigm is a web-based editor that helps users visualize data by offering data widgets, charts, and maps, as well as tables, information, and spreadsheet tools that promote productivity. It is a software tool that enables software development teams to model corporate IT systems and oversee the development workflow. Alongside modelling assistance, this software offers report production and code engineering. It is complimentary for non-commercial purposes such as for personal projects, education, and non-profit organizations [Ideal modeling & diagramming tool for agile team collaboration \(n.d.\)](#).

3.4 Dataset Preparation

3.4.1 Data Collection

We chose to focus on fine-tuning the model for Grammatical Error Correction as a means of enhancing its grammatical capabilities. For this we used the benchmark dataset JFLEG, taken from Hugging Face. The JFLEG dataset is well-acknowledged because, not only does it focus on but also provide sentences with errors and their corresponding corrections to evaluate the GEC performance. It contains multiple reference corrections per sentence, thus providing a diverse and flexible assessment framework. What really sets it apart from other GEC datasets is that JFLEG pays a lot of attention to the fluency and naturalness of the sentence which has been corrected, rather than the grammatical error correction according to some strict grammatical rules. This provides a wide dimension for more human-like corrections, thus fitting the capabilities of the model Llama-2-7B-chat very well. By using this dataset, we had the opportunity to train the model not only for grammatical mistake fixing but also to enhance readability and sentence flow. This made it a very useful contribution in assessing the model's ability to handle real-life language usage, which often comes with subtle corrections other than mere grammar fixes.

3.4.2 Dataset Pre-processing

Preprocessing of the JFLEG dataset was performed to suit the requirement of a seq2seq model, structuring the data so that it could take an incorrect sentence as input and generate its corrected version as output, thereby fine-tuning it correctly for GEC. The JFLEG dataset was loaded from huggingface using the 'Datasets' library. The JFLEG dataset has two columns 'sentence' and 'corrections' where 'sentence' includes the incorrect forms of sentences and 'corrections' include the respective corrected forms of each sentence. The JFLEG dataset is made of 755 rows of validation split and 748 rows of testing split. Our preprocessing function iterates through the dataset, taking in inputs of incorrect sentences and outputs of one of the corrected sentences - the first correction if available. This takes the input and output pairs and tokenizes them using the model's tokenizer to make sure both inputs, that is, the wrong sentences and the correct sentences get padded and truncated

to some maximum sequence length. This will format the data such that the model learns to map sentences that are incorrect to their respective corrected versions in the fine-tuning process.

Chapter 4

Design and Development

4.1 Project Management

Project management is a rigorous approach to planning, executing, and managing projects in order to achieve certain goals while adhering to schedule, budget, and scope constraints. It uses various skills, tools, and techniques to guide the project through different stages such as starting, planning, executing, monitoring, and finishing. Project management also helps in the realization of goals through the breakdown of big tasks into small ones, proper utilization of resources to avoid waste, early identification and management of risks, promotion of clear communication among members and stakeholders of the team, and setting standards for quality in results *What Is Project Management and What Are the Types?* (n.d.). The following section describes the project management components of our system - a chatbot using the CoNLL Dataset and Llama-2-7B model in grammatical error correction. Herein, we review our strategy for handling project management and how the Work Breakdown Structure was utilized to help in dividing up tasks and responsibilities efficiently.

4.1.1 Work Breakdown Structure

A Work Breakdown Structure is a project management approach that breaks down a large project into minor, more accomplishable components. It creates a clear hierarchy for the entire project, beginning with the overall aim and progressing down to significant deliverables, minor tasks, and individual activities. The WBS is important because it aids in the establishment of the project scope, enhances resource utilization, allows for interaction with other team players, and facilitates easy cost estimation and scheduling. Work breakdown structure ensures that all coverage of a project is covered, and hence there will be the production of quicker and better results [What Is a Work Breakdown Structure \(WBS\) In Project Management? \(n.d.\)](#).

Work Breakdown Structure of This Project

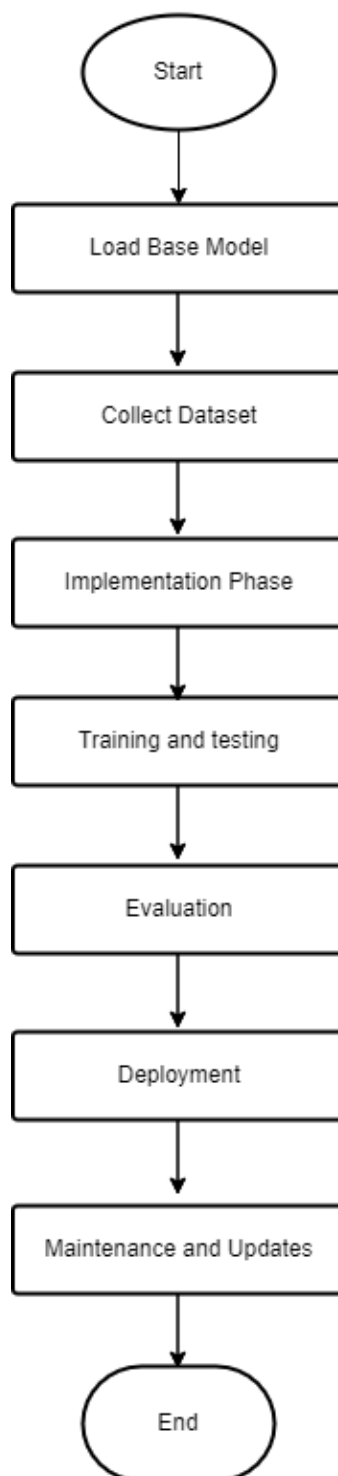


Figure 4.1: WBS of GrammaGenius

4.2 Proposed Model Deployment

4.2.1 Fine Tuning Llama-2 7B Chat Model

The fine-tuning of the Llama-2-7B chat model includes adapting to the pre-trained architecture of the model to attain better performance on certain tasks, such as grammatical error correction. This is considered to be an efficient process where memory consumption is minimal; hence, we used a method called QLoRA or Quantized Low-Rank Adaptation. In this regard, QLoRA still keeps all the original parameters through fine-tuning the Llama-2-7B chat model. This technique is utilized to adjust the pre-trained model for handling new tasks. It adds Low-Rank Adapters in a memory-efficient manner. First, the model Llama-2-7B is taken from the Hugging Face hub, and a dataset of sentence pairs with grammatical errors and corrected versions of those sentences was created. A few training parameters, like the learning rate and size of the batch, are fixed to allow the model to learn correctly. The training should go through a certain number of epochs to improve the model response by generating good responses from our examples in the dataset. Once the entire fine-tuning process is finished, the model is saved for later use. In this way, we not only improve the grammatical correction of the model but also easily handle the training of huge language models on modest hardware platforms using QLoRA and other techniques. The new chatbot provides proper corrections to users and can polish writing. We use a technique, called Total Low-Rank Adapters (QLoRA), which allows us not only to avoid unnecessary computation but also to avoid storing the entire original model in memory. With QLoRA we add Low-Rank Adapters to the pre-trained model, and only additional parameters are trained to solve a new task.

4.2.1.1 Saving and Loading Fine-Tuned Llama-2 7B Chat Model

In this section, we will discuss how we saved the modified model and the tokenizer for future use. This is crucial as it allows access to a model with its newer features without any retraining procedures being initiated. We first load the fine-tuned model in FP16 format as it helps in saving memory and performance. We took this approach as this allows working with fewer resources while still delivering good output. After enhancing the base model, we merge it with the fine-tuned LoRA

weights. Following this, we save the tokenizer associated with our model. The tokenizer is important because it tokenizes input text into the accepted formats, hence allowing the effective use of sentences. In less technical terms, it is also important to set the padding token as the same as the end-of-sequence token, to allow input handling smoothly. We then pushed the model to the Hugging Face Model Hub for future purposes and to share it with others. This way, others can access our fine-tuned model effortlessly without going through training of their own. Once saved, we can do a performance check on our fine-tuned model whenever we want to. For our saved model, we automate text generation, allowing us to see how successful our chatbot is in correcting different prompts with grammatical errors. For instance, we give sentences with certain errors, and we perform the performance checks for our chatbot within this scope.

4.3 Implementation

4.3.1 Front-end using Gradio Library

The following section deals with the creation of a front-end interface for our chatbot using the Gradio library. Gradio is a great tool to develop interactive web applications for machine learning models effortlessly. This would give the users an easy and fun way to connect to the Chatbot powered by our fine-tuned Llama-2 7B model. The front-end design is very user-friendly so that anyone can quickly type their questions or prompts and receive instant responses from the chatbot. One of the great features of this interface is the chat history, which records what has been said so far. This is crucial because it made the model comprehend the conversation and made the interaction more human-like. After the users type in their prompts, the system will instantly accept their input and generate responses by thinking about what has been said earlier. The responses will soon appear in the chat window for the user to read easily and have a better understanding of what the chatbot said. Before outputting the replies for users to read, responses are cleaned from unwanted formatting and empty lines to present better. Using Gradio for our front end has made it much easier for people to engage with our grammatical error correction Chatbot. The simplistic design encourages users to play around with what the model does, making it accessible to anyone. All in all, it showcases the capabilities of our model and provides an interactive experience to the user.

4.3.2 Back-end using Python

We chose Python for the back end since it has a great variety of libraries to suit most machine-learning tasks. As a basic environment, we used Google Colab because it supports cloud-based access to powerful GPUs - a highly desirable element when working with big models, such as Llama-2-7B. We initiated the code by loading the pre-trained Llama-2-7B model from the Hugging Face Model repository via specification with "NousResearch/Llama-2-7b-chat-hf". To simplify this task for ourselves, we used several Python libraries: 'torch', 'transformers', 'datasets', 'peft', and 'trl'. Such libraries are highly valued in terms of loading, fine-tuning, and operating with models. These helped us customize the model for GEC. Lastly, with the intent to make our model consume less memory, we used QLoRA, a variant of LoRA that applies 4-bit quantization and has enabled the management of the model with an additional degree of efficiency. The 'bitsandbytes' library helped do that, and support for GPU in Google Colab provided an avenue for speeding up training by using half-precision or 'float16', reducing the computational load without affecting performance. The training parameters were carefully tuned for the learning rate, batch size, and gradient accumulation steps, so that model training could take place effectively with results accurate enough. It relies on the class 'SFTTrainer' from the library 'trl' for guiding this fine-tuning, while a custom dataset designed for Grammatical Error Correction provided by Hugging Face's Dataset Hub was used. After fine-tuning, we saved the model and began testing the model with a text generation pipeline to check its capability in sentence correction. Armed with the powerful Python libraries and huge computing resources in Google Colab, we successfully worked on developing and deploying a fine-tuned language model for improved grammatical accuracy.

4.3.3 Algorithm: How Sequence-To-Sequence Is Implemented?

We have used the sequence-to-sequence approach to train our model to correct grammatical errors in any sentence. The base model used in our project is Llama-2-7b-chat-hf which we have collected from huggingface. The dataset we have used for training is JFLEG, comprising sentences with grammatical mistakes and their corrected ones. Seq2seq involves feeding the model with an input sequence, and in our case, an incorrect sentence with grammatical errors, to generate its

corresponding output sequence, that is the corrected sentence. This is an ideal for GEC tasks because it allows a model to learn how to transform a grammatically incorrect sentence into a grammatically correct one. In our project, sequence-sequence is incorporated in the preprocessing stage of the dataset, where input sentences with errors and corresponding corrected sentences get tokenized and paired together. In this way, a model could learn to map an incorrect sentence to its correct form. This follows the seq2seq framework where a model generates a target sequence, which is the correct sentence, from an input sequence, which is the incorrect sentence. This helps the model to form a structured relationship between the incorrect and corrected sentences. In this way, sequence to sequence technique is incorporated into our project to implement grammatical error correction.

4.4 Model Output Snapshots

Output snapshots of our model fine-tuned with JFLEG dataset implementing seq2seq approach is given below:

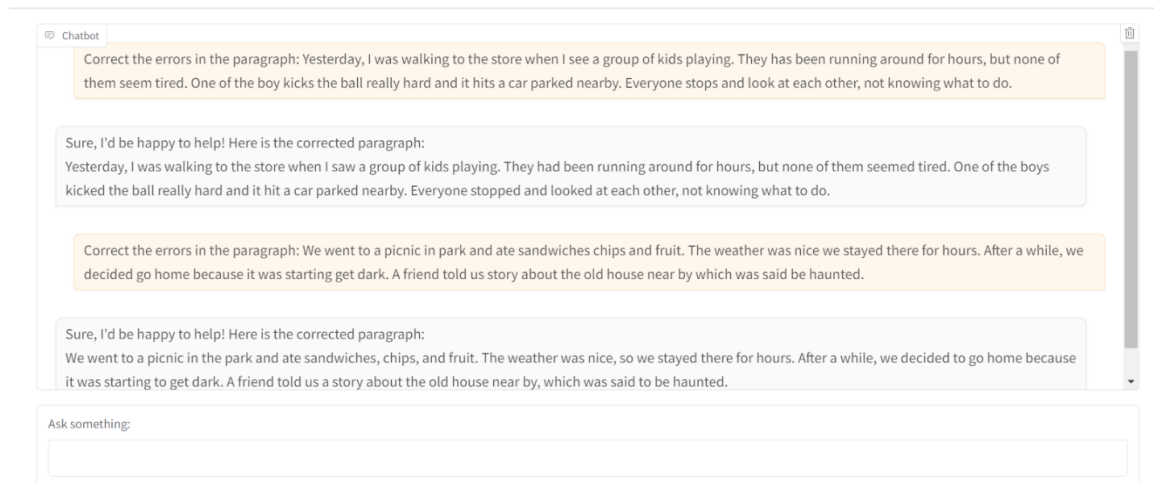


Figure 4.2: Output of Our GrammarGenius Chat Model successfully correcting tense, subject-verb agreement, morphological, punctuation, preposition, syntax, determiner errors in the inputted sentences

Our model, in the first paragraph, makes tense and aspect corrections, like the change of "see" to "saw" and "has been" to "had been"; subject-verb agreement errors, like "They has" becoming "They had"; morphological errors, like "One of the boy" becoming "One of the boys." It also corrects several places where either the verb or subject has disagreed in number or tense.

In the second paragraph the model addresses punctuation errors; for example, missing commas, preposition mistakes, such as "in park" to "in the park," and syntax problems like "decided go" to "decided to go." It also fixes determiner mistakes; for example, it uses "a story" where the original just said "story." It further corrects word placement in order to make sure information flows clearly-for example, "starting get dark" is changed to "starting to get dark."

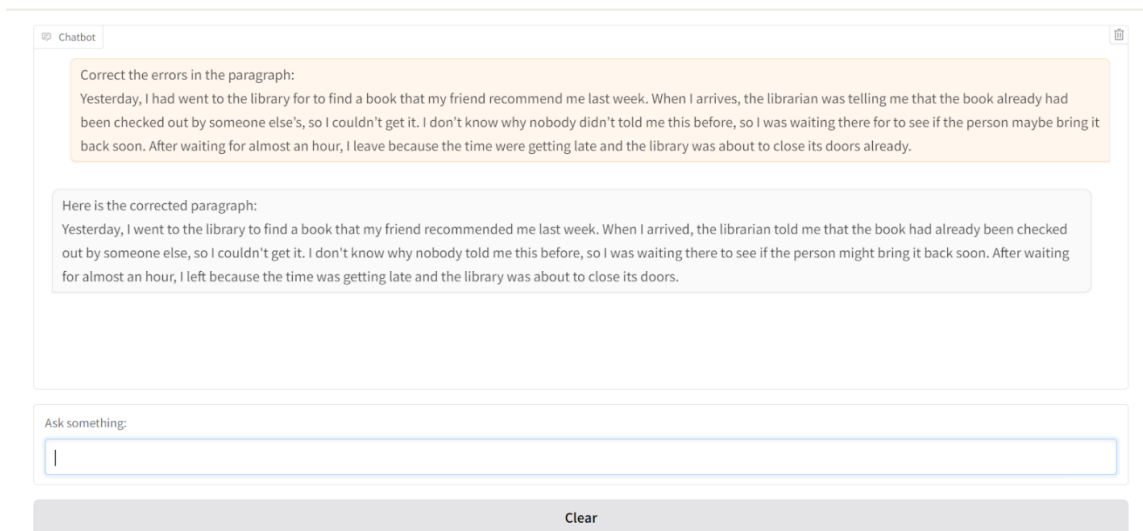


Figure 4.3: Output of our GrammarGenius Chat Model successfully correcting tense, subject-verb agreement, morphological errors in an input paragraph

In the above given paragraph, the model corrected a number of different types of grammatical mistakes, including tense and aspect mistakes: "had went" to "went," "arrives" to "arrived" for consistent past tense. Furthermore, the model could also correct verb morphology mistakes by changing "recommend" to "recommended" and "maybe" to "might" to match appropriate tense. It also detects subject-verb agreement errors, changing "the time were" to "the time was." Last but not least, the model checks negation errors; it changes "nobody didn't" to "nobody told me," whereby the meaning can be clearer.

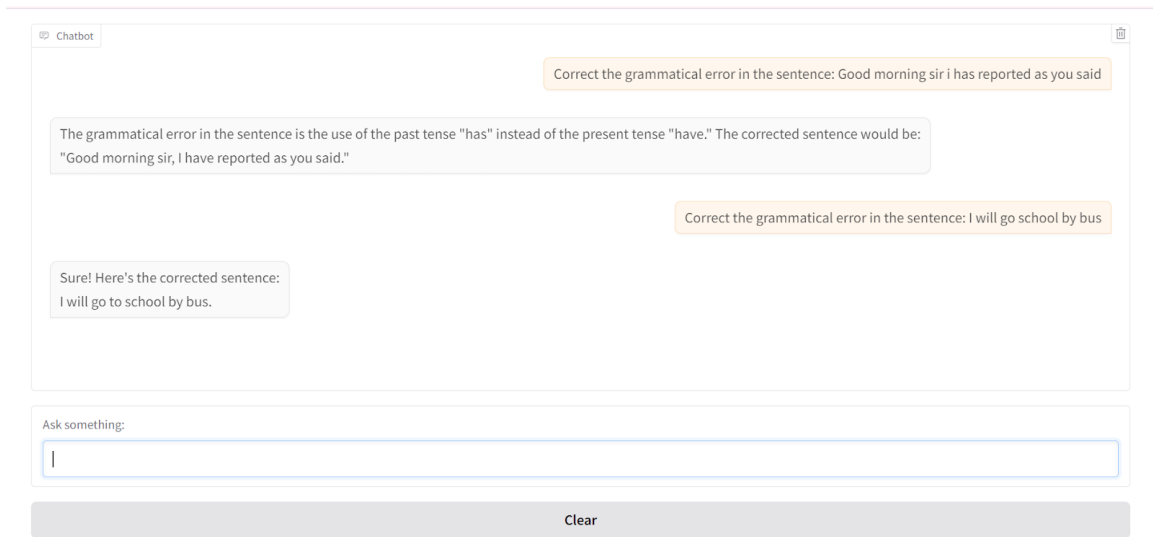


Figure 4.4: Performing GEC Task on The Model Fine-tuned with GEC Dataset

After running the same prompt in GrammaGenius where specific techniques and datasets for GEC is used, we saw the model answered correctly and generated a sentence where it said the correct form is 'I have reported as you said' which is grammatically correct. This correction shows the model's ability to correct tense and structure based on contextual clues.

Chapter 5

Result Analysis

5.1 Performance Analysis

The performance of our model was tracked by using key indicators available in TensorBoard such as epoch progression, dynamics of learning rate, and reduction of training loss across time. TensorBoard is a powerfully integrated tool in machine learning designed for visualization and tracking model training [What is TensorBoard? — ArcGIS API for Python \(n.d.\)](#). It comes with a very user-friendly dashboard where we can see key metrics, such as loss and accuracy over time, so we can get a much better idea of how well our model is doing. We can also visualize our model structure with TensorBoard, view images and histograms, and even analyze how different parameters are affecting training. This visualization will enable us or any other developers to identify some problems, such as overfitting or underfitting, and make reasonable changes to improve their models. In general, TensorBoard improves the work process of machine learning by bringing clarity into the process of training. The performance analysis of the GEC model trained on the JFLEG dataset with sequence-sequence technique is shown and discussed below.

5.1.1 Training Epochs

Typically, an epoch means one full cycle where the algorithm processes every single sample once in the training dataset. More precisely, every sample is used once during processing by the model in training. Epochs are a hyperparameter that defines how many times the learning algorithm is going to iterate through the training data. More epochs allow the model to improve knowledge of the patterns in the data. If too many epochs are allowed, then it will also pick up noise rather than generalizable patterns; hence, one needs to find an optimum number for effective training [Epoch in Machine Learning - GeeksforGeeks \(n.d.\)](#).

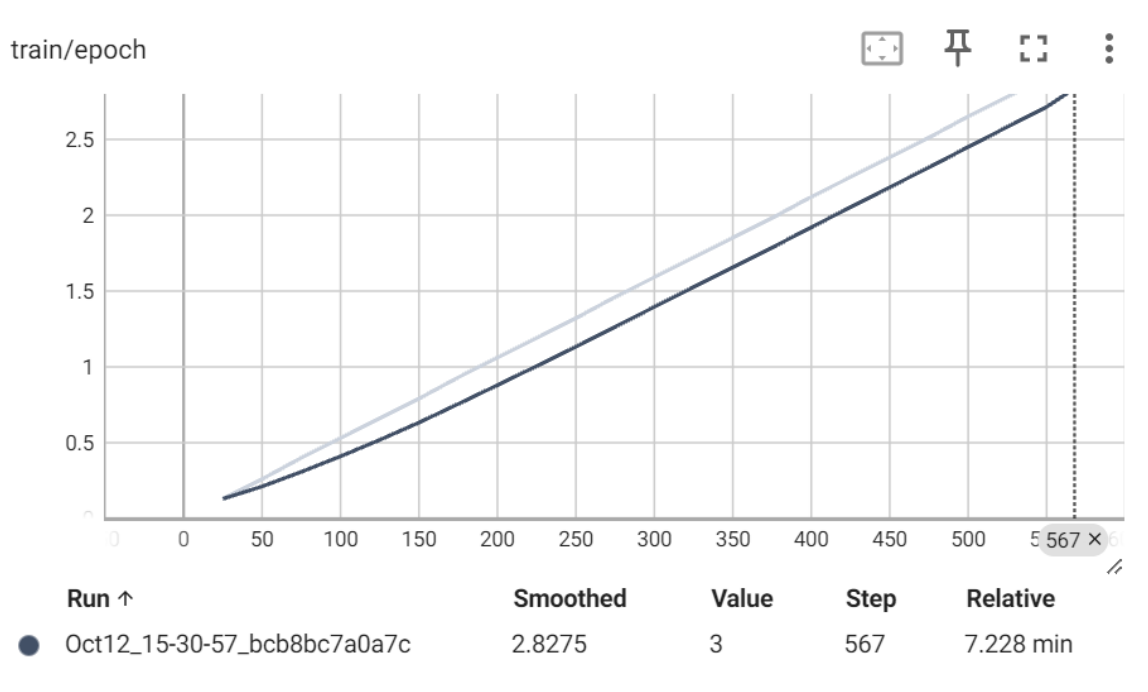


Figure 5.1: Train/Epoch graph of the model GrammaGenius

The graph for the train/epoch shows a linear progress in the number of epochs against the duration of training. The model has been trained for about 550 epochs. From this steady increase, it can be inferred that the training process ran without any interruptions or errors.

5.1.2 Learning Rate Decay

Learning rate decay in machine learning is a technique of gradually decreasing the learning rate during training. The approach first starts with a higher learning rate, which allows the model to learn more quickly and may help jump the model out of local minima; then it decreases this rate in order for finer adjustments while the training progresses. This helps to prevent problems such as oscillations and slow convergence that might arise when the learning rate keeps fixed. Learning rate decay can be realized in different ways, such as step decay, exponential decay, and inverse time decay, based on different training scenarios. If applied appropriately, learning rate decay can lead to faster training, better convergence, and hence generally better performance of a model [Learning Rate Decay - GeeksforGeeks \(n.d.\)](#).

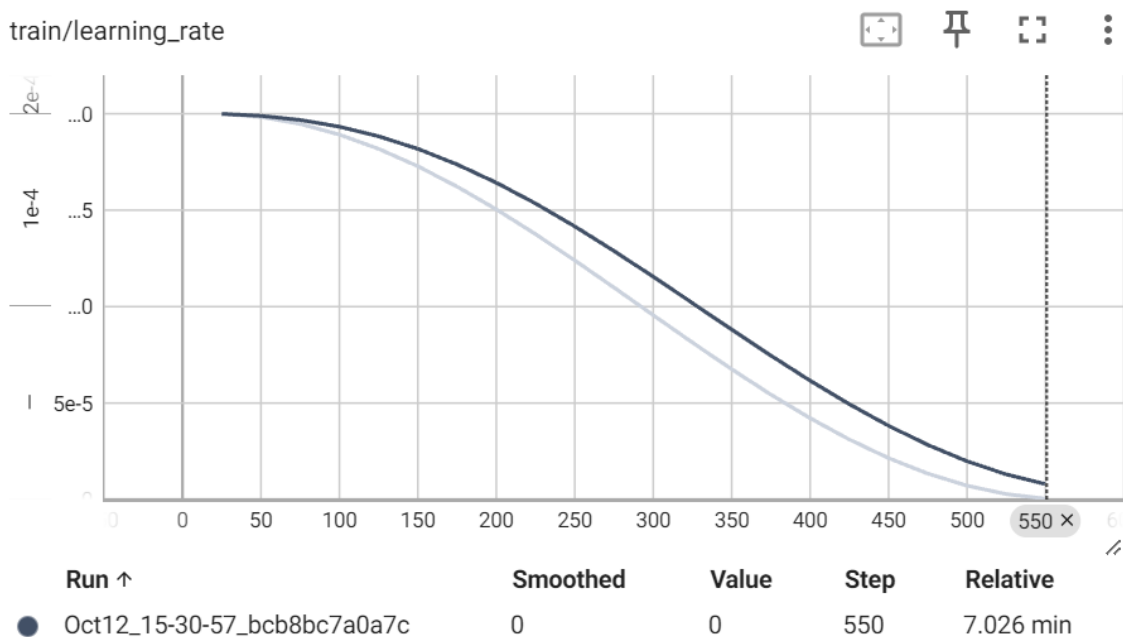


Figure 5.2: Train/learning rate of the model GrammaGenius

In the train/learning_rate graph, the initial learning rate starts off around 1.8×10^{-4} or 0.00018. As it trains, the learning rate decays linearly. After halfway through training, it sits around 1×10^{-4} or 0.00010. By the end of training, it has decayed further down to roughly 2×10^{-5} or 0.00002. The idea here is that a fine reduction in the

learning rate helps the model converge more smoothly since, while approaching an optimal solution, it takes smaller strides.

5.1.3 Training Loss Reduction

Training loss is a key metric that represents how well an LLM learns from the training data. The training loss can be measured by quantifying the difference between what the model predicts and what the actual target value is. It is calculated as a result of a loss function, most usually cross-entropy loss for tasks such as next-word prediction; hence, the training loss is a reflection of model performance, where lower values indicate better accuracy. The monitoring of training loss helps diagnose issues, such as overfitting, where a model memorizes the training data instead of generalizing, and underfitting, where it misses the representation of underlying patterns. In the end, minimization of the training loss is required with the help of some optimization algorithms, such as stochastic gradient descent, to enhance the performance of the model on unseen data [Baeldung on Computer Science \(n.d.\)](#).

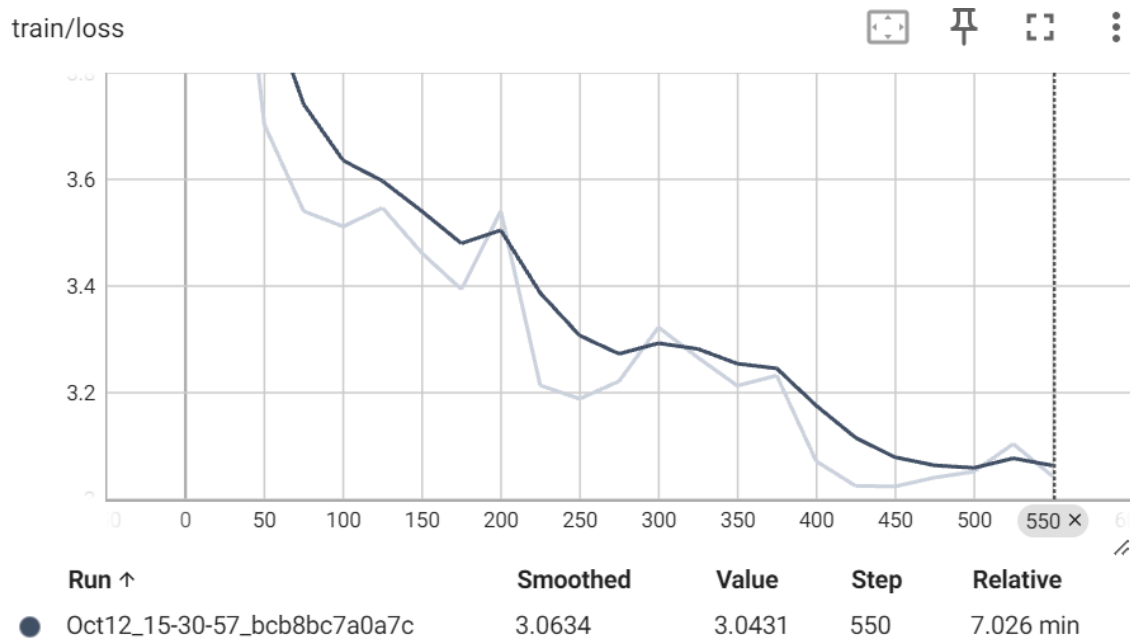


Figure 5.3: Train/loss graph of the model GrammarGenius

This graph of train/loss shows a continually decreasing loss for the training process from a starting value of about 3.7 to a steady approximate drop to about 3.1 at epoch 550. The fact that the graph shows the rate of loss going down means that this model was learning from this training process, improving in time and enhancing its prediction capabilities. Minor fluctuations in the loss curve might suggest minor variations from batch to batch in how well the model generalizes. The remaining fluctuations could be reduced by further fine-tuning.

5.1.4 Testing Loss Graph

The performance statistic of the machine learning model on a separate test dataset that has never been used for training or validation is simply referred to as testing loss. By computing the difference between expected outputs and actual goals on the test dataset, it serves as an indication of generalisation to new, unknown data. Testing loss provides an objective assessment of the model's performance in the actual world and is therefore seen as a crucial indication of the model's resilience and generalisation, in contrast to training and validation loss, which direct model optimisation during the training phase ?.

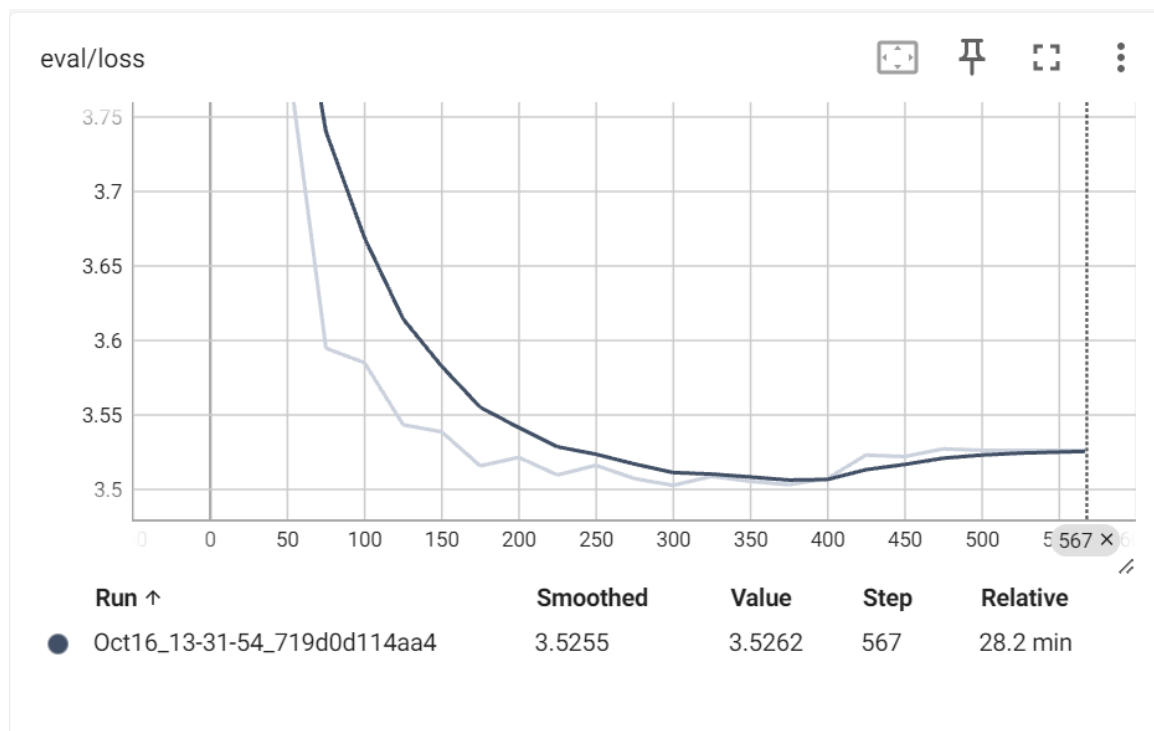


Figure 5.4: Eval/loss graph of the model GrammarGenius

The graph visually represents the learning process of a language model fine-tuned for grammar error correction (GEC). Initially, the model's performance is poor, leading to a high loss value. As training progresses, the model learns to identify and correct grammatical errors, resulting in a decreasing loss. The blue line, a smoothed version of the loss, provides a clearer view of the overall trend. Eventually, the loss plateaus, indicating that the model has reached its optimal

performance level. The low final loss value demonstrates the model's effectiveness in addressing GEC tasks.

5.2 Discussion

These outputs of the training process are reflected in the TensorBoard metrics, showing that this model was developed very well and effectively. Gradual reductions in the learning rate over the course of training hint that the optimization was designed to converge effectively, allowing the model to gradually hone its performance during training. This is further evidenced by the fact that the model was learning well-that the training loss was constantly going down-meaning less error with each iteration. Stability in the progress of epochs without interruptions means the training cycle was seamless and robust. All in all, their combination underlines a successful run of training in which the model fits in effectively to learn the task at hand and demonstrates steady improvement in performance.

Chapter 6

Conclusions

6.1 Social, Ethical, and Environmental Issues and Impacts

This project explored the areas of improvement in the field of grammatical error correction task for the recently hyped conversational AI models, at first through a systematic literature review followed by an attempt of implementing the seq2seq approach in the LLaMA-2 7B chat model. The model was fine-tuned with JFLEG dataset and integrated with the seq2seq approach for improved GEC capability. The model was made more computationally efficient by combining QLoRA quantization and improved fine-tuning approaches. This updated model is a promising tool to rectify the grammatical issues present in the input sentence, prompted by the user to assist in the GEC task. The model has the potential to enhance the quality of written communication while taking into account the technology's social, ethical, and environmental ramifications.

6.1.1 Social

The usage of generative AI tools for completing different sorts of works, including professional articles, report making, other types of documentation, creative writing, homework, and assignments, has become a trend at present times. For all the written tasks, the correctness of grammar in the written document is a crucial factor in getting acceptance. Millions of users are there to get help from these types of generative AI tools that assist them with grammar. Hence, these types of GEC tools have the potential to have a significant impact on society. Not only within the academic community but also in professional domains like news reporting and other official documentation, these tools might come in handy, improving the quality of grammar. Aside from that, non-native speakers might considerably benefit from such systems by overcoming linguistic barriers and gaining fresh opportunities. However, the over-reliance on AI and its negative consequences are undisputed realities. Many students may rely on these programs to fix their grammar and enhance their writing while reducing their own writing and comprehension abilities. Many people may overlook the learning process and favour using these technologies to do their tasks. As a result, their writing skills may deteriorate, and there is little prospect of improvement. Furthermore, AI-generated concepts must be inclusive, retaining linguistic variance and expression rather than adhering to a narrow, biased view of "correct" language use.

6.1.2 Ethical

The project must address ethical concerns about data use, model biases, and AI accountability. The ethical considerations include privacy and the safe usage of AI. However, in order to ethically employ the AI model, transparency is required. Users must recognize that the AI tool's adjustments and suggestions may not always be faultless or appropriate in all instances. Bias can also occur if the AI model only examines specific grammatical standards, discriminating against non-standard dialects or linguistic variants. Moreover, the model should promote academic integrity. Students must use the tool for learning purposes, not merely to copy corrections without comprehending the underlying language.

6.1.3 Environmental

AI model training may have substantial effects on the environment because the process is computationally expensive and hence requires a lot of energy. The project attempted to decrease its computing costs and resources to perform training by employing QLoRA for quantization. Thus, efficient model training could potentially be accomplished with less resources, lessening the carbon footprint. It serves to promote sustainability by raising efficiency while consuming less energy. Whilst the environmental impact of AI remains a concern, the project has made efforts toward sustainability by increasing efficacy and lowering energy use. However, new energy-efficient models and solutions must be studied in order to mitigate environmental impact even more.

6.2 Brief Summary

The research succeeded in creating a grammatical error correction (GEC) system employing the LLaMA-2-7-B chat model, which was supplemented by implementing seq2seq. The project used the JFLEG GEC dataset and a popular quantization approach, QLoRA. Rather than searching for their desired information on the website, they can ask it through the system, and get the answer within seconds. The project looks into the social, ethical, and environmental impacts of resorting to generative AI tools in professional articles, reports, and creative writing, among other forms. Whereas these tools are crucial in so far as grammatical accuracy is concerned and serve millions of users, especially non-native speakers, the challenge of over-reliance on AI at the expense of users' writing and comprehension skills cannot be gained. Ethically, the project underlines that data should not be used without full transparency. The project also nurtures academic integrity in the students, as they learn from the tool instead of just copying corrections. On the environmental side, AI model training could demand large amounts of energy; the project therefore tries to reduce that with QLoRA for quantization, efficiency, and energy consumption, while striving further toward more sustainable and energy-efficient solutions.

6.3 Future Works

This project has several potential paths for future development: For this project, only one GEC dataset (JFLEG) was used. More benchmark GEC datasets such as NUCLE, ConLL-2013, ConLL-2014, FCE, LANG-8 BEA-2019 and other GEC datasets such as CLC, AESW, GMEG, CWEB, GHTC can be used in order to improve the efficiency of the system.

The model focused only on fixing grammatical errors in English sentences. Developing the model for multiple languages would widen the model's usefulness, particularly in multicultural academic settings providing multilingual support. Adding support for languages such as French, Arabic, Spanish, German, Japanese, and Mandarin would increase the tool's global usefulness and vastly improve the system's inclusiveness. A multilingual GEC model might help with cross-language translation and bridge the gap between non-native speakers in international academics.

Another intriguing area for future research is the creation of customized grammatical feedback and explanation of errors. Rather than just offering general corrections, the system might learn about each user's writing patterns and regular mistakes to make more personalized suggestions. This form of adaptive learning would not only increase the quality of the corrections but would also assist users in improving their writing abilities by correcting their most common errors, resulting in a more interesting and instructive experience. Thus, the system can be a grammar-learning tool as well.

Currently, the model uses the Gradio library to quickly develop an interactive interface. However, a user-friendly interface can be designed to improve accessibility and attract more people to utilize the product. The efficiency of the model can be further improved using the newest techniques and algorithms for GEC tasks utilizing several benchmark datasets.

References

- Adams, S. (n.d.). *Research guides: Mendeley support: What is mendeley?* Retrieved from <https://libguides.northwestern.edu/mendeley/whatismendeley> (Retrieved October 13, 2024)
- Algaraady, J., & Mahyoob, M. (2023, 7). Chatgpt's capabilities in spotting and analyzing writing errors experienced by efl learners. *Arab World English Journal*, 3-17. doi: 10.24093/awej/call9.1
- All You Need to Know about State Diagrams. (n.d.). *All you need to know about state diagrams*. Retrieved from <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/about-state-diagrams/> (Retrieved October 13, 2024)
- Alzahrani, F., & Alotaibi, H. (2024, 04). The impact of artificial intelligence on enhancing efl writing skills among high school students. *Journal of Educational and Human Sciences*, 34, 226-240. doi: 10.33193/JEAHS.34.2024.482
- Baeldung on Computer Science. (n.d.). *Training and validation loss in deep learning*. Retrieved from <https://www.baeldung.com/cs/training-validation-loss-deep-learning> (Accessed: 13 October 2024)
- Bandi, A., Adapa, P. V. S. R., & Kuchi, Y. E. V. P. K. (2023, 8). *The power of generative ai: A review of requirements, models, input-output formats, evaluation metrics, and challenges* (Vol. 15). Multidisciplinary Digital Publishing Institute (MDPI). doi: 10.3390/fi15080260
- Behzad, B. S., & Washington, M. (2024). *Language learning meets generative ai: Utilizing large language models for metalinguistic explanations*.
- Božić, V., & Poola, I. (2023, April). *Chat gpt and education*. (Accessed: 13 October 2024) doi: 10.13140/RG.2.2.18837.40168
- Bryant, C., Yuan, Z., Qorib, M. R., Cao, H., Ng, H. T., & Briscoe, T. (2023). *Grammatical error correction: A survey of the state of the art*.

- Caines, A., Benedetto, L., Taslimipoor, S., Davis, C., Gao, Y., Andersen, O., ... Buttery, P. (2023, July). *On the application of large language models for language teaching and assessment technology*. Retrieved from <https://arxiv.org/abs/2307.08393> (Accessed: 13 October 2024) doi: 10.48550/arXiv.2307.08393
- Chen, M., Ge, ., Zhang, ., Wei, F., & Zhou, M. (n.d.). *Improving the efficiency of grammatical error correction with erroneous span detection and correction*. Retrieved from <https://github.com/nusnlp/m2scorer>
- Coyne, S., Sakaguchi, K., Galvan-Sosa, D., Zock, M., & Inui, K. (n.d.). *Analyzing the performance of gpt-3.5 and gpt-4 in grammatical error correction*. Retrieved from <https://platform.openai.com/examples/>
- Databricks. (n.d.). *Jupyter notebook guide — databricks*. Retrieved from <https://www.databricks.com/glossary/jupyter-notebook> (Retrieved October 13, 2024)
- DataCamp. (n.d.). *An introductory guide to fine-tuning llms*. Retrieved from <https://www.datacamp.com/tutorial/fine-tuning-large-language-models> (Retrieved October 12, 2024)
- Davis, C., Caines, A., Øistein Andersen, Taslimipoor, S., Yannakoudakis, H., Yuan, Z., ... Buttery, P. (2024, 1). *Prompting open-source and commercial language models for grammatical error correction of english learner text*. Retrieved from <http://arxiv.org/abs/2401.07702>
- Domino Data Lab. (n.d.). *What is anaconda? — domino data lab*. Retrieved from <https://domino.ai/data-science-dictionary/anaconda> (Retrieved October 13, 2024)
- Epoch in Machine Learning - GeeksforGeeks. (n.d.). *Epoch in machine learning*. Retrieved from <https://www.geeksforgeeks.org/epoch-in-machine-learning/> (Retrieved October 13, 2024)
- Fang, T., Yang, S., Lan, K., Wong, D. F., Hu, J., Chao, L. S., & Zhang, Y. (2023, 4). *Is chatgpt a highly fluent grammatical error correction system? a comprehensive evaluation*. Retrieved from <http://arxiv.org/abs/2304.01746>
- Farr, C. (n.d.). *Unmasking chatgpt: The challenges of using artificial intelligence for learning vocabulary in english as an additional language*.
- Feuerriegel, S., Hartmann, J., Janiesch, C., & Zschech, P. (2024, 2). *Generative ai. Business and Information Systems Engineering*, 66, 111-126. doi: 10.1007/s12599-023-00834-7
- GeeksforGeeks. (n.d.). *Python features - geeksforgeeks*.

- Gharajedaghi, J. (2011, 12). Formulating the mess. In (p. 159-179). doi: 10.1016/B978-0-12-385915-0.00008-8
- Google colab: *the power of the cloud for machine learning*. (n.d.). Retrieved from <https://datascientest.com/en/google-colab-the-power-of-the-cloud-for-machine-learning> (Retrieved October 13, 2024)
- Gough, D., Davies, P., Jamtvedt, G., Langlois, E., Littell, J., Lotfi, T., ... Grimshaw, J. (2020, 7). Guides and databases: Systematic reviews: What are systematic reviews? *Systematic Reviews*, 9. Retrieved from <https://library-guides.ucl.ac.uk/systematic-reviews/what> doi: 10.1186/S13643-020-01415-5
- Grady, J. O. (2016). Chapter 1 - setting the stage. In J. O. Grady (Ed.), *System verification (second edition)* (Second Edition ed., p. 1-56). Academic Press. Retrieved from <https://www.sciencedirect.com/science/article/pii/B9780128042212000012> doi: <https://doi.org/10.1016/B978-0-12-804221-2.00001-2>
- IBM. (n.d.-a). *What are large language models (llms)?* <https://www.ibm.com/topics/large-language-models>. Retrieved from <https://www.ibm.com/topics/large-language-models> (n.d.)
- IBM. (n.d.-b). *What is a chatbot?* Retrieved from <https://www.ibm.com/topics/chatbots> (Retrieved October 12, 2024)
- IBM. (n.d.-c). *What is llama 2?* Retrieved from <https://www.ibm.com/topics/llama-2> (n.d.)
- IBM Developer. (n.d.). *The sequence diagram*. Retrieved from <https://developer.ibm.com/articles/the-sequence-diagram/> (Accessed: 13 October 2024)
- Ideal modeling & diagramming tool for agile team collaboration*. (n.d.). Retrieved from <https://www.visual-paradigm.com/#:~:text=An%20online%20editor%20that%20provides,Check%20Visual%20Paradigm%20online> (Accessed: 13 October 2024)
- Introducing ChatGPT — OpenAI. (n.d.). *Introducing chatgpt*. Retrieved from <https://openai.com/index/chatgpt/> (Retrieved October 12, 2024)
- Introduction to LaTeX. (n.d.). *Introduction to latex*.
- Katinskaia, A., & Yangarber, R. (2024, May). GPT-3.5 for grammatical error correction. In N. Calzolari, M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, & N. Xue (Eds.), *Proceedings of the 2024 joint international conference on computational linguistics, language resources and evaluation (lrec-coling 2024)* (pp. 7831–7843). Torino, Italia: ELRA and ICCL. Retrieved from <https://aclanthology.org/2024.lrec-main.692>

- Learning Rate Decay - GeeksforGeeks. (n.d.). *Learning rate decay*. Retrieved from <https://www.geeksforgeeks.org/learning-rate-decay/> (Retrieved October 13, 2024)
- Leng, Y., Tan, X., Zhu, L., Xu, J., Luo, R., Liu, L., ... Liu, T.-Y. (n.d.). *Fastcorrect: Fast error correction with edit alignment for automatic speech recognition*. Retrieved from <https://github.com/microsoft/NeuralSpeech/tree/>
- Li, J., Guo, J., Zhu, Y., Sheng, X., Jiang, D., Ren, B., & Xu, L. (2022, 10). Sequence-to-action: Grammatical error correction with action guided sequence generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36, 10974-10982. doi: 10.1609/aaai.v36i10.21345
- Li, W., & Wang, H. (2024, 5). Detection-correction structure via general language model for grammatical error correction. Retrieved from <http://arxiv.org/abs/2405.17804>
- Maity, S., Deroy, A., & Sarkar, S. (n.d.). *How ready are generative pre-trained large language models for explaining bengali grammatical errors?* Retrieved from <https://platform.openai.com/docs/models/>
- Malmi, E., Krause, S., Rothe, S., Mirylenka, D., & Severyn, A. (2019, 9). Encode, tag, realize: High-precision text editing. Retrieved from <http://arxiv.org/abs/1909.01187>
- Mani, A., Palaskar, S., Meripo, N., Konam, S., & Metze, F. (2020, 10). Asr error correction and domain adaptation using machine translation. In (p. 6344-6348). doi: 10.1109/ICASSP40776.2020.9053126
- MindManager. (n.d.). *What is an activity diagram?* Retrieved from <https://www.mindmanager.com/en/features/activity-diagram/> (Accessed: October 13, 2024)
- Ng, H. T., Wu, S. M., Wu, Y., Hadiwinoto, C., & Tetreault, J. (n.d.). *The conll-2013 shared task on grammatical error correction*. Retrieved from <http://www.comp.nus.edu.sg/>
- Omar, R., Mangukiya, O., Kalnis, P., & Mansour, E. (2023, 2). Chatgpt versus traditional question answering for knowledge graphs: Current status and future directions towards knowledge graph chatbots. Retrieved from <http://arxiv.org/abs/2302.06466>
- Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., ... Moher, D. (2021, 3). *The prisma 2020 statement: An updated guideline for reporting systematic reviews* (Vol. 372). BMJ Publishing Group. doi: 10.1136/bmj.n71

- Petiška, E. (2023). *Chatgpt creates a review article based on the most influential preprints according to altmetric in dimensions.ai*. Retrieved from <https://example.com>
- Qorib, M., Na, S.-H., & Ng, H. (2022, 10). Frustratingly easy system combination for grammatical error correction. In (p. 1964-1974). doi: 10.18653/v1/2022.naacl-main.143
- Rasul, T., Nair, S., Kalendra, D., Robin, M., de Oliveira Santini, F., Ladeira, W. J., ... Heathcote, L. (2023, 1). The role of chatgpt in higher education: Benefits, challenges, and future research directions. *Journal of Applied Learning and Teaching*, 6, 41-56. doi: 10.37074/jalt.2023.6.1.29
- Ray, P. P. (2023). Chatgpt: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, 3, 121–154. Retrieved from <https://doi.org/10.1016/j.iotcps.2023.04.003> doi: 10.1016/j.iotcps.2023.04.003
- Rothe, S., Mallinson, J., Malmi, E., Krause, S., & Severyn, A. (2021, 10). *A simple recipe for multilingual grammatical error correction*. doi: 10.48550/arXiv.2106.03830
- Roubtsova, E. (2015). Chapter two - advances in behavior modeling. In A. M. Memon (Ed.), (Vol. 97, p. 49-109). Elsevier. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0065245814000047> doi: <https://doi.org/10.1016/bs.adcom.2014.10.003>
- Ruan, S., Jiang, L., Xu, Q., Liu, Z., Davis, G. M., Brunskill, E., & Landay, J. A. (2021, 4). Englishbot: An ai-powered conversational system for second language learning. In *International conference on intelligent user interfaces, proceedings iui* (p. 434-444). Association for Computing Machinery. doi: 10.1145/3397481.3450648
- Sottana, A., Liang, B., Zou, K., & Yuan, Z. (2023, 10). Evaluation metrics in the era of gpt-4: Reliably evaluating large language models on sequence to sequence tasks. Retrieved from <http://arxiv.org/abs/2310.13800>
- Sun, X., Ge, T., Wei, F., & Wang, H. (2021, August). Instantaneous grammatical error correction with shallow aggressive decoding. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *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)* (pp. 5937–5947). Online: Association for Computational Linguistics. Retrieved from <https://aclanthology.org/2021.acl-long.462> doi: 10.18653/v1/2021.acl-long.462
- Sutter, G., & Carvalho, P. D. (2023). *Multilingual grammatical error detection and its applications to prompt-based correction*.

- TechTarget. (n.d.). *What is systems development life cycle? — definition from techtarget*. Retrieved from <https://www.techtarget.com/searchsoftwarequality/definition/systems-development-life-cycle> (Retrieved October 13, 2024)
- The UML 2 Class Diagram - IBM Developer. (n.d.). *The uml 2 class diagram - ibm developer*. Retrieved from <https://developer.ibm.com/articles/the-class-diagram/> (Retrieved October 13, 2024)
- Tyen, G., Brenchley, M., Caines, A., & Buttery, P. (2022). *Towards an open-domain chatbot for language practice*. Retrieved from <https://www.englishprofile.org/>
- University of Leeds. (n.d.). *10 reasons why you should do a systematic literature review – and how to conduct one! — blog — centres and institutes*. Retrieved from <https://business.leeds.ac.uk/research-ceric/dir-record/research-blog/2145/10-reasons-why-you-should-do-a-systematic-literature-review-and-how-to-conduct-one> (Accessed: September 11, 2024)
- Use-Case Diagrams - IBM Documentation. (n.d.-a). *Use-case diagrams - ibm documentation*.
- What Is a Work Breakdown Structure (WBS) In Project Management? (n.d.). *What is a work breakdown structure (wbs) in project management?* Retrieved from <https://www.projectmanager.com/guides/work-breakdown-structure> (Retrieved October 13, 2024)
- What is TensorBoard? — ArcGIS API for Python. (n.d.). *Monitor model training with tensorboard*. Retrieved from <https://developers.arcgis.com/python/latest/guide/monitor-model-training-with-tensorboard/> (Retrieved October 13, 2024)
- What is agile scrum? with proven agile scrum techniques. (n.d.). Retrieved from https://clickup.com/blog/agile-scrum/?utm_source=google-pmax&utm_medium=cpc&utm_campaign=gpm_cpc_arlv_nnc_pro_trial_all-devices_tcpa_lp_x_all-departments_x_pmax&utm_content=&utm_creative=_____&gad_source=1&gclid=Cj0KCQjw6oi4BhD1ARIsAL6pox08iUt8j60_y7eeCF0JbLCSZKik2c1htJ6MssUd2B4bgYwGibLhMwoaAqIAEALw_wcB (n.d.)
- What is project management and what are the types? (n.d.). Retrieved from <https://www.investopedia.com/terms/p/project-management.asp> (Retrieved October 13, 2024)
- Workshops, I. T. . D. T. (n.d.). *What is lucidchart and how to use it for visual collaboration*. Retrieved from <https://www.innovationtraining.org/what-is-lucidchart-and-how-to-use-it-for-visual-collaboration/> (Retrieved October 13, 2024)

- Wu, H., Wang, W., Wan, Y., Jiao, W., & Lyu, M. (2023, 3). Chatgpt or grammarly? evaluating chatgpt on grammatical error correction benchmark. Retrieved from <http://arxiv.org/abs/2303.13648>
- Yang, X., Li, Y., Zhang, X., Chen, H., & Cheng, W. (2023, 2). Exploring the limits of chatgpt for query or aspect-based text summarization. Retrieved from <http://arxiv.org/abs/2302.08081>
- Yoon, S., Park, S., Kim, G., Google, J. C., Park, K., Softlyai, G. K., ... Oh, A. (n.d.). *Towards standardizing korean grammatical error correction: Datasets and annotation* (Vol. 1). Long Papers. Retrieved from <https://forms.gle/kF9pvJbLGvnh8ZnQ6>
- Younis, H. A., Mohammed, O., Muthmainnah, Sahib, T. M., Akhtom, D., Hayder, I. M., ... Shahid, M. (2023, October). Chatgpt evaluation: Can it replace grammarly and quillbot tools? *British Journal of Applied Linguistics*, 3(2), 34-46. Retrieved from <https://doi.org/10.32996/bjal.2023.3.2.4> doi: 10.32996/bjal.2023.3.2.4
- Zeng, M., Kuang, J., Qiu, M., Song, J., & Park, J. (2024). *Evaluating prompting strategies for grammatical error correction based on language proficiency*. Retrieved from <http://arxiv.org/abs/2402.15930> (Accessed: 13 October 2024)
- Zhang, Y., Cui, L., Zhao, E., Bi, W., & Shi, S. (2023, December). RobustGEC: Robust grammatical error correction against subtle context perturbation. In H. Bouamor, J. Pino, & K. Bali (Eds.), *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 16780–16793). Singapore: Association for Computational Linguistics. Retrieved from <https://aclanthology.org/2023.emnlp-main.1043> doi: 10.18653/v1/2023.emnlp-main.1043
- Zhang, Y., & Goyal, R. (2022). Syngec: Syntax-enhanced grammatical error correction with a tailored gec-oriented parser. *ArXiv, abs/2210.12484*. Retrieved from <https://api.semanticscholar.org/CorpusID:249009330>
- Zhao, H., Wang, B., Wu, D., Che, W., Chen, Z., & Wang, S. (2022, 8). Overview of ctc 2021: Chinese text correction for native speakers. Retrieved from <http://arxiv.org/abs/2208.05681>
- Zhou, H., Liu, Y., Li, Z., Zhang, M., Zhang, B., Li, C., ... Huang, F. (2023a, 10). Improving seq2seq grammatical error correction via decoding interventions. Retrieved from <http://arxiv.org/abs/2310.14534>
- Zhou, H., Liu, Y., Li, Z., Zhang, M., Zhang, B., Li, C., ... Huang, F. (2023b, December). Improving Seq2Seq grammatical error correction via decoding interventions. In H. Bouamor, J. Pino, & K. Bali (Eds.), *Findings of the association*

for computational linguistics: Emnlp 2023 (pp. 7393–7405). Singapore: Association for Computational Linguistics. Retrieved from <https://aclanthology.org/2023.findings-emnlp.495> doi: 10.18653/v1/2023.findings-emnlp.495

Ömer AYDIN, & KARAARSLAN, E. (2023, 9). Is chatgpt leading generative ai? what is beyond expectations? *Academic Platform Journal of Engineering and Smart Systems*, 11, 118-134. doi: 10.21541/apjess.1293702