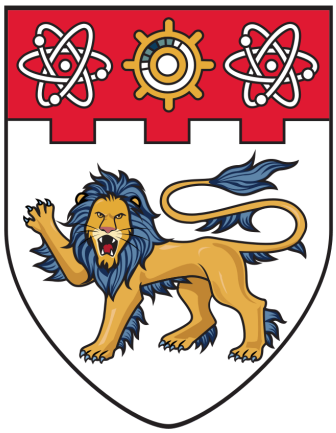

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

PUNCTUATION RESTORATION FOR SPEECH TRANSCRIPTS USING LARGE LANGUAGE MODELS

Liu Changsong

Project Supervisor: Assoc. Prof Chng Eng Siong

School of Computer Science and Engineering
Academic Year 2023/2024

NANYANG TECHNOLOGICAL UNIVERSITY

SCSE23-0753
PUNCTUATION RESTORATION FOR SPEECH TRANSCRIPTS USING LARGE LANGUAGE MODELS

Submitted in Partial Fulfillment of the Requirements
for the Degree of Bachelor of Computer Science
of the Nanyang Technological University

by

Liu Changsong

School of Computer Science and Engineering
Academic Year 2023/2024

Abstract

This thesis explores punctuation restoration in speech transcripts using Large Language Models (LLMs) to enhance text readability and comprehension. We focus on the efficacy of LLMs, specifically XLM-RoBERTa and Llama-2. The primary contributions include the refinement of the existing XLM-RoBERTa model and the fine-tuning of Llama-2, which possesses 13 billion parameters, using several advanced techniques. For the XLM-RoBERTa model, we implement an evaluation pipeline and utilize a model checkpoint ensemble technique that improved its F1-Score by 3%. The fine-tuned Llama-2 model incorporates prompt engineering and Low-Rank Adaptation (LoRA), achieving an F1-Score of 0.73. This score indicates performance that is comparable and even superior across all punctuation classes compared to Google's state-of-the-art Gemini model. Additionally, this project develops and details the fine-tuning processes, data processing strategies, and standardized evaluation methodologies for different LLMs. Our experimental analysis provides a thorough evaluation of model performance and draws meaningful conclusions. Based on the refined model architecture and research conducted in this project, two papers have been published and accepted to ACIIDS and ICAICTA conferences. Future work will extend the scope of LLM evaluations to include additional datasets and will focus on further refining and fine-tuning the models to address challenges that emerge during our experiments.

Acknowledgments

I would like to express my deepest gratitude to Associate Professor Chng Eng Siong, my project supervisor, for his invaluable guidance, encouragement, and expertise throughout the duration of this project. His insights and suggestions have been crucial in shaping both the direction and execution of this research.

I am also immensely grateful to my project mentor, Senior Researcher Ho Thi Nga, for her dedicated mentorship, support, and invaluable contributions. Her expertise and practical advice have been pivotal in navigating the complexities of this project.

My appreciation also extends to High Performance Computing Centre of Nanyang Technological University Singapore and National Supercomputing Centre Singapore, for providing the resources and facilities which have been instrumental in the progress of this work.

Finally, I must acknowledge the invaluable contributions of the open-source community, whose discussions and perspectives have enriched my understanding and approach to this research.

Contents

List of Tables

List of Figures

List of Codes

Chapter 1

Introduction

1.1 Background

Recent advancements in speech recognition technologies have significantly improved the quality of transcribed texts. Yet, the lack of punctuation in these raw transcripts continues to be a major hurdle, affecting readability and the preservation of intended meanings. To address this issue, this project aims to leverage the potential of Large Language Models (LLMs) to insert punctuation accurately, thereby enhancing the comprehensibility of transcribed texts.

1.2 Motivations

The prowess of LLMs in various Natural Language Processing (NLP) tasks, underscored by their expansive parameter scales—ranging from millions to trillions—sets the foundation of our research. We are particularly interested in assessing and enhancing the punctuation restoration abilities of cutting-edge LLMs for speech transcripts. By doing so, this project aims to evaluate whether smaller LLMs can be optimized to match the performance of their state-of-the-art counterparts in punctuating speech transcripts effectively.

1.3 Objectives and Scope

This project's core goal is to develop an LLM that excels in restoring punctuation in speech transcripts with high accuracy. Our approach involves evaluating different LLMs, with a special focus on refining transformer models such as XLM-RoBERTa and fine-tuning LLMs like Llama-2 13B using the Low-Rank Adaptation (LoRA) technique. By improving the models' ability to comprehend

and punctuate transcribed texts, we aim to significantly enhance their coherence and readability.

The project's scope includes refining the XLM-RoBERTa model, applying innovative fine-tuning techniques to the Llama-2 13B model, and experimenting with various leading-edge LLMs. Additionally, we will analyze the model's performance on diverse datasets, highlighting noteworthy observations and insights gained throughout the experimentation phase.

1.4 Contributions

This project enriches the NLP field by advancing punctuation restoration models for speech transcripts. By customizing and refining an off-the-shelf XLM-RoBERTa model - through modifying the number of punctuation classes, refining the tokenization process, developing an evaluation pipeline, and incorporating detokenization for output at word level - we enhance the model's usability. Moreover, our fine-tuning of the Llama-2 13B model with LoRA and prompt engineering achieves performance on par with Google's Gemini-pro, one of the most advanced models to date. Through comprehensive performance analyses, this research not only benchmarks various models but also provides critical insights and analyses, potentially guiding future advancements in the field.

Two papers [3] and [4] have been published and accepted to ACIIDS and ICAICTA conferences based on the refined model architecture and research conducted in this project.

1.5 Thesis Structure

Chapter 1 sets the stage for the thesis, introducing the motivation behind the research, the problem statement, and the objectives aimed to be achieved. It briefly outlines the challenges associated with punctuation restoration in speech transcripts and the potential of Large Language Models (LLMs) in addressing these challenges, which motivates this project.

Chapter 2 provides a comprehensive background on punctuation restoration, including the approaches used, datasets relevant to the research, and evaluation metrics for assessing performance. This chapter lays the groundwork for under-

standing the complexity of the problem and the metrics by which solutions are evaluated.

Chapter 3 delves into the role of LLMs in punctuation restoration, starting with a background on LLMs in chapter 3.1, their definitions, and their applications in this specific domain. It then surveys examples of LLMs that have been applied to punctuation restoration in chapter 3.2, including BERT, XLM-RoBERTa, GPT-3, Gemini, and Llama-2, alongside a discussion on various fine-tuning techniques in chapter 3.3 such as Low-Rank Adaptation (LoRA), Prompt Tuning, Adapters, and Full Model Fine-tuning.

Chapter 4 focuses on the practical aspects of applying and fine-tuning LLMs specifically for punctuation restoration, which serves as the innovation of this research project. This includes refinements to model architectures, implementation details, and the challenges and solutions encountered. Models such as XLM-RoBERTa and Llama-2 are highlighted respectively in chapter 4.1 and chapter 4.2, with specific attention to techniques and outcomes of fine-tuning processes.

Chapter 5 describes the datasets used in this research, detailing their characteristics in chapter 5.1, the preprocessing steps taken, and how the output of LLMs was postprocessed in chapter 5.2. It also elaborates on the evaluation framework adopted for this study in chapter 5.3, discussing the metrics used and the rationale behind the chosen methods.

Chapter 6 presents the three experiments conducted and a detailed analysis of the results. This chapter is divided into sections based on different experimental scenarios, such as baseline model performance in chapter 6.1, the impact of prompt engineering on Llama-2 in chapter 6.2, and comparisons between fine-tuned Llama-2, Gemini, and baseline models in chapter 6.3. Each section discusses the setup, performance comparisons, and specific analyses.

Chapter 7 summarizes the key findings of the research, the contributions to the field of NLP, and particularly to punctuation restoration in speech transcripts. It also discusses the limitations of the current study and proposes directions for future research, suggesting areas where further improvements and explorations could be made.

Chapter 2

An Overview of Punctuation Restoration

This chapter provides an overview of the methodologies, datasets, and evaluation metrics involved in punctuation restoration for speech transcripts. Punctuation restoration is a critical task for enhancing the readability and structural integrity of speech transcripts generated by ASR systems. In section 2.1, we conduct a literature review of various approaches to this challenge, including prosodic feature-based models, lexical feature-based models, deep learning architectures like LSTM and RNNs with attention mechanisms, transformer-based models, and the application of Large Language Models (LLMs). Additionally, section 2.2 reviews popular datasets and their suitability for training and testing punctuation restoration models. In section 2.3, the evaluation of model performance is discussed through metrics like Precision, Recall, F1 Score, and Error Rate.

2.1 Approaches

Punctuation restoration in speech transcripts has evolved through the utilization of both prosodic and lexical features. Prosodic features, extracted from audio signals, include elements like pause duration and pitch. Earlier research, such as in [5] and [6], leveraged Long Short-Term Memory (LSTM) networks, incorporating prosodic elements like pause durations for punctuation. This approach was further refined by subsequent studies, like [7], which introduced bidirectional LSTM cells and an attention mechanism for enhanced performance. However, the dependence on prosodic features poses challenges due to variations in individual speech patterns.

Alternatively, lexical features have been the focus of different models, including Hidden Markov Models (HMM) [8], Conditional Random Fields (CRF) [9], and attention-based deep learning methods [10]. These methods, reliant solely on text processing, are simpler to implement than prosody-based approaches. Some studies, like [11], have shown improved performance by integrating both lexical and prosodic features, thus balancing the limitations of each method. For this research, the focus is on lexical features, due to their straightforward implementation and effectiveness.

Besides leveraging LSTM networks, deep Recurrent Neural Networks (RNNs) with layer-wise multi-head attentions [12] have also been explored for punctuation restoration, indicating the ongoing exploration of deep learning architectures beyond the conventional LSTM models. This highlights a trend towards leveraging more complex models that can better capture the features of speech for punctuation prediction.

Another research [13] involves data generation and reinforcement learning to boost punctuation restoration. This approach signifies a shift towards more adaptive training methods that can further refine the performance of models on this task. By generating data and using reinforcement learning, models can potentially better understand and predict punctuation in speech transcripts, especially in challenging or ambiguous contexts.

Advancements in punctuation restoration in speech transcripts have also leveraged transformer-based models and innovative training approaches to enhance performance. [14] involves the use of external data to improve punctuation restoration. Researchers have experimented with data annotation strategies for Automatic Speech Recognition (ASR) data, including sampling a large number of utterances from business phone calls, to address the challenges posed by noisy human-to-human conversations. A two-stage fine-tuning approach, utilizing BERT models, has shown promise in adapting these models more effectively to the punctuation restoration task. The approach involves initial fine-tuning on a dataset similar to the target domain followed by further fine-tuning on the target dataset itself, demonstrating improved performance across various punctuation marks. [15] explores various transformer-based models for punctuation restoration including BERT, RoBERTa, ALBERT, DistilBERT, and XLM-RoBERTa, focusing on both high-resource and low-resource languages. They specifically in-

vestigate the effectiveness of these models in handling punctuation restoration tasks, proposing an augmentation strategy to enhance performance. [16] frames the task as a sequence-to-sequence challenge. This approach utilizes T5, a pre-trained encoder-decoder transformer model, offering improved accuracy over traditional classification-based systems. This evolution underscores the transformer models' flexibility and effectiveness in handling linguistic tasks such as punctuation restoration in speech transcripts. [17] and [18] also highlight Transformers' efficiency in speech transcripts, attributing their success to their ability to understand long-range dependencies and context.

In recent developments, the application of large language models (LLMs) for punctuation restoration has shown promising progress. [13] introduces an approach that leverages reinforcement learning and data generation using a pre-trained generative language model, GPT-2, to create synthetic labeled data for punctuation restoration in automatic speech recognition (ASR) texts. [19] offers a novel approach by integrating pretrained Large Language Models (LLMs) with speech data, specifically leveraging models with encoder-decoder or decoder-only architectures. This research stands out by directly applying LLMs to generate fully formatted end-to-end ASR transcriptions, showcasing the potential of pre-trained LLMs to significantly improve the readability and structural integrity of transcribed speech. Their methodology, which surpasses existing ASR models in recognition error rate and formatting accuracy, showcases the relevance and applicability of LLMs in enhancing speech transcription systems. This paper provides a reference point for this project, demonstrating the effectiveness of LLMs in punctuation restoration.

2.2 Datasets

Speech transcripts serve as a foundational component for Automatic Speech Recognition (ASR) systems. The task of punctuation restoration, often a crucial post-processing step for ASR transcripts, addresses the challenge of inserting appropriate punctuation marks, which are typically absent in raw speech transcripts. Therefore, this section reviews the datasets frequently utilized in speech transcription for ASR applications, with a focus on their suitability for punctuation restoration tasks.

The SPGISpeech dataset [20] is an extensive collection of 5,000 hours of professionally-transcribed financial audio, designed for academic research in speech recognition. This dataset stands out because it includes fully formatted transcriptions that are cross-checked for accuracy, including sentence structure, capitalization, punctuation, and denormalization of non-standard words like numbers and acronyms. Such detailed transcription makes SPGISpeech uniquely suited for training end-to-end models that require fully formatted text.

Regarding its relevance to punctuation restoration, SPGISpeech's inclusion of fully formatted transcriptions with proper casing and punctuation directly supports the development of models aimed at this task. By providing a rich resource of transcribed financial audio that mirrors real-world use cases, it offers a solid foundation for training and testing punctuation restoration algorithms. However, this dataset is too domain-specific and may not generalize well to other domains for punctuation restoration.

The LibriSpeech dataset [21] is a substantial corpus utilized widely in the Automatic Speech Recognition (ASR) domain. The dataset contains approximately 1,000 hours of audiobooks from the LibriVox project, primarily derived from Project Gutenberg. The dataset is meticulously segmented and aligned, catering to both 'clean' and 'other' categories to facilitate ASR system performance assessment across varying speech clarity levels. The development and test sets each span around 5 hours of audio. In the context of punctuation restoration, datasets like LibriSpeech offer a foundational platform for training and evaluating ASR systems, which is a precursor step before punctuation restoration.

However, the transcripts provided within the LibriSpeech dataset do not include punctuation, since the primary focus is on recognizing and transcribing spoken words rather than on restoring the exact formatting of written text, which includes punctuation and capitalization. The absence of punctuation in ASR datasets like LibriSpeech simplifies the transcription task, but is not as suitable for punctuation restoration tasks.

The Libriheavy dataset [22] provides a large corpus of 50,000 hours of read English speech derived from LibriVox. This dataset is the largest freely available corpus of speech with annotations, distinguishing itself from other datasets by including detailed supervision such as punctuation, casing, and text context. The creation

of Libriheavy involved an innovative pipeline for aligning audio segments with their corresponding texts from the previously published Librilight, adjusting for variances in audio duration by segmenting long audios into manageable 30-second pieces with overlaps. This was followed by a complex alignment process involving close matches and longest increasing pairs to align the transcribed audio with the original book text. Notably, the segmentation process was designed to break long audio into shorter segments suitable for ASR training, utilizing punctuation as a guide for segment boundaries, thus directly linking the dataset's construction to punctuation restoration.

Apart from the scale, Libriheavy is also useful in tasks requiring detailed linguistic features like punctuation restoration. The dataset's inclusion of punctuation and casing information directly supports the development of punctuation restoration models by providing them with high-quality, richly annotated training data. Furthermore, the dataset has been made available on GitHub to encourage widespread use for various speech recognition and linguistic tasks. Therefore, this project utilized Libriheavy dataset for training, evaluating, and fine-tuning.

The IWSLT datasets, specifically the TED Talks dataset, are also suitable for punctuation restoration in speech transcripts. The IWSLT 2012 TED Talks dataset is a common benchmark for punctuation restoration models. It contains 1,066 unique transcripts of TED talks, totaling 2.4 million words. This dataset is split into a training set of 2.1 million words (87.5%), a validation set of 296,000 words (12.3%), and a test set of 12,000 words. This makes it a substantial resource for developing and testing punctuation restoration models, as it provides a rich set of real-world spoken language data. The dataset is used widely in research such as [23] to improve the performance of punctuation restoration methods, indicating its relevance and utility for punctuation restoration.

2.3 Evaluation Metrics

The efficacy of punctuation restoration models, particularly those powered by large language models (LLMs), is critically dependent on robust evaluation frameworks which provide quantifiable measures to assess the accuracy, reliability, and overall performance of these models in restoring punctuation.

The most widely adopted metrics for evaluating punctuation restoration models

include Precision, Recall, and the F1 Score. Precision measures the proportion of correctly predicted punctuation marks out of all predicted marks, emphasizing the model's accuracy in placement. Recall assesses the proportion of correctly predicted punctuation marks out of all actual punctuation marks, highlighting the model's completeness in restoring necessary punctuation. The F1 Score provides a harmonic mean of Precision and Recall, offering a balanced measure of the model's overall performance. These metrics have been widely used in studies such as by [7] and [13] to evaluate the performance of neural network-based models in punctuation prediction tasks. These evaluation metrics are also employed in this project to analyze the performance of a model.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.2}$$

$$F1 \text{ score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.3}$$

Another metric occasionally employed is the Error Rate, which quantifies the overall error in punctuation prediction, including omissions, incorrect insertions, and substitutions. This evaluation metric, Slot Error Rate to be specific, is used in [24] to capture all types of potential errors directly impacting the transcript's readability.

In this section, we will also introduce Word Error Rate (WER) and Sentence Error Rate (SER). WER is a common metric used to measure the performance of a speech recognition or transcription system. It compares the transcription produced by the system with the groundtruth transcription to quantify the errors made by the system. SER measures the proportion of sentences that contain any errors compared to the total number of sentences in the reference transcription. Essentially, if a sentence in the generated transcription differs from its reference, it is counted as an error. Even though they are not directly used for evaluating punctuation restoration performance, they are employed in this project to demonstrate the generative nature of LLMs and how this nature affects the punctuation restoration performance.

2.4 Summary

Chapter 2 has systematically explored the evolution and current state of punctuation restoration in speech transcripts, emphasizing both methodological advancements and the critical role of datasets and evaluation metrics. We begin with an analysis of various approaches to punctuation restoration, tracing from prosody-based models to LLMs. This progression underscores a significant shift towards leveraging more complex and adaptable models, such as transformer-based and sequence-to-sequence models, to enhance the performance of punctuation restoration systems.

Regarding datasets, chapter 2 highlights the importance of richly annotated and domain-specific resources like SPGISpeech, LibriSpeech, Libriheavy, and IWSLT datasets. These datasets provide the foundation for training and evaluating punctuation restoration models and reflect the diverse applications within the field. Evaluation metrics such as Precision, Recall, F1 Score, and Slot Error Rate were examined for their role in assessing model performance.

This chapter serves as a bridge to chapter 3, which delves into the specifics of LLMs for punctuation restoration. By building on the foundational knowledge presented here, the next chapter aims to offer detailed insights into how LLMs are revolutionizing punctuation restoration, marking a milestone in the field.

Chapter 3

Large Language Models for Punctuation Restoration

This chapter delves into the utilization of LLMs for punctuation restoration. To begin, it offers a background understanding of LLMs in section 3.1, detailing their architecture, operational mechanisms, and transformer technologies in facilitating their language comprehension capabilities in section 3.1.1. This chapter then examines the application of these models in punctuation restoration in section 3.1.2. The chapter then progresses to showcase specific examples of LLMs in section 3.2, including BERT (section 3.2.1), XLM-RoBERTa (section 3.2.2), GPT-3 (section 3.2.3), Gemini (section 3.2.4), and Llama-2 (section 3.2.5). Furthermore, this chapter contains a literature review on various fine-tuning techniques in section 3.3) including Low-Rank Adaptation (LoRA) (section 3.3.1), prompt tuning (section 3.3.2), adapters (section 3.3.3) and full model fine-tuning (section 3.3.4), offering insights into their methodologies, advantages, and application scenarios.

3.1 Background

3.1.1 Definition of Large Language Models (LLMs)

Large Language Models (LLMs) are sophisticated machine learning models that specialize in understanding and generating human language. They are a subset of neural networks trained on extensive text datasets, which include books, articles, and other written documents. This training enables them to learn the statistical patterns of language, including the relationships between words and phrases, making them capable of producing text that is coherent, contextually relevant, and grammatically correct.

LLMs operate using millions to trillions of parameters, which are variables that the models adjust during training to better predict the next word in a sequence. For instance, OpenAI's GPT-3 has 175 billion parameters, and its successor, GPT-4, is reported to have up to 1.7 trillion parameters. These parameters allow LLMs to generate text based on the input they receive, making educated guesses about what comes next in a sentence or document.

One of the core techniques underpinning LLMs is the use of transformer architectures, which allow for the processing of words in relation to all other words in a sentence, rather than in a sequential manner. This is crucial for understanding context and generating text that is both relevant and accurate. LLMs like Generative Pretrained Transformer (GPT), Bidirectional Encoder Representations from Transformers (BERT), Cross-Lingual Language Model RoBERTa (XLM-RoBERTa), and others have been pivotal in advancing the capabilities of natural language processing technologies.

LLMs have a wide range of applications, from generating human-like text for chatbots and virtual assistants to more complex tasks such as translation, summarization, content creation, and even code generation. Their ability to understand and generate language has made them invaluable in both commercial and research settings.

3.1.2 Applications of LLMs in Punctuation Restoration

Large Language Models (LLMs) have been increasingly applied to punctuation restoration tasks, especially in fields such as Automatic Speech Recognition (ASR) and sentiment analysis from social media data [12]. These models, particularly transformer-based ones, are pivotal in improving the readability of text by accurately inserting punctuation marks, which is crucial for further processing and understanding of the text.

In the context of ASR, punctuation restoration is essential as ASR systems typically transcribe speech into text without any punctuation marks, making the text difficult to read and understand. The absence of punctuation can hinder the performance of downstream NLP tasks such as sentiment analysis, where the correct interpretation of the sentiment conveyed in a sentence can significantly depend on its structure and punctuation. For example, studies including [12] have shown

that restoring punctuation in text can enable more accurate sentiment analysis by allowing the sentiment analysis models to consider the sentiment polarity of individual sentences within a longer text segment.

Moreover, [25] states that the application of LLMs in punctuation restoration not only benefits readability and sentiment analysis but also enhances machine translation effectiveness and the general processing of text data. This is because properly punctuated text is easier for both humans and machines to interpret, ensuring that the grammatical structure and meaning of sentences are preserved. Various approaches, including the use of regular expressions, conditional random fields, and natural language toolkits, have been employed to improve the accuracy of punctuation restoration, emphasizing its importance in the broader field of NLP.

3.2 Examples of LLMs for Punctuation Restoration

3.2.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a transformative model in the field of NLP. BERT has previously set new benchmarks for a wide range of NLP tasks, including but not limited to text classification, question answering, named entity recognition, as well as punctuation restoration. The architecture of BERT is illustrated in Figure 3.1. BERT's innovation involves its use of the transformer architecture, specifically the attention mechanism, allowing the model to capture the context of a word based on all other words in a sentence, both to its left and right.

3.2.2 XLM-RoBERTa

Derived from BERT, XLM-RoBERTa [18] represents a milestone in handling multilingual and cross-lingual tasks. This model, an extension of the RoBERTa architecture, is pre-trained on a large, diverse multilingual corpus, making it capable of processing text across various languages. This capability is crucial for tasks like punctuation restoration in multilingual contexts. The choice to utilize XLM-RoBERTa for punctuation restoration results from its robust handling of syntactic structures across different languages. XLM-RoBERTa's comprehensive

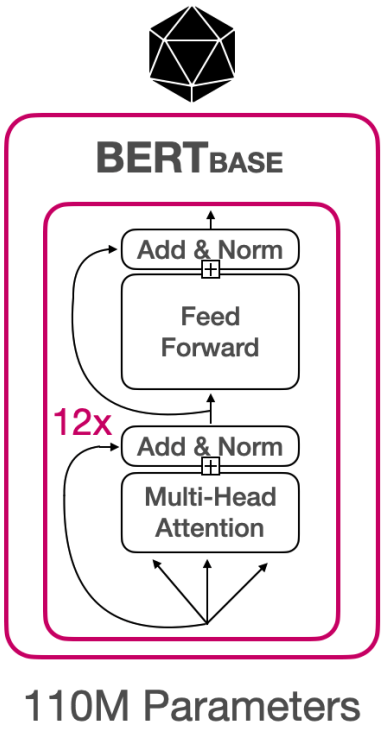

Figure 3.1: Illustration of BERT key architecture [1]

pre-training enables it to better capture language-specific punctuation rules, an essential aspect of accurately transcribing speech into text. The model's ability to generalize across languages while maintaining relatively high quality understanding makes it an ideal candidate for addressing the challenges in multilingual punctuation restoration.

3.2.3 GPT3

GPT-3 [16], an autoregressive language model with 175 billion parameters, exemplifies the potential of massive LLMs in understanding and generating text based on a few examples or even a single example of a task. This model sets itself apart by not requiring any gradient updates or fine-tuning when applied to new tasks; instead, it uses in-context learning, where tasks and demonstrations are specified purely via text interactions with the model. This approach is significantly different from traditional methods, which rely heavily on task-specific fine-tuning datasets. It opens up new avenues for research, including in the context of punctuation restoration, where such models could potentially be applied to understand and correct punctuation with few examples. However, due to the resource limitation and non-disclosure of models, it is impossible to train and

fine-tune the GPT3 models.

3.2.4 Gemini

Gemini model [26], with its versions Ultra, Pro, and Nano, contributes to AI advancements, particularly in its ability in the Massive Multitask Language Understanding (MMLU) benchmark. This achievement showcases Gemini's exceptional capability in understanding and solving complex problems across diverse subjects. Gemini's sophisticated understanding and multimodal approach could also extend to punctuation restoration, leveraging its deep comprehension of language and context to enhance the accuracy and efficiency of punctuation restoration in text and speech transcripts. However, the model is not publicly available and can only be accessed via Application Programming Interface (API).

3.2.5 Llama-2

Llama-2 model [27], also marks a significant advancement, offering a comprehensive suite of pre-trained and fine-tuned models that range from 7 billion to 70 billion parameters. This work distinguishes itself by optimizing Llama 2-Chat models for dialog use cases, demonstrating superior performance over existing open-source chat models across various benchmarks. This model's open release, together with its performance, positions Llama-2 as a satisfying substitute for closed-source models in this project, offering a valuable resource for conducting research on restoring punctuations.

3.3 Fine-tuning Techniques for LLMs

3.3.1 Low-Rank Adaptation (LoRA)

LoRA [28], which stands for Low-Rank Adaptation, is a fine-tuning technique for LLMs that emphasizes efficiency and effectiveness. As shown in Figure 3.2, it operates by modifying the fine-tuning process to freeze the original model weights and apply changes to a separate set of low-rank matrices that are then added to the original parameters. This approach significantly reduces the number

of parameters that need training, making the process faster and less resource-intensive.

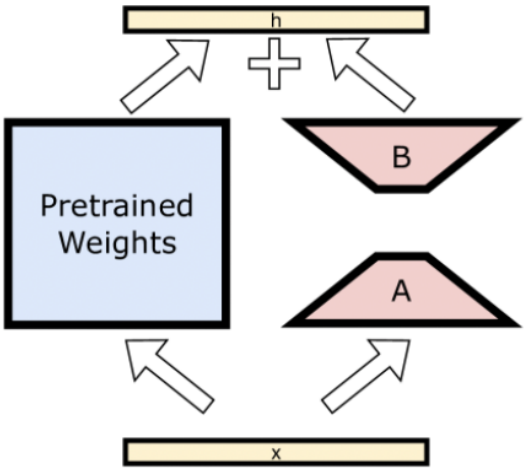

Figure 3.2: Illustration of LoRA key architecture [2]

One of the primary advantages of using LoRA is its efficiency in training and adaptation. Unlike traditional fine-tuning methods that require updating all model parameters, LoRA introduces low-rank matrices that modify only a subset of the original model's weights. This selective updating approach streamlines the adaptation process, making it quicker and more efficient. It is especially beneficial for applications requiring regular updates or adaptations, such as adapting a model to specialized domains or continuously evolving datasets.

In addition, LoRA offers a reduced computational resources requirement, enabling the fine-tuning of LLMs like Llama-2 with significantly lower memory and processing power. This reduction is vital for practical applications since fully retraining LLMs is often impractical due to resource limitations. Furthermore, LoRA preserves the integrity of pre-trained model weights, ensuring that the core structure and knowledge embedded in the model are largely maintained while adapting to specific tasks or datasets.

3.3.2 Prompt Tuning

Prompt tuning [29] is an efficient and cost-effective method for adapting LLMs to new downstream tasks without the need for retraining or updating the model's weights. This technique has become increasingly important as larger and larger models come into presence, making traditional fine-tuning computationally in-

tensive and resource-demanding.

Prompt tuning works by introducing specific cues or prompts that provide task-specific context to the model. This method effectively guides the model towards making the desired decision or prediction without altering the model's core parameters. It's particularly useful for applications where data is scarce or when there's a need to quickly adapt a model to a specific task without extensive retraining.

One of the major benefits of prompt tuning is its efficiency in both training and adaptation. It significantly reduces the computational resources required for fine-tuning LLMs by focusing only on a small set of embedding parameters for a particular task. This allows for the customization of the model for downstream tasks with minimal memory and storage requirements compared to traditional fine-tuning methods.

3.3.3 Adapters

Adapters [2] is a fine-tuning technique used in the context of Large Language Models (LLMs) that involves inserting small, trainable modules between the layers of a pre-trained model. These modules, or adapters, allow for the efficient adaptation of LLMs to specific tasks without modifying the original model's weights. The primary concept behind adapters is to retain the general knowledge captured during pre-training while introducing a minimal number of additional parameters to adapt the model to a new task.

Adapters work by adding bottleneck feed-forward layers either in series or parallel with the existing layers of a Transformer-based LLM. The Series Adapter [30] adds these layers in series to each multi-head and feed-forward layer of a Transformer block, as shown in Figure 3.3. Meanwhile, as shown in Figure 3.4, the Parallel Adapter [31] integrates bottleneck feed-forward layers in parallel with these components of the Transformer architecture. This approach ensures that the original parameters of the LLM remain untouched, focusing the fine-tuning process on the adapter layers, which are much smaller in scale compared to the entire model.
Adapter-based fine-tuning allows for the tailored adaptation of large models to specific tasks with minimal computational resources. This technique's efficiency and versatility make it a valuable tool in the LLM fine-tuning toolkit, offering

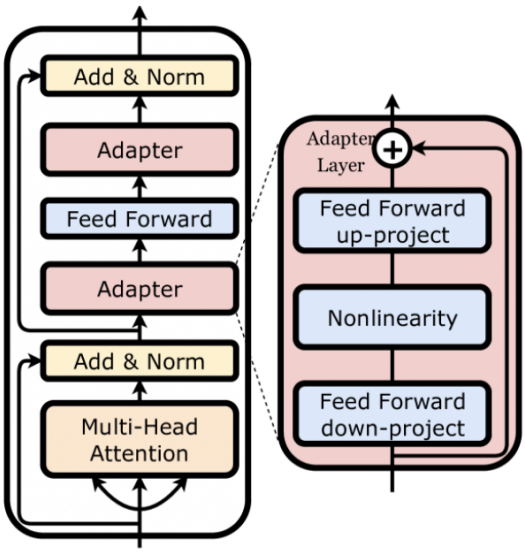

Figure 3.3: Illustration of Series Adapter key architecture [2]

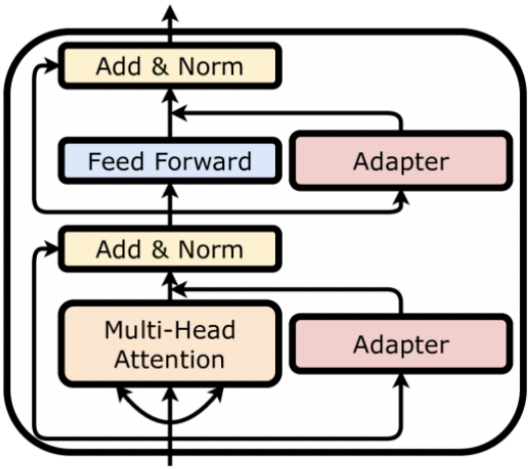

Figure 3.4: Illustration of Parallel Adapter key architecture [2]

a practical solution for enhancing the performance of pre-trained models on targeted tasks while preserving their general capabilities.

3.3.4 Full Model Fine-tuning

Full model fine-tuning [32] is a comprehensive method used for adapting Large Language Models (LLMs) to specific tasks or domains by updating all the parameters of a base model, creating a new version of the model that's specifically adapted to a given task or domain. While this approach allows for a high degree of customization and can significantly improve the model's performance on

specific tasks, it is also resource-intensive. It requires substantial computational power and memory to process and store all the adapted parameters, which is most often impossible for LLMs.

Despite its resource demands, full model fine-tuning is often justified by the benefits it brings. A fine-tuned model can perform a wider range of tasks more accurately than its pretrained counterpart, making it highly valuable for applications that require domain-specific knowledge or custom functionalities. However, challenges such as the potential for catastrophic forgetting, where the model loses its ability to perform tasks it was capable of before fine-tuning, could possibly arise.

3.4 Summary

Chapter 3 has systematically presented the critical role of LLMs in punctuation restoration. It begins with an introduction to LLMs, emphasizing their architectural sophistication, extensive training on diverse datasets, and the impact of transformer architectures on their operational efficacy.

The discussion then extends to specific LLM examples such as BERT, XLM-RoBERTa, GPT-3, Gemini, and Llama-2. These models demonstrate the possibilities of applying LLMs to address the challenges of accurately inserting punctuation in speech transcription.

The chapter further explores fine-tuning techniques that can enhance LLMs' performance on specific tasks, including LoRA, prompt tuning, adapters, and full model fine-tuning. These methodologies offer various ways to optimize LLMs, balancing between efficiency, resource requirements, and customization levels to meet specific task demands.

Chapter 3 explains the critical role of LLMs in punctuation restoration and sets the stage for chapter 4, which will delve deeper into refining and fine-tuning LLMs for enhanced performance in punctuation restoration tasks. The subsequent chapter will detail the methodology and innovation for this project, providing a focused exploration of optimizing LLMs for punctuation restoration.

Chapter 4

Methodology and Innovations

This chapter delves into the methodologies and innovations applied to refine and fine-tune LLMs, specifically XLM-RoBERTa in section 4.1 and Llama-2 in section 4.2, for punctuation restoration. Section 4.1 outlines the architecture of XLM-RoBERTa (section 4.1.1), the enhancements which include model architecture refinements (section 4.1.2) and implementation of a model checkpoint ensemble strategy (section 4.1.3), as well as the code implementation of core algorithm (section 4.1.4). Section 4.2 outlines the background of Llama-2's fine-tuning (section 4.2.1). The focus of fine-tuning is on experimenting with prompt engineering (section 4.2.2) and leveraging LoRA for efficient model adaptation (section 4.2.3) to boost its punctuation restoration capability. Lastly, section 4.2.4 introduces the implementation of the fine-tune pipeline of Llama-2.

4.1 XLM-RoBERTa

4.1.1 Model Architecture

This project employs the off-the-shelf model architecture proposed by [23] for multilingual punctuation which consists of a pre-trained XLM-RoBERTa (XLM-R) model followed by a classifier. The architecture is illustrated in 4.1. The XLM-RoBERTa model generates multiple embeddings in a self-supervised way and contains multiple self-attention transformer-encoder layers. The input data is tokenized using the multilingual tokenizers, i.e. SentencePiece and Jieba, and subsequently the <mask> token is added after every word. The classifier comprising two fully connected (FC) layers predicts punctuation for each masked token. The output contains tokens that represent punctuations like commas, periods, or spaces.

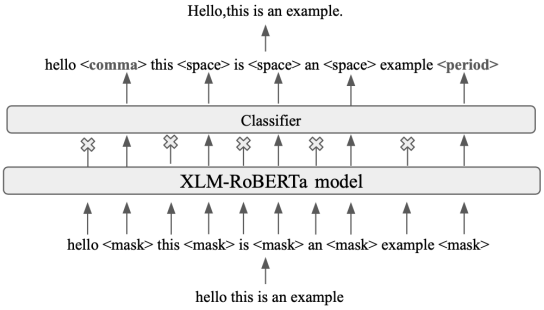

Figure 4.1: XLM-RoBERTa architecture

4.1.2 Existing Model Architecture Refinements

Several refinements have been introduced by this project to the base model architecture and its implementation to cater to diverse punctuation prediction needs and improve overall performance.

Initially, the model was designed to predict four punctuation classes: comma, period, question mark, and space (representing the absence of punctuation). To offer more flexibility in this project, the model has been adapted to include versions that predict three punctuation marks (comma, period, and space) and one that can identify five (comma, period, exclamation mark, question mark, and space). This adaptation allows for customized punctuation prediction based on specific requirements.

Furthermore, the tokenization process in the original model assigned a placeholder value of 6 to represent spaces. This posed a challenge as the digit 6 also appeared within word tokens, potentially compromising the following punctuation masking process and, consequently, the model's accuracy. To address this, the placeholder value for spaces has been updated to 250004, a number not found in any word tokens, improving the model's reliability.

Additionally, the original script focused predominantly on training aspects, such as saving epoch checkpoints and resuming training sessions. However, it lacked functionality for evaluating the model against external datasets. Enhancements to the script now include the capability to define a testing dataset and implement a testing pipeline. This allows for the assessment of the model's performance after each training epoch, providing insights into its efficacy and areas for further refinement.

Lastly, the original script evaluates performance at the token level rather than the word level, omitting the ability of the model to generate punctuated sentences from the test dataset. This limitation has been addressed by incorporating a detokenization process. Chapter 5.3.2 delves deeper into this improvement, discussing the standardization of evaluation methods across different models.

4.1.3 Model Checkpoint Ensemble

In traditional approaches, the optimal punctuation restoration model is chosen based on the highest performance metrics observed on development datasets among the training checkpoints. This project explores an innovative strategy, leveraging the aggregation of multiple checkpoints from various epochs within a single training session to enhance the model's effectiveness. This involves systematically saving the model's state at the conclusion of each epoch across the entire training, which usually contains 10 epochs. Subsequently, an evaluation of each epoch's performance on development datasets is conducted to select the best-performing checkpoints. These checkpoints are then aggregated, employing both weighted and arithmetic averaging methods to refine the final model. Weighted averaging assigns greater significance to checkpoints from epochs with better performance, whereas arithmetic averaging assigns equal importance across all selected epochs, with the aim to consolidate the strengths of multiple training phases into a singular, improved model.

4.1.4 Implementation

The core of this approach is based on [23] with refinements discussed above and involves fine-tuning the XLM-RoBERTa model, a transformer-based model optimized for multilingual understanding, for the task of punctuation restoration in speech transcripts. The following outlines the key aspects:

Data Preparation and Loading:

Data is read from pickle (.pkl) files and converted into datasets suitable for training and validation. Data loaders are prepared using custom dataset classes and collate functions to handle batch processing.

Model and Tokenizer Configuration:

The BertPunctuator model and preprocessor are utilized to handle XLM-RoBERTa's specific requirements. The tokenizer is configured to match the model family, set-

ting appropriate tokens for masking and padding.

Training Setup:

The model is configured for GPU acceleration if available, ensuring efficient training. Datasets are loaded into the model with punctuations masked, a technique crucial for teaching the model to predict missing punctuation.

```
    def full_mask_out(self,text,targets):
        """
        mask every single punctuation mark
        """
        period = self._preprocessor._tokenizer.encode(".",
            add_special_tokens=False)[1]
        chinese_period = self._preprocessor._tokenizer.encode("
              ",add_special_tokens=False)[1]
        comma = self._preprocessor._tokenizer.encode(",",
            add_special_tokens=False)[1]
        question = self._preprocessor._tokenizer.encode("?",
            add_special_tokens=False)[0]
        exclaimation = self._preprocessor._tokenizer.encode("!",
            add_special_tokens=False)[0]
        mask_token_number = self._preprocessor._tokenizer.encode(
            self.mask_token,add_special_tokens=False)[0]
        self.mask_token_number = mask_token_number
        space_placeholder = 250004

        values_mask = [period,comma,chinese_period,
            space_placeholder,question,exclaimation]
        mask =  sum(text == i for i in values_mask)
        mask &=  (targets != -1).type(torch.uint8)
        mask = mask.bool()

        text[mask] = mask_token_number
        return text
```

Code 4.1: XLM-RoBERTa punctuation masking code

Training Loop:

The training process involves iterating over batches of data, applying the masking strategy, computing predictions, and backpropagating the losses. A linear learning rate scheduler is employed to optimize the learning process.

Validation and Evaluation:

The validation step includes the evaluation of the model on a separate dataset to assess its performance in terms of loss and other metrics like precision, recall, and F1-score. Model performance is logged using TensorBoard and Weights & Biases (wandb) for visualization and monitoring.

4.2 Llama-2

4.2.1 Background for Fine-tuning Llama-2

In Chapter 3, we introduced Llama-2, a sophisticated open-source variant of large language models optimized for a variety of natural language processing tasks, including the challenge of punctuation restoration. To augment its capabilities for this specific task, this project leverages the Low-Rank Adaptation (LoRA) technique together with prompt engineering for fine-tuning the LLM. LoRA, renowned for its efficiency and efficacy, facilitates the fine-tuning of substantial models like Llama-2-13B without requiring vast computational resources. This attribute is particularly crucial and helpful given the model's extensive scale. Concurrently, prompt tuning is employed as a complementary strategy to further amplify the model's performance.

4.2.2 Prompt Engineering for Llama-2

In this project, we focus on fine-tuning the Llama-2 base model using the preprocessed LibriHeavy datasets. The preprocessing steps incorporate not only cleaning and normalization but also the addition of specific prompts before presenting each utterance to the model. This raises an intriguing question: does a detailed and clear prompt enhance the LLM's performance? To investigate, we crafted three distinct versions of prompts, starting with the simplest: "Restore punctuations for the sentences:". During experimentation, it is observed that the fine-tuned Llama-2, as a generative model, alters the input sentences in an undesired manner. These modifications include changing names and performing inverse text normalization. To curb such alterations, a second prompt is introduced: "Restore punctuations for the sentences without changing any words: ". The third prompt is designed to more explicitly define the task, incorporating detailed instructions and examples: "As a master of English grammar and punctuation, your task is to meticulously restore capitalization and insert missing

punctuation marks into the following text without changing any words or adding new content. Ensure the text's original meaning and integrity are preserved, focusing solely on restoring punctuation and capitalization where needed.
Example 1
Example 2
Using the guidance provided by the examples above, proceed to restore capitalization and insert the necessary punctuation marks into the text below:".

The analysis of the model's performance and the outcomes of these prompts are elaborated on in Chapter 5. This examination seeks to understand the influence of prompt specificity on the effectiveness of fine-tuning LLMs for specific text processing tasks.

4.2.3 Fine-tuning Llama-2 with LoRA

Aiming to enhance the punctuation restoration capabilities of the open-source Llama-2 model, this project explores not only prompt engineering but also updating the model's weights with Low-Rank Adaptation (LoRA). Given the extensive scale of LLMs like Llama-2, and constrained by limited time and resources, fully fine-tuning the Llama-2 base model is impractical. LoRA addresses this challenge by updating a selective set of low-rank matrices, significantly reducing the quantity of parameters that require training. The decision to utilize LoRA for fine-tuning the Llama-2 model is also supported by its demonstrated effectiveness in refining LLMs for specialized tasks, as outlined in [12]. Consequently, this project aims to evaluate to what extent a fine-tuned Llama-2 model, employing LoRA, surpasses the performance of the base model in the punctuation restoration task.

4.2.4 Implementation

In the implementation phase, LoRA was integrated into the training pipeline of Llama-2 shown in Figure 4.2. The key steps involved:

Dataset Preparation

The first segment of the code focuses on preparing the dataset for training: The training data are preprocessed into the format of <s>[INST] Restore punctuations for sentences: unpunctuated sentence [/INST] punctuated sentences </END>. The testing data are preprocessed into the format of <s>[INST]

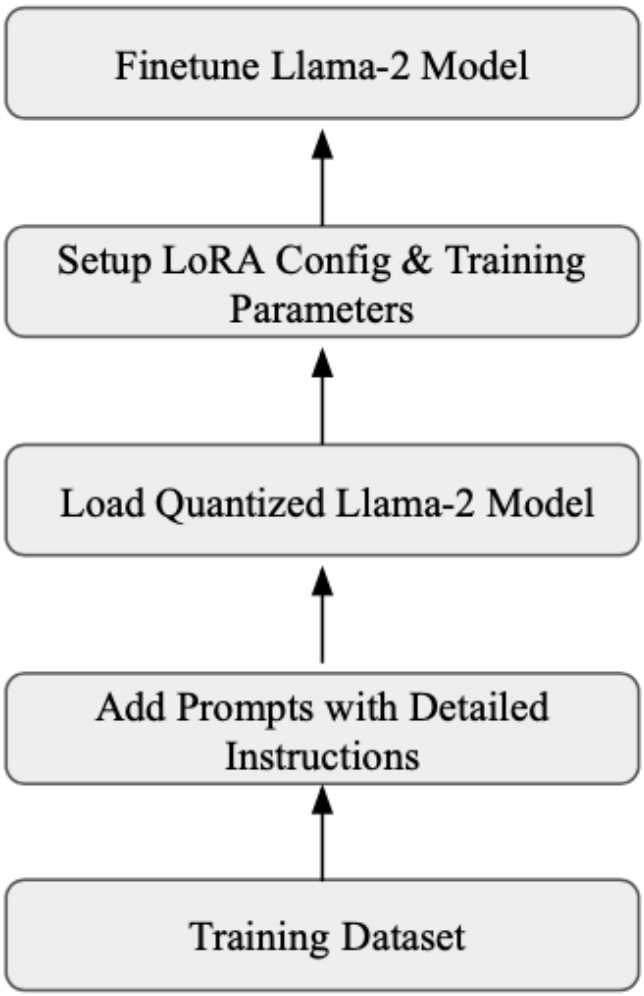

Figure 4.2: Llama-2 Fine-tune Pipeline

Restore punctuations for sentences: unpunctuated sentence [/INST]. The training and testing data are then read from text files containing speech transcripts. The data is then converted into a Dataset object, a format compatible with the training process. This step is crucial as it ensures that the model is correctly trained and evaluated with high efficiency.

```
from datasets import Dataset

training_data={}
testing_data={}
with open("/path/to/train/text/file",'r') as f:
    training_data['text'] = f.readlines()
with open("/path/to/text/file",'r') as f:
    testing_data['text'] = f.readlines()

# Convert dictionaries to datasets
```

```
    training_dataset = Dataset.from_dict(training_data)
    testing_dataset = Dataset.from_dict(testing_data)
```

Code 4.2: Llama-2 fine-tune data preparation code

Model and Tokenizer Configuration

The AutoModelForCausalLM and AutoTokenizer from Hugging Face's Transformers library are used to load the base Llama-2 13B model and its tokenizer. The tokenizer is configured to align with the requirements of Llama-2, setting the pad token and padding side. The model is loaded with specific parameters for quantization, which are vital for managing computational efficiency and memory usage.

```
    from transformers import (
    AutoModelForCausalLM,
    AutoTokenizer,
    BitsAndBytesConfig,
    TrainingArguments,
    pipeline
    )

    # Model and tokenizer names
    base_model_name ='scratch/Llama-2-13b-chat-hf'
    refined_model = "llama-2-chat-13b-FT" #You can give it your
        own name

    # Tokenizer
    llama_tokenizer = AutoTokenizer.from_pretrained(
        base_model_name)
    llama_tokenizer.pad_token = llama_tokenizer.eos_token
    llama_tokenizer.padding_side = "right"

    # Quantization Config
    quant_config = BitsAndBytesConfig(
        load_in_4bit=True,
        bnb_4bit_quant_type="nf4",
        bnb_4bit_compute_dtype=torch.float16,
        bnb_4bit_use_double_quant=False
    )

    # Model
    base_model = AutoModelForCausalLM.from_pretrained(
        base_model_name,
```

```
        quantization_config=quant_config,
        device_map="auto",
    )
    base_model.config.use_cache = False
    base_model.config.pretraining_tp = 1
```

Code 4.3: Llama-2 fine-tune model and tokenizer configuration code

LoRA Configuration

The LoRA technique is applied using the LoRAConfig class, with parameters like lora_alpha, lora_dropout, and r (rank) defined. These parameters determine how the LoRA adaptation modifies the model's weights. The choice of these parameters determines the balance between model flexibility and efficiency.

```
    from peft import LoraConfig,PeftModel, get_peft_model

    # LoRA Config
    peft_parameters = LoraConfig(
        lora_alpha=16,
        lora_dropout=0.1,
        r=8,
        bias="none",
        task_type="CAUSAL_LM"
    )
```

Code 4.4: Llama-2 fine-tune LoRA configuration code

Training Parameters

The TrainingArguments class defines key training parameters, such as the number of epochs, batch size, learning rate, and others. These settings are crucial for optimizing the training process, ensuring effective learning while preventing overfitting.

Training Process

The SFTTrainer class is used for the training process. It takes the model, datasets, LoRA configuration, tokenizer, and training arguments as inputs. The train() method initiates the fine-tuning process, where the model learns to restore punctuation in the given speech transcripts. The model is then saved using save_pretrained, preserving the fine-tuned state for later use.

```
    from trl import SFTTrainer

    # Trainer
    fine_tuning = SFTTrainer(
        model=base_model,
        train_dataset=training_dataset,
        eval_dataset=testing_dataset,
        peft_config=peft_parameters,
        dataset_text_field="text",
        tokenizer=llama_tokenizer,
        args=train_params,
        max_seq_length=1024 #TODO 2048->512
    )

    # Training
    fine_tuning.train()

    # Save Model
    fine_tuning.model.save_pretrained(refined_model)
```

Code 4.5: Llama-2 fine-tune training process code

4.3 Summary

Chapter 4 presents the methodical and innovative steps to refine and fine-tune the XLM-RoBERTa and Llama-2 models for punctuation restoration. It begins by covering XLM-RoBERTa's model architecture and the strategic refinements made to enhance its functionality for diverse punctuation prediction needs, highlighting the model checkpoint ensemble technique as a novel strategy to aggregate the strengths of various training epochs. Additionally, the chapter provides insight into the comprehensive implementation process, covering data preparation, model and tokenizer configuration, and the intricate details of the training loop.

In the latter section, the focus shifts to Llama-2, outlining the background for fine-tuning, the application of prompt engineering, and the integration of the LoRA technique. Prompt engineering for Llama-2 examines the impact of prompt on model performance. Moreover, the utilization of LoRA emerges as a critical strategy for updating Llama-2's weights efficiently, without the need for extensive computational resources.

With the insights and methodologies discussed in this chapter, we will explore datasets and evaluation strategies next. Chapter 5 aims to shed light on the datasets utilized for training and testing, the processes involved in data preparation and output postprocessing, and the evaluation metrics employed, before we can move on to analyze the experiments.

Chapter 5

Datasets and Evaluation

This chapter focuses on the datasets and evaluation methods essential for training and testing LLMs for punctuation restoration. Three diverse datasets are introduced (section 5.1) and meticulously prepared (section 5.2). This chapter details the preprocessing steps designed for each dataset in section 5.2.1. Additionally, it introduces a necessary postprocessing step for aligning LLM outputs with ASR transcripts, crucial for handling generative models' unpredictable alterations in section 5.2.2. Lastly, the evaluation segment (section 5.3) leverages metrics (section 5.3.1) such as precision, recall, and F1 scores to assess the models' performance comprehensively and establishes a unified evaluation algorithm (section 5.3.2).

5.1 Descriptions of the Datasets

This project utilizes three datasets during training and testing the LLMs, they are of different sizes and cover different areas and topics. The statistics of the three datasets are shown in Table 5.1. The details will be introduced in the following paragraphs.

Dataset	train			valid			test		
	word	comma	period	word	comma	period	word	comma	period
NTU-English	1.7M	106K	144K	192K	14.3K	18.3K	43.5K	2.3K	4.1K
NTU-EnMan	1.1M	67.4K	82K	144K	9.9K	10.3K	132K	8.5K	10K
Libriheavy-small	3.0M	199.2K	156.7K	143K	9.3K	7K	152K	10.4K	7.6K

Table 5.1: Statistics of datasets.

5.1.1 NTU-English

This in-house dataset provides a rich compilation of transcribed medical discussions focused on psychiatric evaluations and patient management within a clinical setting. It encompasses a wide array of psychiatric conditions, including depression, suicidal ideation, and conversion disorder, alongside their diagnostic and therapeutic processes. The data is meticulously structured, featuring timestamps and speaker identifiers, which offer a detailed insight into the dynamics of medical consultations. Entirely spoken in English, this dataset is utilized to train models and evaluate their performances.

5.1.2 NTU-EnMan

This in-house dataset is recorded with recording devices and manually transcribed by human transcribers. NTU-EnMan contains conversational speech between two Singaporeans. The speakers may speak in English, Mandarin, or codeswitch between the two languages naturally during the entire conversation. The topics of discussion varied from daily life conversations such as hobbies, personal opinions on specific events, or road direction. The conversation between any pair of speakers was recorded in separate audio channels and transcribed accordingly into separate transcript files.

5.1.3 Libriheavy

The Libriheavy dataset is a significant advancement in the field of ASR, providing an extensive corpus of read English speech. Derived from the LibriLight dataset, it comprises a total of 50,000 hours of audio, making it one of the largest freely-available corpora of speech with supervision. Containing both texts from original books and ASR transcripts, the dataset stands out from other open-sourced datasets due to its inclusion of richer information like punctuation and casing. This makes it more flexible and useful for various tasks including punctuation restoration. In this project, we utilize a small subset of the Libriheavy dataset, namely Libriheavy-small, for model training and testing.

5.2 Data Processing

5.2.1 Dataset Preprocessing

Dataset preprocessing is a critical step in preparing data for machine learning models. This section covers the preprocessing steps for the three datasets before conducting training.

NTU-English & NTU-EnMan

For NTU-English and NTU-EnMan datasets, which are stored in TextGrid files, we first extract the text from each file and combine the resulting text files into a single complete text corpus. We then perform text normalization to remove unnecessary symbols such as brackets, "–EMPTY–", and non-speech noises. For all the mentioned datasets, punctuations are converted or removed to match three classes: period, comma, and space. We convert question marks and exclamation marks to periods to indicate the end of a sentence.

After preprocessing the datasets, we split them into training, validation, and test sets. For NTU-English, we perform a 90:10 split for train and validation sets. Since the test set is predefined, we keep it as it is. For NTU-EnMan, we choose an 80:10:10 split for train, validation, and test sets. It's worth noting that when splitting the data into train, validation, and test sets, we perform the splitting in a conversation-based manner to ensure that the XLM-RoBERTa model learns to predict punctuation within the context of a complete conversation. To provide insights into the characteristics of the preprocessed datasets, we present statistics on the number of words, periods, and commas in both datasets in Table 5.1.

Libriheavy

For the LibriHeavy-small dataset, our preprocessing approach is slightly different from the two datasets above to address the unique structure of this open-source resource, which contains original, punctuation-rich book texts (written form) and their corresponding ASR outputs (spoken form) without punctuation. The initial step involves text normalization of the written form to remove uncommon symbols and characters found in books. Additionally, this written form includes elements not present in the spoken form, such as image captions or footnotes, which are not read out by narrators. To clean these discrepancies, we employ regular expres-

sions to identify and remove patterns associated with such non-vocalized content.

Given our project's emphasis on punctuation restoration in speech transcripts, our primary interest lies in the ASR transcripts. However, the transcripts are in raw form and lack the necessary punctuation for direct use as groundtruth in training and testing. Moreover, the linguistic styles of book texts, such as abbreviations, create discrepancies between the written and spoken forms. To bridge this gap, we come up with a method to extract punctuations from the written form and reinsert them into the spoken form, thus generating a refined groundtruth dataset. The specifics of this punctuation extraction and reinsertion method will be discussed in the subsequent section, since it also applies to outputs from Large Language Models (LLMs).

After establishing the groundtruth text, we partitioned the dataset into training, validation, and testing sets, stored in text files. Since generative LLMs are mostly trained with large datasets, we distribute more utterances into the training set. Table 5.1 presents the statistics for these subsets. Additionally, we prefixed each line across all three subsets with prompts for generative LLMs.

Overall, the dataset preparation process was designed to ensure standardization and suitability for machine learning models, fostering a consistent evaluation framework across datasets. The derived statistics offer insights into the punctuation distribution within the datasets.

5.2.2 LLM Output Postprocessing

LLMs such as Llama-2 and Gemini exhibit generative capabilities, meaning their outputs can diverge from the input, often altering names, grammar, and spelling. While prompt engineering has been leveraged to mitigate these discrepancies, it doesn't entirely eliminate them, as the inherent generativity of LLMs can introduce unpredictability to the outputs. This unpredictability underscores the importance of implementing additional safety measures at the source level. Consequently, the outputs generated by such LLMs often require further post-processing prior to evaluation. A crucial technique in this context is again the extraction and reinsertion method, which targets the punctuation in LLM outputs. This method extracts punctuation marks from the generative outputs and reinserts them into the original input, namely the ASR transcription that lacks punctuation. The subsequent paragraphs will delve into the algorithm's core functionalities, illus-

trating how it addresses the challenges posed by the generative nature of LLMs.

The key functions of the extraction and reinsertion algorithm are shown in Code Snippets 5.1, 5.2 and 5.3. This algorithm operates in two major phases: extraction and reinsertion, with an emphasis on maintaining the integrity of the original text's structure and meaning. Initially, it identifies and extracts words that are immediately followed by one or more punctuation marks (e.g. periods, commas, question marks, exclamation points, semicolons, and colons). This extraction process respects case insensitivity and ensures that matches adhere to word boundaries, thus preventing the erroneous extraction of substrings from within longer words. The words, along with their trailing punctuation and the positions where they were found, are stored for subsequent processing.

```
import re

def extract_words_with_punctuation(text):
    """Extract words followed by punctuation marks from the text,
        case-insensitively."""
    pattern = r'\b(\w+)([\.,\?!;:]+)'
    return [(match.group(1) + match.group(2), match.start()) for
       match in re.finditer(pattern, text, re.IGNORECASE)]
```

Code 5.1: Libriheavy extraction and reinsertion algorithm (Part 1)

The reinsertion phase involves placing each extracted punctuation mark back into the original text. However, rather than simply reverting them to their original locations, the algorithm seeks the best possible match for their reinsertion. It does so by locating the position in the original text where the extracted word (minus its punctuation) appears and is closest to its original position. This step is crucial for maintaining the relative positioning of words and their associated punctuation, especially in cases where the same word occurs multiple times within the text.

```
def find_closest_index(word, original_text, start_index=0):
    """Find the index of the word in the original text that is
       closest to the given start index, case-insensitively."""
    clean_word = re.escape(word.rstrip('.,!?;:'))
    indices = [m.start() for m in re.finditer(r'\b' + clean_word
       + r'\b', original_text, re.IGNORECASE)]
    if not indices:
        return -1
```

```
    closest_index = min(indices, key=lambda x: abs(x -
        start_index))
    return closest_index
```

Code 5.2: Libriheavy extraction and reinsertion algorithm (Part 2)

Special considerations are made for words with punctuation that do not appear in the original input or when dealing with multiple occurrences of the same word. In such scenarios, the algorithm opts to skip the reinsertion of punctuation for non-existent words and employs a method to match each occurrence of a repeated word with its closest counterpart in the original text, based on their indices. This approach ensures that the reinsertion of punctuation accurately reflects the structure and meaning of the original text while accommodating variations in word frequency and distribution.

```
def insert_punctuation(original_text, words_with_punctuation):
    """Insert punctuation marks back into the original text at
        the position of their closest corresponding words."""
    new_text_list = list(original_text)
    offset = 0
    for word, original_position in words_with_punctuation:
        closest_index = find_closest_index(word, original_text,
            original_position)
        if closest_index >= 0:
            insert_position = closest_index + len(word.rstrip('
                .,!?;:')) + offset
            new_text_list.insert(insert_position, word[-1])
            offset += 1

    return ''.join(new_text_list)
```

Code 5.3: Libriheavy extraction and reinsertion algorithm (Part 3)

5.3 Evaluation

5.3.1 Evaluation Metrics

For this project, when evaluating punctuation restoration models, Precision, Recall, Micro-F1, and Macro-F1 Scores are employed as the primary metrics to assess performance. The Micro-F1 Score aggregates the contributions of all classes to compute the average performance, essentially weighting each instance equally regardless of class. This contrasts with the Macro-F1 Score, which is calculated

by determining the F1 score for each class independently and then averaging these scores across all classes. By doing so, the Macro-F1 Score ensures each class is weighted equally in the final score, making it a fair measure even in the face of dataset imbalance. This balanced approach is pivotal for this project, as it aims to uniformly emphasize the significance of every class in the evaluation process, regardless of its frequency within the dataset. The inclusion of both Micro-F1 and Macro-F1 Scores allows for a comprehensive analysis that considers both the overall accuracy (through Micro-F1) and the importance of achieving balanced performance across all classes (through Macro-F1), thereby providing a more comprehensive understanding of the models' capabilities in punctuation restoration.

Word Error Rate (WER) and Sentence Error Rate (SER), as introduced in Chapter 2.3, are also employed in this project in order to evaluate the extent to which generative LLMs modify the input sentences in their output.

5.3.2 Unifying Evaluation Methods across Models

XLM-RoBERTa

The off-the-shelf XLM-RoBERTa model [23] utilizes a token-level evaluation method. After predicting the punctuation for each masked token, the output tokens are directly compared against the groundtruth. The evaluation quantifies the corresponding True Positives (TP), False Positives (FP), and False Negatives (FN) for each punctuation type. To align this evaluation method with that used for generative LLMs, modifications were introduced to the original script. These refinements include the implementation of a detokenization process and the generation of text files containing the punctuated sentences, thus streamlining the evaluation process across different models.

Generative LLMs

For the fine-tuned Llama-2 model, the base Llama-2 model, and the Gemini API, the models are tasked with generating punctuated sentences from input data without punctuation. The approach for evaluating these generative LLMs involves compiling their outputs into text files. This method allows for a straightforward comparison and further assessment of the model's performance in punctuation prediction tasks, thereby facilitating a unified evaluation framework that encompasses all models.

Unified Evaluation Method

To guarantee a fair comparison of the results of different models, it is necessary to use a unified approach for evaluation. Since the predictions from both XLM-RoBERTa and generative LLMs are saved into text files, we designed an algorithm for evaluating the performance of their punctuation restoration capabilities. The algorithm employs dynamic programming to align sequences from the model's predictions with the groundtruth references, thereby ensuring a thorough comparison of the two datasets at the word level. The code is shown as Code Snippet 5.4. The aligned sequences are then analyzed to identify and categorize punctuation marks across prediction and groundtruth, specifically commas, periods, exclamation marks, and question marks.

```
def align_tokens(predicted_, reference_):
    predicted = [token.lower() for token in predicted_]
    reference = [token.lower() for token in reference_]

    predicted = predicted[:len(reference)+1]

    # Initialize a matrix for dynamic programming
    dp_matrix = [[0] * (len(reference) + 1) for _ in range(len(
        predicted) + 1)]

    # Fill in the matrix based on the Needleman-Wunsch algorithm
    for i in range(1, len(predicted) + 1):
        for j in range(1, len(reference) + 1):
            if predicted[i - 1] == reference[j - 1]:
                dp_matrix[i][j] = dp_matrix[i - 1][j - 1] + 1
            else:
                dp_matrix[i][j] = max(dp_matrix[i - 1][j],
                    dp_matrix[i][j - 1])

    # Backtrack to find the aligned sequences
    aligned_predicted = []
    aligned_reference = []
    i, j = len(predicted), len(reference)
    while i > 0 and j > 0:
        if predicted[i - 1] == reference[j - 1]:
            aligned_predicted.insert(0, predicted[i - 1])
            aligned_reference.insert(0, reference[j - 1])
            i -= 1
            j -= 1
        elif dp_matrix[i - 1][j] > dp_matrix[i][j - 1]:
```

```
            aligned_predicted.insert(0, predicted[i - 1])
            aligned_reference.insert(0, '|SPACE|')  # Insert a
                space for unmatched token in reference
            i -= 1
        else:
            aligned_predicted.insert(0, '|SPACE|')  # Insert a
                space for unmatched token in predicted
            aligned_reference.insert(0, reference[j - 1])
            j -= 1

    # Handle any remaining tokens
    while i > 0:
        aligned_predicted.insert(0, predicted[i - 1])
        aligned_reference.insert(0, '|SPACE|')
        i -= 1
    while j > 0:
        aligned_predicted.insert(0, '|SPACE|')
        aligned_reference.insert(0, reference[j - 1])
        j -= 1

    return aligned_predicted, aligned_reference
```

Code 5.4: Aligning LLM prediction with ASR groundtruth

This analysis calculates the True Positives (TP), False Positives (FP), and False Negatives (FN) for each punctuation mark type, enabling the assessment of precision, recall, and F1 score metrics for each category. Additionally, a Macro-F1 score is derived, providing an overview of the model's performance across all punctuation types. The evaluation focuses on four punctuation types: commas, periods, exclamation marks, and question marks. For scenarios where only commas and periods are being evaluated, the TPs, FPs, and FNs for exclamation marks and question marks are aggregated with those for periods to form a combined category as period.

5.4 Summary

Chapter 5 delves into the selection, preparation, and evaluation of speech transcript datasets critical for punctuation restoration using LLMs. It presents an overview of three key datasets, NTU-English, NTU-EnMan, and Libriheavy-small, each chosen for their unique characteristics and relevance to the task at hand. The chapter elaborates on the specialized preprocessing steps undertaken for each dataset, which include text normalization, conversation-based data split-

ting, and the adaptation of punctuation marks to suit model requirements.

A contribution of chapter 5 is the postprocessing technique designed to align LLM outputs with original ASR transcripts. This method is particularly vital for addressing the generative nature of LLMs, which can introduce modifications that deviate from the input text. By extracting and reinserting punctuation from the LLM-generated text back into the original transcripts, this approach enables a more accurate evaluation of the models' punctuation restoration capabilities.

The evaluation framework outlined in this chapter employs precision, recall, micro-F1, and macro-F1 scores to provide a comprehensive analysis of model performance across punctuation types. This unified methodology ensures that the performance of different models can be fairly compared.

As the chapter concludes, it sets the stage for chapter 6, where the experimental setup and analyses are discussed. This next chapter will explore the performance of baseline models, the impact of prompt engineering on Llama-2, and a comparative analysis involving fine-tuned Llama-2, Gemini, and base Llama-2. Building on the datasets and evaluation strategies developed in chapter 5, chapter 6 aims to offer insightful analyses and findings that contribute to the enhancement of LLMs for punctuation restoration for speech transcripts.

Chapter 6

Experiment and Analysis

This chapter delves into a series of experiments and analyses to evaluate and compare the performance of baseline models, the impact of prompt engineering, and the effectiveness of fine-tuning strategies on LLMs for punctuation restoration for speech transcripts. Section 6.1 establishes a performance benchmark using XLM-RoBERTa (section 6.1.1) and base Llama-2-13B model (section 6.1.2) on diverse datasets. Subsequent analyses explore the influence of detailed prompt engineering on the fine-tuned Llama-2 model in section 6.2, examining its ability to mitigate generative LLMs' propensity for altering input text. Section 6.3 then covers a comparative study of the fine-tuned Llama-2 model against Gemini developed by Google, and base Llama-2 using 2 datasets, highlighting advancements in punctuation restoration capability and generalizing ability, especially concerning exclamation marks.

6.1 Experiment 1: Baseline Models Performance

6.1.1 XLM-RoBERTa

We use models trained with XLM-RoBERTa as baseline models. The training of XLM-RoBERTa on NTU-English, NTU-EnMan, and Libriheavy-small datasets spanned 10 epochs, with the performance metrics depicted in Figures 6.1, 6.2 and 6.3. These figures illustrate the updates of the model's F1-score over the epochs, shedding light on the learning stability and generalization capability of XLM-RoBERTa across the two datasets.

Analyzing the performance of XLM-RoBERTa on NTU-English, NTU-EnMan, and Libriheavy-small datasets involves considering the characteristics and com-

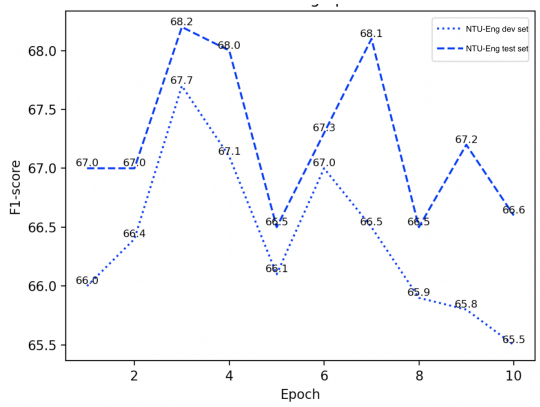

Figure 6.1: Performance of XLM-RoBERTa (Micro-F1) on NTU-English dataset

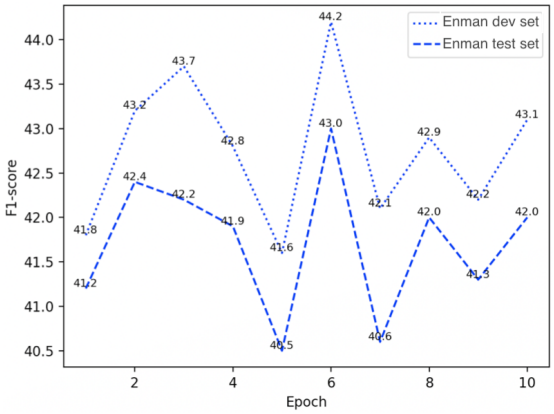

Figure 6.2: Performance of XLM-RoBERTa (Micro-F1) on NTU-EnMan dataset

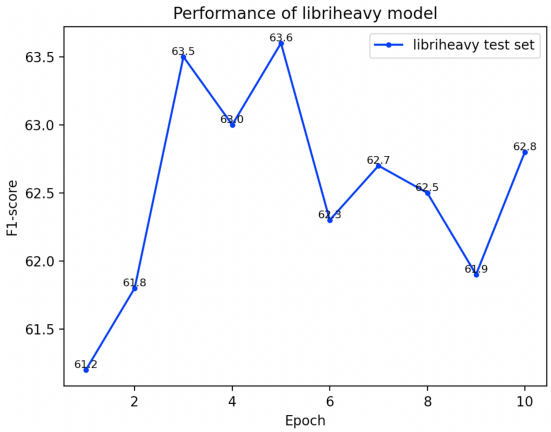

Figure 6.3: Performance of XLM-RoBERTa (Micro-F1) on Libriheavy dataset

plexities of each dataset. The statistics in Table 5.1 suggest a substantial difference in the size and distribution of punctuation in these datasets.

Dataset Size and Complexity:

The Libriheavy-small dataset possesses a significantly larger training corpus compared to both NTU-English and NTU-EnMan datasets. The abundance of data likely allows XLM-RoBERTa to learn from a more diverse set of examples, leading to potentially better generalization when faced with unseen data. Moreover, the distribution of commas and periods across the datasets varies, which could influence the learning efficacy of the model, especially if one punctuation type is more difficult to predict than the other.

The NTU-English dataset, despite being smaller than Libriheavy-small, is still quite substantial in size and offers structured conversations within a medical setting. This particular setting may present its own challenges for the model to learn. On the other hand, the NTU-EnMan dataset is the smallest and includes conversational speech with code-switching, which adds even more complexity due to varied linguistic structures and informal language.

Model Performance:

XLM-RoBERTa shows the highest Micro-F1 scores on the Libriheavy-small dataset. The large amount of training data in this dataset likely helps the model develop more robust representations. For the NTU-English dataset, the performance is also relatively high, suggesting that the model can cope well with the specialized language and structure of medical discussions to some extent. The NTU-EnMan dataset exhibits lower F1 scores, which could be due to its smaller size, and the increased complexity from code-switching in conversational speech, leading to less effective generalization.

The contrasting performances of XLM-RoBERTa on these three datasets might also be partly due to their distinct linguistic characteristics. NTU-EnMan's conversational and bilingual nature poses a challenge for punctuation prediction. The NTU-English dataset, with its clinical dialogues, may resemble more closely the structured scenarios that XLM-RoBERTa was initially trained on, but with added domain-specific language challenges. Finally, the Libriheavy-small dataset, predominantly consisting of read speech from books, provides a more orderly and formal language environment, which likely aids the model's learning process by aligning more closely with structured training data. This discrepancy in text domains across datasets is a factor worth noticing in the observed differences in model performance.

Model Ensemble:

As detailed in 4.1.3, this project investigates a novel approach by leveraging checkpoint ensemble from various epochs within a single training session of the XLM-RoBERTa model, which spans 10 epochs. Throughout this session, checkpoints at each epoch are saved, followed by an evaluation process to identify the top three checkpoints based on their performance metrics. These selected checkpoints are then combined through averaging and aggregation techniques. As shown in Figure 6.1, the empirical findings from this experiment demonstrate an enhancement in the Macro-F1 score, observing an improvement of 1-3% over the highest-scoring individual checkpoint. Adjusting the weights of the checkpoints during aggregation shows a 0.1% fluctuation and does not lead to obvious performance improvement. This outcome underscores the efficacy of model ensembles in bolstering model performance. Nonetheless, despite these gains, the performance still falls short when compared to generative LLMs.

	NTU-English	NTU-EnMan	Libriheavy
F1-Score	0.70	0.45	0.66

Table 6.1: Performance of Model Ensemble (Micro-F1)

6.1.2 Llama-2-13B base model

Without LLM Output Postprocessing

This section evaluates the performance of the Llama-2 13B base model on the Libriheavy-small dataset, serving as a baseline comparison. This analysis is conducted without applying the output postprocessing technique described in Chapter 5.2.2, which involves extracting punctuation marks from the model's output and reinserting them into the input ASR transcript. The results are presented in Table 6.3.

The results indicate that the base configuration of Llama-2 shows a fairly good performance in predicting periods as well as question marks. However, its ability to accurately predict exclamation marks is low. This discrepancy raises a question about the challenges associated with exclamation mark prediction across models and whether fine-tuning could enhance this aspect of model performance.

Additionally, the F1 Score in the "All Punctuation" row denotes the Micro-F1

Score. The observation that the Micro-F1 Score exceeds the Macro-F1 Score can be attributed to the distribution of punctuation marks within the test set. Specifically, periods occur more frequently than exclamation marks, as detailed in Table 6.2.

Dataset	Comma	Period	Exclamation	Question
LibriHeavy Test Set	10.4K	6.6K	502	412

Table 6.2: Distribution of punctuation marks in the LibriHeavy test set

	Precision	Recall	F1 Score
Comma	0.60	0.55	0.57
Period	0.66	0.74	0.70
Exclamation	0.40	0.25	0.31
Question	0.74	0.61	0.67
All Punctuation	0.62	0.60	0.61
Macro-F1 Score: 0.56			

Table 6.3: Llama-2 13B baseline performance without output postprocessing

With LLM Output Postprocessing

The subsequent phase involves implementing an output post-processing step, as detailed in Table 6.4. The F1 Score for the restoration of exclamation marks remains unchanged, however, improvements are observed in other punctuation categories, with the F1 score for periods increasing from 0.70 to 0.79. These enhancements underscore the critical role of the output postprocessing step, particularly in addressing the alignment issues between the base Llama-2 output and the groundtruth that negatively affect performance. Additionally, the result highlights the tendency of the base Llama-2 to alter the input, due to its generative nature.

6.2 Experiment 2: Prompt Engineering for Llama-2

As detailed in Chapter 4.2.2, we designed three distinct prompt variations to explore whether more explicit and detailed instructions enhance the performance of the fine-tuned Llama-2 model. A notable observation was that the fine-tuned

	Precision	Recall	F1 Score
Comma	0.67	0.56	0.61
Period	0.77	0.82	0.79
Exclamation	0.44	0.24	0.31
Question	0.76	0.63	0.69
All Punctuation	0.70	0.62	0.66
Macro F1 Score: 0.60			

Table 6.4: Llama-2 13B baseline performance with output postprocessing

model's generative nature occasionally altered input sentences. As a result, these prompt variations also aim to minimize such alterations. The outcomes are presented in Table 6.5. Specifically, Prompt 1 simply requested, "Restore punctuations for the sentences: <Sentences without punctuation>." Prompt 2 added a directive against modifying sentences, "Restore punctuations for the sentences without changing any words: <Sentences without punctuation>." Prompt 3, the most comprehensive, included detailed instructions and examples, instructing the model to meticulously restore capitalization and punctuation without altering the original text or its meaning, "As a master of English grammar and punctuation, your task is to meticulously restore capitalization and insert missing punctuation marks into the following text without changing any words or adding new content. Ensure the text's original meaning and integrity are preserved, focusing solely on restoring punctuation and capitalization where needed.
<Example 1> <Example Sentences without punctuation> <Groundtruth with punctuation>
<Example 2> <Example Sentences without punctuation> <Groundtruth with punctuation>
Using the guidance provided by the examples above, proceed to restore capitalization and insert the necessary punctuation marks into the text below: <Sentences without punctuation>".

6.2.1 Model Performance Comparison

Analysis of the data in Table 6.5 reveals minimal differences in performance across the three fine-tuned Llama-2 model versions. The simplest and most complex prompts exhibit a mere 1% variation in both Micro and Macro F1 Scores. However, the model tuned with the most detailed prompt achieved a 2% improvement in restoring periods and exclamation marks, suggesting a modest impact of

prompt complexity on punctuation restoration capabilities. However, Prompts 1 and 3 outperformed Prompt 2, which had a significantly higher Word Error Rate (WER) and Sentence Error Rate (SER) as shown in Table 6.6. This discrepancy is highly likely to affect the evaluation alignment, even with output postprocessing, as the algorithm cannot reintroduce punctuation if the word prior to that punctuation does not match between the output and ASR groundtruth. The following experiment refers to fine-tuned Llama-2 model with Prompt 3 as the fine-tuned model.

	Comma	Period	Exclamation	Question	Micro-F1	Macro-F1
Prompt 1	0.67	0.79	0.32	0.74	0.71	0.63
Prompt 2	0.65	0.76	0.30	0.72	0.68	0.61
Prompt 3	0.67	0.81	0.34	0.74	0.72	0.64

Table 6.5: F1 Scores of fintuned Llama-2 models with 3 different prompts

6.2.2 Modification Issues of Generative LLM

This experiment also aims to assess whether prompt engineering could mitigate the generation of irrelevant sentences and input text modifications by the model. This issue has been observed not only with Llama-2 but also with Gemini, where given instructions to restore punctuation, both models would generate superfluous content, including but not limited to "Sure! Here are the sentences with punctuations:", "Note: xxxx", separating one sentence into multiple lines, and outputting duplicates of the same result. This behavior is undesired as it complicates subsequent evaluation steps and challenges the alignment algorithm which is unable to handle such unpredictable generation of irrelevant information. Therefore, it has become necessary for the model to overcome this issue.

Unfortunately, detailed instructions like Prompt 3 cannot fully curtail the generation of irrelevant content. In addition, they do not really address input sentence modification. As shown in Table 6.6, the model fine-tuned with Prompt 3, which explicitly instructs against modifications, paradoxically shows increased sentence modification rates from 0.53 to 0.59 compared to the model associated with Prompt 1. Notably, the model fine-tuned with Prompt 2 exhibited substantial sentence alterations with a sentence error rate of 0.92, which is much higher than 0.53 or 0.59, suggesting potential training issues such as catastrophic for-

getting.

The findings from Table 6.6 conclude that detailed instructions can only partly limit irrelevant content generation, while altering the intrinsic generative behavior of LLMs remains challenging.

	Average WER	SER	Word Error	Sentence Error
Prompt 1	0.05	0.53	7412	2120
Prompt 2	0.16	0.92	24438	3664
Prompt 3	0.05	0.59	8302	2360

Table 6.6: Error rates of fine-tuned Llama-2 models with 3 different prompts

6.3 Experiment 3: Fine-tuned Llama-2 vs. Gemini vs. Base Llama-2

6.3.1 Settings for Comparison

In addition to the modification challenges of generative Large Language Models (LLMs) discussed in the previous section, the Gemini API, developed by Google, introduces further complexities. Unlike open-source models, Gemini is proprietary yet accessible via API. Notably, it occasionally encounters errors with certain input sentences, resulting in a failure to generate outputs. This issue is particularly problematic in batch processing scenarios, where the absence of output for specific utterances disrupts the process. Attempts to manage these errors through "try-except" mechanisms and subsequent retries after a brief pause have been unsuccessful. As a consequence, our approach has been to record the indices of sentences that fail to generate output and revisit them after the initial generation session.

Given these challenges, ensuring a fair comparison and maintaining control over output quality necessitates a specific methodology. For this experiment, sentences are sampled from the test set, and the output files from LLMs are manually checked and aligned with the Automated Speech Recognition (ASR) groundtruth. To test the models' generalization capability outside Libriheavy, we crawl a book named Frankenstein from Project Gutenberg and perform data preprocessing. Table 6.7 presents the statistics for the punctuation distribution within the sampled test set and the test set outside Libriheavy.

Dataset	Comma	Period	Exclamation	Question
Sampled Libriheavy Test Set	2.3K	1.2K	385	140
Test Set outside Libriheavy	1.5K	0.7K	236	87

Table 6.7: Distribution of punctuations in two test sets

6.3.2 Model Performance Comparison

We first perform the testing of Gemini and fine-tuned Llama2 model as well as the base Llama-2 model. We manually check the files and align the sentences if necessary, especially to remove the superfluous content, including but not limited to "Sure! Here are the sentences with punctuations:", "Note: xxxx", aggregate the sentence fragments scattering across multiple lines back into one line, and remove duplicates of the same result. For Gemini, we also need to regenerate the outputs of sentences that failed in the first session and insert them into proper line numbers. After that, we can perform the LLM output postprocessing step, where the punctuation marks are extracted and reinserted back into the unpunctuated ASR transcripts to evaluate against ASR groundtruth.

The results are shown in Table 6.8. FT_Llama-2_postprocess refers to the fine-tuned Llama-2 model with output postprocessing step taken before evaluation. For both base Llama-2 model and Gemini, the output postprocessing step improves the performance. This again demonstrates the necessity of the postprocessing step, as previously stated in Chapter 6.1.

From the next section onwards, we will compare only the models with the output postprocessed.

Base Llama-2 vs. Fine-tuned Llama-2

The comparison between the base and fine-tuned Llama-2 models reveals significant improvements in performance across all four types of punctuation. Specifically, the greatest enhancement is observed in the exclamation mark class, where the F1 score surged from 0.20 to 0.36. This substantial increase underscores the

efficacy of fine-tuning with LoRA, enabling the model to better capture language styles related to the Libriheavy dataset, particularly in the use of exclamation marks. Such findings underscore the value of employing LoRA for fine-tuning Large Language Models (LLMs), as it markedly enhances the capability of these models to accurately restore punctuation in speech transcripts.

Gemini vs. Fine-tuned Llama-2

Gemini, a state-of-the-art model developed by Google, boasts up to 50 trillion parameters in its Gemini-pro variant, which we have utilized for testing and evaluation. Specifically designed for text-only prompts, Gemini-pro demonstrates superior performance across all four punctuation types when compared to the base Llama-2 model. When contrasted with the fine-tuned Llama-2 model, which employs LoRA and comprises 13 billion parameters, the fine-tuned variant exhibits a remarkable capacity for restoring punctuation in speech transcripts, despite its significantly lower parameter count. The enhancements are evident, with the fine-tuned Llama-2 model outperforming Gemini-pro by 4% in comma restoration and 5% in exclamation marks, achieving parity with Gemini-pro in period accuracy. This underscores the fine-tuned model's enhanced proficiency in discerning language cues particular to the Libriheavy speech transcript. However, in the domain of question mark prediction, Gemini-pro surpasses the fine-tuned Llama-2 by 4%, achieving an F1 score of 0.76 against Llama-2's 0.72, indicative of Gemini's advantage in leveraging extensive training datasets.

	Comma	Period	Exclamation	Question	Micro-F1	Macro-F1
Base_Llama-2_postprocess	0.58	0.72	0.20	0.67	0.60	0.54
Gemini_postprocess	0.63	0.74	0.31	**0.76**	0.65	0.61
FT_Llama-2_postprocess	**0.67**	**0.75**	**0.36**	0.72	**0.67**	**0.63**

Table 6.8: F1 scores of Llama-2 compared with Gemini for Libriheavy test subset

Models' Generalizing Ability outside Libriheavy

As indicated in Table 6.9, our fine-tuned Llama-2 model demonstrates a performance comparable to the state-of-the-art Gemini model across all punctuation classes when evaluated on the text of the Frankenstein book. This underscores the robust generalization capabilities and exceptional performance of the fine-tuned Llama-2 model beyond the Libriheavy dataset. Notably, both the fine-tuned Llama-2 and Gemini models achieve an F1-score of 0.61 for exclamation marks.

When compared to their performance on the Libriheavy test subset, there is an approximate 30% improvement in the accuracy of exclamation mark restoration for both models. This improvement likely stems from more distinct and consistent cues present in the text that signal the use of exclamation marks.

	Comma	Period	Exclamation	Question	Micro-F1	Macro-F1
Base_Llama-2_postprocess	0.70	0.76	0.43	0.77	0.70	0.67
Gemini_postprocess	**0.69**	**0.79**	0.61	0.84	**0.72**	0.73
FT_Llama-2_postprocess	0.67	0.78	**0.61**	**0.84**	0.71	**0.73**

Table 6.9: F1 scores of Llama-2 compared with Gemini for test set outside Libriheavy

Overall Comparison

Upon examining the performance differences among all the models, it becomes evident that fine-tuning significantly enhances model performance. The fine-tuned Llama-2 not only surpasses its base version across all punctuation categories but also approaches and surpasses Gemini's punctuation prediction capability. This improvement, especially in comma and exclamation mark prediction, highlights the effectiveness of fine-tuning in refining the model's understanding and application of punctuation, showcasing its improved linguistic comprehension and predictive precision. The performance metrics, as reflected in Table 6.8 and 6.9, underline the tangible benefits of fine-tuning, suggesting that even with a substantially smaller parameter size, the fine-tuned Llama-2 model achieves competitive, if not superior, performance in key areas compared to both its base version and the more parameter-heavy Gemini model.

6.3.3 Analysis of Exclamation Mark Performance

In our experimental analysis, we observed a relatively low performance in exclamation mark prediction across all models. This discrepancy in performance, compared to other punctuation classes, prompts an in-depth investigation into potential underlying causes.

As shown in Figure 6.4, one significant challenge lies in the inherent ambiguity of exclamation mark usage. Unlike periods or question marks, which often follow more predictable patterns, exclamation marks can be subject to a wider

range of interpretations. Many sentences do not exhibit clear linguistic or syntactic indicators for the appropriate application of exclamation marks, leading to a frequent misclassification, typically as periods. This issue is compounded by the variable thresholds different authors maintain for employing exclamation marks, further complicating the models' ability to accurately predict their placement.

Moreover, the contextual dependence of exclamation marks poses an additional layer of complexity. In certain instances, the determination of whether a sentence warrants an exclamation mark requires an analysis of preceding or succeeding text, beyond the immediate sentence. This contextual requirement highlights a limitation in the models' capacity to infer the necessary emphasis or emotional tone that justifies an exclamation mark, especially in sentences where exclamations are used to highlight specific words or phrases for emphasis.

In addition, the inconsistent use of exclamation marks to convey emphasis or emotional intensity across different writing styles further exacerbates the challenge. Our models, despite their sophistication, struggle to capture the nuanced cues that signal the use of exclamation marks, reflecting a broader limitation in understanding the expressive dimensions of language.

This analysis underscores the need for enhanced model training strategies that can better accommodate the subtle cues and contexts governing exclamation mark usage. Addressing this gap not only requires refining the models' linguistic sensitivity but also entails developing more advanced techniques for contextual analysis, enabling a better interpretation of text that aligns with human writing styles.

6.4 Summary

Chapter 6 presents a systematic exploration of punctuation restoration capabilities across models including XLM-RoBERTa, Llama-2-13B, and Gemini. The baseline performance analysis of XLM-RoBERTa across different datasets highlights the model's learning stability and underscores the impact of dataset size and complexity on performance. The introduction of the model checkpoint ensemble strategy demonstrates a potential pathway to performance improvement, albeit with limitations when compared to generative LLMs.

groundtruth: instead of the usual formula, respect this! the emperor added, spare my people! he then hanged himself.

prediction: instead of the usual formula, respect this, the emperor added, spare my people. he then hanged himself.

groundtruth: release your mind. free your soul. be vacuous. be nothing!

prediction: release your mind. free your soul. be vacuous. be nothing.

groundtruth: lo, the immortal, his hand grasping a lotus flower, away to time everlasting, trackless through the regions of space!

prediction: lo! the immortal, his hand grasping a lotus flower, away to time everlasting, trackless through the regions of space.

groundtruth: but on coming to offer him the post, they found out that he had died of fright!

prediction: but on coming to offer him the post they found that he had died of fright.

groundtruth: nay, replied the master, it is indeed bad to forget one's wife; but tis worse to forget one's self!

prediction: nay, replied the master, it is indeed bad to forget one's wife, but tis worse to forget one's self.

Figure 6.4: Examples of wrongly predicted exclamation marks

Prompt engineering experiments with the fine-tuned Llama-2 model underscore the challenges inherent in generative LLMs, particularly their tendency to generate irrelevant content or alter input text. Despite the detailed prompt designs, fully mitigating these issues remains difficult, suggesting areas for future research and model refinement.

The comparative analysis between fine-tuned Llama-2 and Gemini, alongside base version, showcases the effectiveness of fine-tuning techniques like LoRA in enhancing model performance. Notably, the fine-tuned Llama-2 model demonstrates competitive, if not superior, performance in certain aspects compared to the more parameter-heavy Gemini model. By testing with a dataset outside Libriheavy, the fine-tuned Llama-2 model also demonstrates competitive performance, indicating its good generalizing ability. This comparison not only highlights the fine-tuned model's capabilities but also points to the potential for optimization and efficiency in model design and application.

Moreover, the focused analysis on exclamation mark performance uncovers the challenges associated with predicting less frequently used punctuation marks. The findings suggest the need for advanced training strategies and improved contextual analysis techniques to better capture the expressive dimensions of language.

As the chapter concludes, it paves the way for the next steps in LLM research, particularly in punctuation restoration for speech transcripts. Future work will likely explore more sophisticated fine-tuning methods and innovative training

datasets to further enhance model accuracy in real-world scenarios.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis embarks on a comprehensive exploration of the potential of LLMs in enhancing punctuation restoration for speech transcripts, with a particular focus on transformer models such as XLM-RoBERTa. Despite the novel integration of a model checkpoint ensemble strategy, which yields a marginal improvement in performance, the XLM-RoBERTa model is outperformed by both the Llama-2 and Gemini models.

An achievement of this research is the fine-tuning of the Llama-2 model with the Low-Rank Adaptation (LoRA) method, which showcases remarkable superiority in punctuation restoration tasks for speech transcripts. This approach not only sets new benchmarks when compared to the state-of-the-art Gemini-pro model but also underscores the potential of fine-tuning LLMs in this domain. Although the investigation into prompt engineering reveals little success in addressing the input alteration problem inherent in generative LLMs, it provides modest benefits in enhancing the punctuation restoration capabilities of Llama-2.

The generative nature of LLMs motivates us to develop an LLM output post-processing algorithm, to neutralize the adverse effects on model performance. Moreover, the effort to standardize evaluation methods across different models is essential for ensuring fair and unbiased comparisons. Throughout our experimental analysis, the challenge of accurately restoring exclamation marks is identified, prompting a deeper inquiry into the underlying causes.

The insights gained from this study not only contribute to the enhancement of

model performance but also deepen our understanding of the strengths and limitations of LLMs in punctuation restoration. This work paves the way for future research, particularly in refining model architectures, exploring more advanced fine-tuning techniques, and expanding the datasets for a more comprehensive evaluation of LLM applications in punctuation restoration.

7.2 Future Work

The findings of this thesis open several avenues for future research in punctuation restoration using LLMs. Firstly, expanding the evaluation of LLMs to include additional datasets and languages could provide deeper insights into the models' adaptability across diverse linguistic contexts. Secondly, exploring alternative fine-tuning techniques and model architectures could uncover more efficient and effective methods for enhancing punctuation restoration performance. Furthermore, the integration of contextual analysis and the development of training strategies for less frequently used punctuation marks, such as exclamation marks, could address current performance gaps.

References

[1] Britney Muller. BERT 101: State Of The Art NLP Model Explained. https://huggingface.co/blog/bert-101, 2022.

[2] Zhiqiang Hu, Yihuai Lan, Lei Wang, Wanyu Xu, Ee-Peng Lim, Roy Ka-Wei Lee, Lidong Bing, and Soujanya Poria. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*, 2023.

[3] Changsong Liu, Thi Nga Ho, and Eng Siong Chng. An empirical study on punctuation restoration for english, mandarin, and code-switching speech. In *Asian Conference on Intelligent Information and Database Systems*, pages 286–296. Springer, 2023.

[4] Liu Changsong, Ho Thi Nga, Yip Jia Qi, and Chng Eng Siong. Transformer-based punctuation restoration models for indonesian with english codeswitching speech transcripts. In *2023 10th International Conference on Advanced Informatics: Concept, Theory and Application (ICAICTA)*, pages 1–6. IEEE, 2023.

[5] Fernando Batista, Diamantino Caseiro, Nuno Mamede, and Isabel Trancoso. Recovering punctuation marks for automatic speech recognition. In *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[6] Ottokar Tilk and Tanel Alumäe. Lstm for punctuation restoration in speech transcripts. In *Sixteenth annual conference of the international speech communication association*, 2015.

[7] Ottokar Tilk and Tanel Alumäe. Bidirectional recurrent neural network with attention mechanism for punctuation restoration. In *Interspeech*, pages 3047–3051, 2016.

[8] Madina Hasan, Rama Doddipatla, and Thomas Hain. Multi-pass sentence-end detection of lecture speech. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[9] Wei Lu and Hwee Tou Ng. Better punctuation prediction with dynamic conditional random fields. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 177–186, 2010.

[10] Karan Makhija, Thi-Nga Ho, and Eng-Siong Chng. Transfer learning for punctuation prediction. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 268–273. IEEE, 2019.

[11] György Szaszák and Máté Akos Tündik. Leveraging a character, word and prosody triplet for an asr error robust and agglutination friendly punctuation approach. In *INTERSPEECH*, pages 2988–2992, 2019.

[12] Adebayo Mustapha Bakare, Kalaiarasi Sonai Muthu Anbananthen, Saravanan Muthaiyah, Jayakumar Krishnan, and Subarmaniam Kannan. Punctuation restoration with transformer model on social media data. *Applied Sciences*, 13(3):1685, 2023.

[13] Viet Dac Lai, Abel Salinas, Hao Tan, Trung Bui, Quan Tran, Seunghyun Yoon, Hanieh Deilamsalehy, Franck Dernoncourt, and Thien Huu Nguyen. Boosting punctuation restoration with data generation and reinforcement learning. *arXiv preprint arXiv:2307.12949*, 2023.

[14] Xue-Yong Fu, Cheng Chen, Md Tahmid Rahman Laskar, Shashi Bhushan TN, and Simon Corston-Oliver. Improving punctuation restoration for speech transcripts via external data. *arXiv preprint arXiv:2110.00560*, 2021.

[15] Tanvirul Alam, Akib Khan, and Firoj Alam. Punctuation restoration using transformer models for high-and low-resource languages. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 132–142, 2020.

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

[17] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

[18] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[19] Shaoshi Ling, Yuxuan Hu, Shuangbei Qian, Guoli Ye, Yao Qian, Yifan Gong, Ed Lin, and Michael Zeng. Adapting large language model with speech for fully formatted end-to-end speech recognition. *arXiv preprint arXiv:2307.08234*, 2023.

[20] Patrick K O'Neill, Vitaly Lavrukhin, Somshubra Majumdar, Vahid Noroozi, Yuekai Zhang, Oleksii Kuchaiev, Jagadeesh Balam, Yuliya Dovzhenko, Keenan Freyberg, Michael D Shulman, et al. Spgispeech: 5,000 hours of transcribed financial audio for fully formatted end-to-end speech recognition. *arXiv preprint arXiv:2104.02014*, 2021.

[21] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[22] Wei Kang, Xiaoyu Yang, Zengwei Yao, Fangjun Kuang, Yifan Yang, Liyong Guo, Long Lin, and Daniel Povey. Libriheavy: a 50,000 hours asr corpus with punctuation casing and context. *arXiv preprint arXiv:2309.08105*, 2023.

[23] Abhinav Rao, Ho Thi-Nga, and Chng Eng Siong. Punctuation restoration for singaporean spoken languages: English, malay, and mandarin. In *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 546–552. IEEE, 2022.

[24] Máté Ákos Tündik and György Szaszák. Joint word-and character-level embedding cnn-rnn models for punctuation restoration. In *2018 9th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, pages 000135–000140. IEEE, 2018.

[25] Vasile Păiş and Dan Tufiş. Capitalization and punctuation restoration: a survey. *Artificial Intelligence Review*, 55(3):1681–1722, 2022.

[26] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

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

[28] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[29] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[30] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.

[31] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[32] Kai Lv, Yuqing Yang, Tengxiao Liu, Qinghui Gao, Qipeng Guo, and Xipeng Qiu. Full parameter fine-tuning for large language models with limited resources. *arXiv preprint arXiv:2306.09782*, 2023.