

Hybrid Bilingual Question-Answering System with Human-Level Naturalness Text-to-Speech Extension

Pham Minh Tuan^{*}, Nguyen Quoc Hung^{*}, Le Phuoc Gia Thinh^{*}
Supervisors: Nguyen Quoc Trung, and Truong Hoang Vinh
FPT University, Ho Chi Minh Campus, Vietnam
{tuanpmse160561, hungnqse162007, thinhlpkse161384}@fpt.edu.vn
{trungnq46, vinhth8}@fe.edu.vn^{*}

April 18, 2024

^{*}These authors contributed equally to this work.

Acknowledgements

We would like to express our sincere gratitude to the following people for their invaluable support:

- **Our families**, for their unwavering love, encouragement, and belief in our abilities.
- **Our supervisors, M.S. Nguyen Quoc Trung and Ph.D. Truong Hoang Vinh**, for their expert guidance, mentorship, and dedication to our success.
- **Mr. Nguyen Trong Tai and Mr. Phan Thanh Hai**, for their insightful feedback and valuable contributions.
- **FPT University's dedicated educators**, for providing us with the knowledge and foundation to excel.
- **Our friends at FPT University**, for their invaluable advice, unwavering encouragement, and steadfast support, which greatly enriched our project journey.
- **The entire Capleaf team**, for their collaboration, trust, and tireless commitment to realizing our shared vision.

We are deeply grateful for the support of each and every one of you.

Author Contributions

- **Conceptualization:** Pham Minh Tuan, Nguyen Quoc Hung, Le Phuoc Gia Think.
- **Methodology & Software:** Pham Minh Tuan (LLM Module), Nguyen Quoc Hung (RAG module), Le Phuoc Gia Think (TTS module).
- **Data Curation:** Pham Minh Tuan (LLM dataset), Nguyen Quoc Hung (LLM dataset), Le Phuoc Gia Think (TTS dataset).
- **Writing - Original Draft Preparation:** Pham Minh Tuan, Nguyen Quoc Hung, Le Phuoc Gia Think.
- **Writing - Review and Editing:** Pham Minh Tuan, Nguyen Quoc Hung, Le Phuoc Gia Think,
- **Supervision:** Mr. Nguyen Quoc Trung, Dr. Truong Hoang Vinh.
- **Project Administration:** Pham Minh Tuan (Lead), Nguyen Quoc Hung, Le Phuoc Gia Think.
- **Tools & Resources:** Pham Minh Tuan, Nguyen Quoc Hung, Le Phuoc Gia Think.
- **Funding Acquisition:** Pham Minh Tuan, Nguyen Quoc Hung, Le Phuoc Gia Think.

All authors have read and agreed to the Final Capstone Project document.

Abstract

This project introduces a robust bilingual question-answering and natural Text-to-Speech (TTS) system, primarily designed to support both Vietnamese and English languages. Leveraging pre-trained Large Language Models (LLMs), the system enables seamless interactions in both text and speech domains. Key components include a Retrieval Augmented Generation Pipeline (RAG) for efficient information retrieval, an LLM module for response generation and an integration of a TTS module for human-like speech synthesis, and the development of a user-friendly Demo Web Application. The proposed solutions encompass various stages, including data collection, RAG framework development, fine-tuning of LLMs, and rigorous TTS evaluation using metrics and user surveys. Notable achievements of the project include the development of "T-LLama" - a 7-billion-parameter Bilingual QA LLM, ranked among the top 5 on the VMLU Leaderboard, the implementation of a Bilingual RAG system, the creation of "viXTTS" - the first LLM-based Vietnamese TTS model with voice cloning and multilingual capability through transfer learning, and the assembly of "viVoice" - a gigantic Vietnamese speech dataset with over 1000 hours of audio.

Keywords: Vietnamese - English Bilingual, Large Language Model (LLM), Text Dataset, Speech Dataset, Large-Scale Data Collection, Text-to-Speech (TTS), Retrieval-Augmented Generation (RAG), Natural Language Processing (NLP)

Contents

Acknowledgements	2
Author Contributions	3
List of Figures	7
List of Tables	9
List of Abbreviations	10
1 Introduction	11
1.1 Large Language Model Module.....	11
1.2 Retrieval Augmented Generation Module	12
1.3 Text-to-speech Module	12
2 Related Works	14
2.1 State of Vietnamese large language model	14
2.1.1 Vietnamese large language model	14
2.1.2 Vietnamese corpus	14
2.2 Retrieval-Augmented Generation.....	14
2.2.1 Naive RAG.....	14
2.2.2 Advanced RAG	15
2.2.3 Modular RAG	15
2.3 State of Vietnamese Text-to-Speech	16
2.4 Vietnamese Speech Datasets.....	17
3 Project Management Plan	18
4 Materials and Methods	20
4.1 Large Language Model.....	20
4.1.1 Instruction tuning	20
4.1.2 Data selection.....	21
4.1.3 Data parsing/translating pipeline.....	22
4.1.4 Model selection	24
4.1.5 Model training	25
4.2 RAG Framework.....	29
4.2.1 Document Parsing.....	29
4.2.2 Indexing and Vector Database	30
4.2.3 Retriever Module	31
4.2.4 Advance Rerank Method.....	33
4.2.5 Complete pipeline	34
4.3 TTS Dataset Collection and Description	35
4.3.1 Stage 1: Data Acquisition and Preprocessing	36
4.3.2 Stage 2: Get Transcription and Timestamps.....	37
4.3.3 Stage 3: Post-Filtering and Quality Assurance	37
4.4 viVoice Statistic.....	38
4.4.1 Dataset Overview	38
4.4.2 Utterance Duration Characteristics	39
4.5 Experiments.....	41
4.5.1 TTS Model Architecture.....	41
4.5.2 TTS Training	42
4.6 TTS Evaluation.....	42
4.6.1 Subjective Evaluation Metrics.....	42
4.6.2 Objective Evaluation Metrics.....	43

5	Results	44
5.1	LLM Results	44
5.1.1	Vietnamese Multitask Language Understanding Benchmark	44
5.1.2	Open LLM leaderboard	45
5.2	RAG Results	46
5.2.1	RGB Benchmark	46
5.2.2	Experiment	47
5.3	TTS Results	49
5.3.1	Subjective Evaluation Results	49
5.3.2	Objective Evaluation Results	50
5.4	Full pipeline results of LLM + RAG + TTS	50
5.4.1	Hardware Requirements	50
5.4.2	API Specification	51
5.4.3	Full Demo App	52
6	Discussion	57
6.1	Large language model merging	57
6.2	LLM are in context learners	60
6.3	viVoice Limitations and Future Work	61
7	Conclusion: A Synergistic Approach to Bilingual Interaction	63
A	Detail of Vietnamese Multitask Language Understanding Benchmark for T-Llama	71
B	Large language model dataset format	73
C	RAG prompt examples	74
D	In-context learning examples	77
E	XTTSv2 model architecture details	78
F	Detailed Comparison of viVoice and Existing Datasets	79
G	Data Inspection and MOS Collection Tools	82
G.1	Streamlit Data Inspector	82
G.2	Gradio MOS Collector	82
G.3	Error Transcription Labeling	83
H	Detail MOS collection process	83
H.1	Noise introduction to reference speeches with audiomentations	83
H.2	MOS Guidance	84
I	Dataset and Model Access	85

List of Figures

1	Three paradigms of RAG [1].....	16
2	Instructions, inputs, and outputs for three tasks in Flan 2021. [2].....	21
3	Translation split logic	23
4	Comparison between full fine-tuning, Lora and Qlora	26
5	Visualization of how we create the base model.....	28
6	Last 10 runs.....	29
7	Splitting Process.....	30
8	Indexing And Searching Query Process.....	31
9	Hybrid Searching Process	32
10	Rerank Visualization	34
11	Our RAG Pipieline Visualization	35
12	Data processing pipeline	36
13	Duration distribution per channel	39
14	Violin plots illustrating the distribution of utterance durations across various speech corpora, including viVoice, LibriTTS, BUD500, VIVOS, VinBigData, and VietnamCeleb. To enhance visualization, the y-axis is capped at 50.	40
15	Box plots depicting the distribution of speaking rate (words per minute) across viVoice and other speech corpora, including VIVOS, VinBigData, BUD500, and LibriTTS	41
16	Abstract architecture of the XTTSv2 model. The newly added part is highlighted in green, indicating that the tokenizer is modified for adaptation to the Vietnamese language.	42
17	An example of VMLU test set.....	44
18	An example of general chit chat with T-Llama	53
19	An example question answering on the Qlora paper asking <code>Bạn có thể giải thích cho tôi Low rank adapters là gì được không ?</code>	54
20	An example question answering on the Qlora paper asking <code>So sánh cho tôi hiệu suất giữa QLora với full fine-tuning, method nào có hiệu suất tốt hơn ?</code> Internal prompt for LLM+RAG is shown here 28	55
21	An example question answering on the Qlora paper asking <code>Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit Float</code> Internal prompt for LLM+RAG is shown here 27	56
22	Here is an example chat with the sft version of llama2 from Meta	58
23	Here is an example chat with the model using the our procedure 5 but with meta-llama/Llama-2-7b-chat-hf instead of yeen214/llama2_7b_merge_orcafamily and then tuned on our mixed En-Vi dataset. The chat model from Meta have a very chatty response so we use this as an example for our hypothesis of why this method work.	59
24	Intent classifier LLM function, a more detail prompt is shown in this figure here 29 and here 30	61
25	One example of the parsed and translated dataset	73
26	One example of the formatted prompt.....	73

27	Here is an example of T-Llama answering the query <code>Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit float</code> based on the provided documents that the retriever extracted and returned from the QLORA [3] paper.	75
28	Here is an example of T-Llama answering the query <code>So sánh cho tôi hiệu suất giữa Qlora với full finetuning, method nào có hiệu suất tốt hơn ?</code> based on the provided documents that the retriever extracted and returned from the QLORA [3] paper.	76
29	Here is an example of T-Llama correctly classify the prompt <code>Hãy viết cho tôi một bài văn miêu tả cô giáo</code> as an open ended question creative prompt and determined that the model can response on its own and can bypass the RAG module.	77
30	Here is an example of T-Llama correctly classify the prompt <code>Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit float</code> as a technical question and determined that the model need outside information to response and activate the RAG module.	78
31	Violin Plot of viVoice parts	79
32	Comparison of the mel-spectrograms of viVoice samples with others. Notably, viVoice samples exhibit consistent cleanliness, akin to the samples from the LibriTTS-R datasets.	80
33	Screenshot of the Streamlit Data Inspector web application.	82
34	Screenshot of the Gradio MOS Collector web application.	83

List of Tables

1	Summary of Public Available Vietnamese Speech Datasets	17
2	Project Plan	18
3	Source Code and Data	19
4	Summary of subsets of viVoice dataset. This table summarizes key metrics including the number of speakers, total samples, duration in hours, male proportion and error text proportion for various parts of the viVoice dataset, as well as the combined statistics for the entire dataset.....	39
5	Leaderboard of fine-tuned models on the VMLU benchmark. Our model T-Llama ranks in the top 5 as of the time of writing. Detail results can be found here: Appendix A	45
6	TOP-10 of the open LLM leaderboard by huggingface on 15/4/2024 only for llama2-7B backbone	46
7	Noise Robustness testbed. Higher noise ratio means more negative documents existing in context provided for Generator.....	47
8	Information Integration testbed. Rej means the rejection rate (%) and Rej* means the rejection rate evaluated by ChatGPT. Higher Rej mean this model having ability to reject the noise information.....	48
9	Information Integration testbed. Higher noise ratio means more negative documents existing in context provided for Generator.	48
10	Subjective evaluation results for different training configurations. nMOS and sMOS scores are reported with 95% confidence intervals.....	49
11	Combined results showing both objective evaluation (Word Error Rate and Character Error Rate) and speaker verification (Equal Error Rate and Speaker Encoder Cosine Similarity) for different training configurations.	50
12	Summary of T-Llama scores by Category	71
13	Data sources and text	81

List of Abbreviations

ASR Automatic Speech Recognition. [13](#)

DNN Deep Neural Network. [12](#)

TTS Text-To-Speech. [11](#)

Generation

In the context of natural language processing, generation refers to the process of creating new text or data based on a given input or prompt. It is commonly used in tasks such as text generation, machine translation, and image captioning.. [12](#)

Generator

In the context of language models, a generator, such as a Large Language Model (LLM), is a model trained to produce new text based on a given input or prompt. It typically generates text by predicting the next word or token in a sequence, often using techniques such as autoregressive generation or sampling from a probability distribution.. [12](#)

LLM Large Language Model. [11](#)

Multi-speaker

Refers to speech synthesis datasets containing recordings from multiple people. These datasets are valuable for tasks requiring synthesized speech with diverse voices and styles, such as speaker adaptation and voice conversion.. [13](#)

NLP Natural Language Processing. [11](#)

Pre-trained

Referring to a model that has been trained on a large dataset before being fine-tuned or used for a specific task. Pre-trained models often exhibit better performance and generalization compared to models trained from scratch.. [11](#)

QA Question Answering. [11](#)

RAG Retrieval-Augmented Generation. [12](#)

Sampling rate

The number of data points per second in the sound wave. It is typically measured in hertz (Hz) and determines the frequency range that can be accurately represented in the digital signal.. [13](#)

Single-speaker

Refers to speech synthesis datasets containing recordings from only one person. These datasets are useful for tasks where the synthesized speech should maintain consistency in voice characteristics and speaking style across different utterances.. [13](#)

Supervised Fine-Tuning

Supervised fine-tuning (or SFT for short) is a crucial step in RLHF. In TRL we provide an easy-to-use API to create your SFT models and train them with few lines of code on your dataset.. [11](#)

Zero-shot

A method in machine learning where a model learns to perform tasks it has never seen before during training. Instead of being trained on specific examples of each task, the model relies on its understanding of related tasks to make predictions on new tasks.. [13](#)

1 Introduction

In recent years, the field of **Natural Language Processing (NLP)** has witnessed remarkable advancements, paving the way for innovative applications such as **Question Answering (QA)** systems and **Text-To-Speech (TTS)** synthesis. Our project seeks to harness these technological advancements to develop a robust and versatile system capable of bilingual (Vietnamese and English) question-answering and human-level naturalness text-to-speech applications.

The significance of our project lies in its potential to enhance the way users interact with information, particularly in bilingual contexts. By leveraging **Pre-trained Large Language Model (LLM)**, we aim to create a system that not only provides accurate and contextually relevant responses in both English and Vietnamese but also replicates the natural nuances of human speech, enhancing the overall user experience.

Key objectives of our project include developing a bilingual support system for question-answering, designing a RAG to efficiently fetch relevant information, and integrating a text-to-speech module that enhances accessibility and engagement.

The proposed solutions outlined in this introduction encompass data generation and collection, data processing, RAG framework architecture development, fine-tuning techniques, evaluation metrics, and the development of a demo web application. By addressing these key aspects, we aim to provide a comprehensive and impactful solution to the challenges posed by bilingual question-answering and text-to-speech synthesis.

1.1 Large Language Model Module

Large Language Models (**LLMs**) have emerged as trans-formative tools in **NLP**, revolutionizing applications like question answering, text generation, and language understanding. These powerful neural network architectures, trained on vast amounts of text data, possess an exceptional ability to capture intricate patterns and nuances of human language. This enables them to perform a wide range of language-related tasks with remarkable fluency and coherence.

LLMs like ChatGPT ¹, Gemini ², and Claude ³ have demonstrated remarkable performance in understanding and generating human-like text, thanks to their transformer-based architectures and pre-training on massive datasets. These models can comprehend and reason over complex linguistic structures, allowing them to provide nuanced and coherent responses to natural language queries. However, developing a LLM for Vietnamese presents unique challenges, including:

- **Lack of supervised fine-tuning datasets:** There is a scarcity of datasets specifically designed for instruction tuning in Vietnamese.
- **Limited pre-trained models:** The availability of large pre-trained base models for Vietnamese is limited, mainly because the lack of large Vietnamese corpus.

These challenges necessitate building a complete translation pipeline from scratch and creating a Vietnamese instruction dataset. This project will explore strategies to overcome these hurdles and leverage the power of LLM for bilingual question-answering applications in Vietnamese and English.

So in this paper, we introduce a novel approach to developing a bilingual Large Language Model for both English and Vietnamese. Our key contributions are:

- A general parsed/translation pipeline utilized multi-thread processing that can be used with any dataset for easy parsing and translating different dataset format.
- Creation of a large-scale Vietnamese language **Supervised Fine-Tuning** dataset by converting large English instruction datasets to Vietnamese by using our translation pipeline. This dataset serves as the foundation for fine-tuning our Vietnamese LLM.

¹<https://chat.openai.com>

²<https://gemini.google.com/app>

³<https://claude.ai/chats>

- Fine-tuning of a state-of-the-art transformer-based architecture, on ours mixed English-Vietnamese SFT dataset. This allows us to create a high-performing Vietnamese LLM capable of tasks like question answering, text generation, and language understanding.
- Evaluation of the Vietnamese LLM's performance on a range of Vietnamese Natural language processing benchmarks, demonstrating its effectiveness in real-world applications.
- Development of a bilingual question-answering system that leverages the Vietnamese LLM and an existing English LLM to provide seamless cross-lingual support.

By addressing the key challenges in developing LLM for Vietnamese, this work aims to make significant contributions to the advancement of Vietnamese NLP and enable more accessible and efficient language-based applications for Vietnamese users.

However, LLMs are not infallible, and their knowledge can be limited by the data they were trained on, potentially leading to biases, hallucinations, or outdated information. This is where the retrieval component of the RAG module comes into play, complementing the LLM's generative capabilities with the ability to dynamically retrieve and incorporate up-to-date information from external sources, such as databases, knowledge bases, or the internet. This is where our next section come into play [1.2](#).

1.2 Retrieval Augmented Generation Module

The synergy between the LLM's language understanding and generation abilities, and the retrieval component's accessing to external knowledge, forms the core backbone of the [Retrieval-Augmented Generation \(RAG\)](#) framework. This RAG term was first introduced by [\[4\]](#) in 2020, which has been existed long time ago but not having a systematic consolidation and abstraction in this field until appearing the Gao's survey [\[1\]](#). We split RAG frameworks into three aspects:

- Retriever module is responsible for searching supportive chunks (from database, uploaded knowledge) that relating to user query by computing embedding or matched keyword. After that, set of top **K** chunks, which relating to user's query, are returned by computing and sorting similarity scores.
- Generation module combine both user's question and provided documents to create coherent and relevant responses. The model's responses can vary based on task-specific criteria, model can draw upon its pretrained knowledge or from retrieved documents to restrict which information is generated. In ongoing dialogues, the model can integrate with existing conversational history, enabling effective multi-turn interactions.
- Augmentation module integrate some playground techniques on both retrieval module and [Generation](#) module to optimize as much as possibly the ability and performance. Some methods focus on fine-tuning [Generator](#) such as LLM on specific domain, or extending types of resources for not only text but also with image, table,... On top of that, some proposed ones focus on refining retriever module during the retrieval process, including iterative retrieval, recursive retrieval, and adaptive retrieval to enhance the depth, relevance, and adaptability of the obtained information, which addressing challenges as the "lost in the middle" phenomenon in RAG framework.

In this work, we mainly research how to leverage the most ability of LLM Module in real case. That's why experiments in [Section 5.3](#) will play around with LLM module and have some naive results on Retriever Module, such as tuning hyper-parameter like chunk size or chunk overlap to enhance the ability of retrieving correct knowledge. Our contribution can generalized as proposing open-source RAG framework for supporting some naive techniques such as retrieving based on matching keyword, or retrieve using query embedding through client's databases,... To dive deeper about how this framework activate, consider to [section 4.2](#).

1.3 Text-to-speech Module

The field of text-to-speech has witnessed remarkable advancements, going from concatenative and statistical parametric approaches to [Deep Neural Network \(DNN\)](#) techniques [\[5\]](#). Recent TTS works [\[6-8\]](#) have shown a growing interest

in [Zero-shot](#) techniques for voice adaptation with high speaker similarity and naturalness, allowing the rapid adoption of a new voice with minimal training data. However, despite their efficiency, these methods typically require at least hundreds of hours of good pre-training data [9, 10] to effectively generalize between different voices.

The application of zero-shot techniques for voice adaptation has been particularly successful in high-resource languages such as English, where ample pre-training data are readily available. However, the situation is markedly different for low-resource languages such as Vietnamese, where the availability of such extensive speech data is limited [11–13]. Available Vietnamese speech resources have traditionally catered to [Automatic Speech Recognition \(ASR\)](#) applications [14–18], comprising recordings with inappropriate length distribution and subpar audio quality characterized by noise and low [Sampling rate](#). There are also limited public datasets on [Single-speaker](#) TTS synthesis such as [19, 20], which lack speaker diversity. The scarcity of [Multi-speaker](#) resources in the Vietnamese language landscape has impeded research progress, evidenced by the absence of comprehensive studies or investigations into multi-speaker speech synthesis specifically tailored to the Vietnamese language.

In response to this challenge, we introduce viVoice, a pioneering dataset tailored specifically for multi-speaker TTS synthesis in Vietnamese. viVoice offers a comprehensive repository of high-quality speech samples. Drawing upon the knowledge gleaned from well-known datasets such as LibriTTS [9] and LibriTTS-R [21], viVoice was developed through a multi-step pipeline, integrating advanced speech restoration techniques to ensure optimal quality and fidelity. viVoice incorporates several essential features aimed at enhancing multi-speaker TTS synthesis:

- **Adequate Sampling Rate:** Audio files are captured at 24kHz.
- **Sentence-Based Segmentation:** Speech is segmented at sentence breaks, enabling coherent and contextually relevant synthesis.
- **Preserved Punctuations:** The original text is maintained in its unnormalized form, capturing the linguistic richness and stylistic nuances.
- **Balanced Distribution:** Balancing the duration per sample distribution.
- **Speech Restoration:** Techniques are applied to remove background music and noise, improving speech clarity and quality.

To evaluate the effectiveness of the viVoice dataset, we used it to train viXTTS, a multi-speaker model built on the pre-trained weight of XTTSv2 [8]. This achievement marks a significant milestone as the first publicly accessible Vietnamese multi-speaker model. This model serves as a pivotal benchmark for evaluating the dataset’s capacity to support multi-speaker Text-to-Speech (TTS) synthesis. By leveraging viVoice in this pioneering endeavor, we aim to not only demonstrate its suitability for training advanced TTS models but also pave the way for future research and development in the domain of Vietnamese speech synthesis.

The contributions of this module can be outlined as follows:

- We provide an overview of existing Vietnamese speech datasets, shedding light on their limitations in effectively supporting multi-speaker TTS tasks by analyzing the text diversity, audio quality, speaker diversity. Through this examination, we underscore the pressing need for a dataset tailored specifically for multi-speaker synthesis in Vietnamese.
- We thoroughly describe the characteristics of the viVoice dataset, giving details about its size, variety of speakers, and methods of data creation. Specifically, viVoice is a comprehensive and carefully selected collection of high quality speech samples, addressing the shortcomings found in existing datasets.
- Additionally, we provide an overview of the data pipeline used to create viVoice, offering insight into the process behind the dataset’s development and curation. This comprehensive overview aims to enhance transparency and reproducibility while facilitating future research efforts in the field of multi-speaker TTS synthesis.
- Furthermore, we delve into the potential applications and implications of viVoice in accelerating research and development in the realm of Vietnamese TTS synthesis. By training a multi-speaker model using viVoice, we not only demonstrate its efficacy but also establish a landmark achievement as the first publicly available Vietnamese multi-speaker model.

2 Related Works

This review encompasses various aspects of inquiry, including bilingual question-answering systems, text-to-speech synthesis methodologies, speech recognition and translation technologies, advancements in Large Language Models, and innovative approaches such as Retrieval-Augmented Generation (RAG). By synthesizing insights from these diverse areas of research, this section aims to contextualize the current study within the broader landscape of natural language processing and speech synthesis.

2.1 State of Vietnamese large language model

2.1.1 Vietnamese large language model

The development of Vietnamese large language models has seen significant progress in recent years, with various research groups and organizations contributing to this growing field.

One of the prominent models is GemSura [22], a pretrained LLM based on the Gemma [23] architecture from Google developed by the URA team at the Ho Chi Minh City University of Technology (HCMUT).

PhoGPT [24], a state-of-the-art 7.5B-parameter generative model series for Vietnamese, includes the base pretrained monolingual model PhoGPT-7B5 and its instruction-following variant PhoGPT-7B5-Instruct. Both of these however is trained from scratch only on the Vietnamese language and thus cannot leverage the vast amount of data on the internet, which is primarily English.

Sailor [25] is a suite of open LLMs tailored for Southeast Asian languages, including Vietnamese, Indonesian, Thai, Malay, and Lao. Similarly, SeaLLM [26] is a multilingual LLM focused on Southeast Asian languages.

Other notable Vietnamese LLMs include ToRoLaMa [27], an instruction-following and chat model, Vistral-7B-Chat [28], a conversationally-tuned model with a strong English base model from Mistral AI [29]. Additionally, VBD-LLaMA2-Chat is a conversationally-tuned LLaMA2 model for Vietnamese from VinBigdata fine-tuned from meta-llama/Llama-2-7b [30], and Vietnamese LLaMA 2 [31] is a 7B version of LLaMA 2 with 120GB of Vietnamese text.

2.1.2 Vietnamese corpus

In terms of datasets, the Vietnamese research community has access to several large-scale corpora, including the VN News Corpus, 10000 Vietnamese Books, and the VietnameseWAC dataset. Additionally, multilingual datasets like CulturaX [32] and Bactrain-X [33] include Vietnamese data, while open-source repositories like Common Crawl and WikiDumps provide broader multilingual resources.

These advancements in Vietnamese LLMs and datasets demonstrate the growing interest and momentum in this field, paving the way for further research and development to enhance the capabilities of language models for the Vietnamese language.

2.2 Retrieval-Augmented Generation

2.2.1 Naive RAG

The Naive RAG research approach marks one of the initial methodologies to become notable following the broad adoption of ChatGPT. [34] referred this method as a "Retrieve-Read" method, which involves indexing, retrieval, and content generation. The process begins by parsing the documents into chunks, then transforming into embeddings and stored in a vector database. Next, when a user asks a question, this query is used to compare with the prepared embedding to return the top K retrieved chunks. Finally, the language model generates a response by adapting its response strategy based on the specific task, which leverages its in-built knowledge or limits its responses from the selected chunks.

Despite its advantages and simplification, the Naive RAG approach faces several challenges. The retrieval stage can suffer from accuracy issues, leading to the selection of irrelevant sections or the omission of important content. During generation, there's a risk of the model creating content that doesn't align with the context ("hallucination") or producing responses marred by irrelevance, toxicity, or bias. Moreover, blending retrieved information with varying

tasks can result in disjointed outputs, and managing redundant information from different sources can cause repetitive answers. That's why the Advanced and Module RAG come into play, which can address most of these previous problems.

2.2.2 Advanced RAG

The Advanced RAG model introduces significant enhancements to address the shortcomings found in the Naive RAG model, with a particular emphasis on improving the retrieval phase through both pre-retrieval and post-retrieval tactics. With pre-retrieval phase, enhancing data granularity play a vital role, by removing irrelevant information, dispelling ambiguity in terms, managing outdated information, quality of generated response is enhanced. On top of that, tracking the size of chunks after parsing documents can also ensure the only needed information is contained, helping model create the consistent and informative response better. Besides, adding meta-data or relationship between chunks before indexing can also improve retrieval efficiency. Except from that, [35] proposed method to address addresses alignment issues between documents by using "hypothetical questions"

At the core of retrieval phase, the focus is on the similarity between query and knowledge to have appropriate content. Fine-tuning Embedding models is proved that significantly improving performance in specific domain, which contains lots of professional terms. For general domains, BGE embedding models from BAAI ⁴ or embeddings-ada-02 from OpenAI ⁵ are widely used to generate meaning embedding. On top of that, MTEB Leaderboard ⁶ is now emerging a new method, which leveraging the LLM's ability of understanding language to generate embedding and has demonstrated formidably results on the top rank. Even multilingual field also has been achieved from BAAI's BGE-M3 model for generating embedding represented in multi language.

The post-retrieval phase is also pivotal for integrating the retrieved context with the query effectively. Key practices include re-ranking the retrieved segments to prioritize the most relevant content and compressing the context to streamline the information before feeding it into LLM. Re-ranking helps in positioning the most crucial information more prominently within the prompt, a strategy seen in systems like LlamaIndex2 ⁷, LangChain ⁸, and HayStack ⁹. Given the risk of overwhelming LLMs with too much information, post-retrieval actions focus on distilling the content to its essence, highlighting the most critical parts, and reducing the overall context size for processing. This approach helps in maintaining a sharp focus on the most pertinent details, avoiding the dilution of key information with irrelevant data.

2.2.3 Modular RAG

The updated RAG framework, known as Modular RAG, represents a significant evolution from its predecessors, improving flexibility and adaptability. This approach introduces a variety of methods to enhance its functions, including a new search module for advanced similarity searches and the refinement of the retrieval system via fine-tuning [36]. Key innovations include the redesign of RAG modules [37] and the iteration of the RAG processing pipelines [38] to address specific issues. The trend towards adopting a modular RAG configuration supports both the sequential processing of tasks and the holistic, end-to-end training across its various elements. Modular RAG, while unique, continues to build on the essential concepts of its predecessors, Advanced and Naive RAG, demonstrating ongoing development and sophistication within the RAG ecosystem.

There also exist some new, specialized components to improve retrieval and processing efficiency. For specific, [39] facilitates tailored searches using code across multiple data platforms such as search engines, databases, and knowledge graphs. [40] implement a multi-query approach that broadens user queries, employing parallel vector searches and smart re-ranking to reveal both obvious and insightful knowledge. Selfmem [41] utilizes LLM's memory for better retrieval, establishing a vast memory pool that aligns more closely with data patterns through continuous self-improvement. Except from that, [42] capability efficiently navigates through various data sources to find the best route for a query. [37] aims to minimize redundancy and irrelevance by creating context directly via the LLM, enhancing the

⁴FlagEmbedding: <https://github.com/FlagOpen/FlagEmbedding>

⁵OpenAI 2022: <https://openai.com/>

⁶MTEB: <https://huggingface.co/spaces/mteb/leaderboard>

⁷LlamaIndex: <https://www.llamaindex.ai/>

⁸LangChain: <https://www.langchain.com/>

⁹HayStack: <https://haystack.deepset.ai/overview/intro>

precision and relevancy of results. [43] and [44] customize RAG for different downstream tasks, enabling automated prompt retrieval for zero-shot inputs and developing task-specific retrievers for few-shot query generation.

Moving beyond the module replacement or reconfiguration of earlier mention, there also exist some flexible combination by the adjustment of interactions among existing modules, broadening its applicability. For example, the Rewrite-Retrieve-Read model [34] utilizes the LLM to refine retrieval queries with a rewriting module and feedback mechanism, enhancing task efficacy. Other strategies, such as Generate-Read [37] and Recite-Read [45], introduce innovative retrieval methods focusing on LLM-generated content and model weights, respectively, to better manage knowledge-intensive tasks. Hybrid retrieval techniques combine various search methods to accommodate different quality retrieved results, including keyword-based search, semantic search, and vector search. The reconfiguration of modules and interactions, as seen in frameworks like DSP [46] and the iterative ITER-RETGEN flow [38], highlights the dynamic utilization of module outputs to enhance another module's performance, demonstrating a complex understanding of improving module interaction. Complex techniques like FLARE [47] and Self-RAG [48], show the advantages of a flexible architecture, which can easily incorporate other technologies such as fine-tuning or reinforcement learning for improved retrieval outcomes, more personalized outputs, or cooperative fine-tuning efforts.

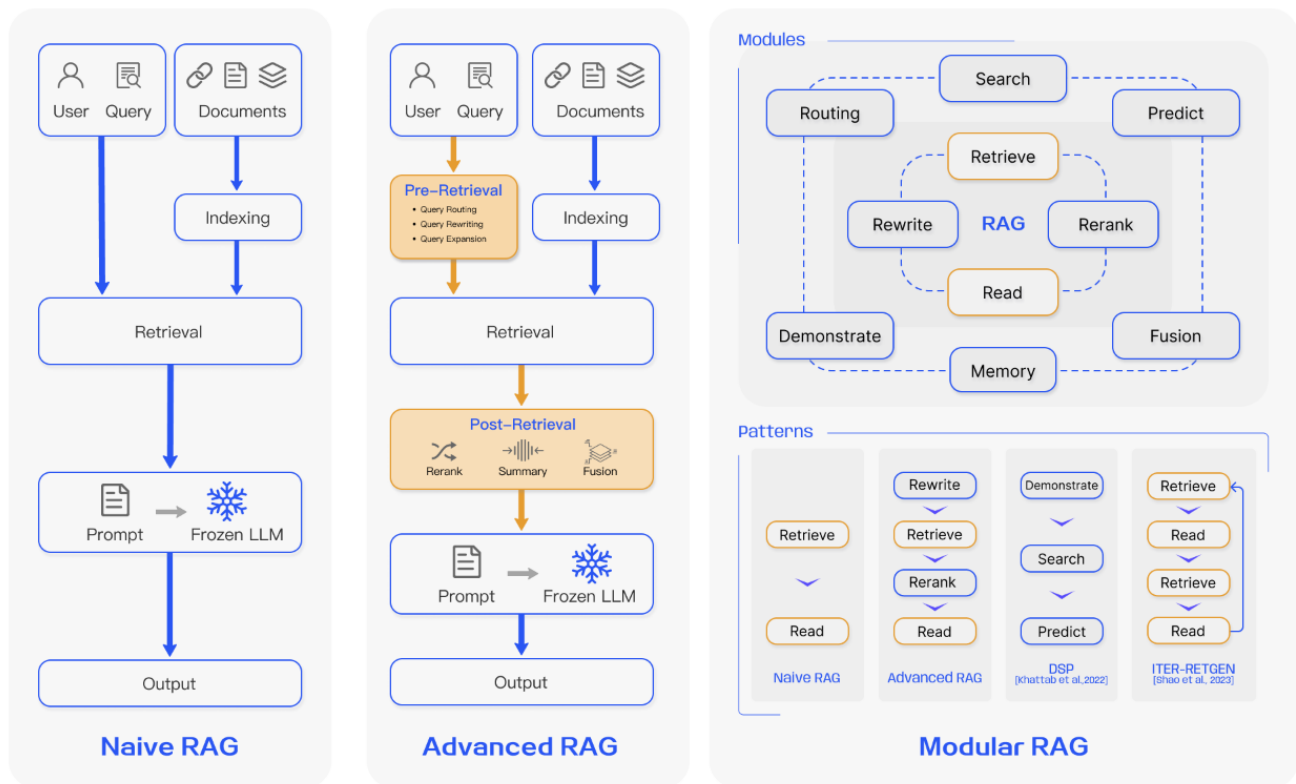


Figure 1: Three paradigms of RAG [1]

2.3 State of Vietnamese Text-to-Speech

The recent advancements in Vietnamese text-to-speech (TTS) technology are evident across various applications, including voice assistant platforms [49], reminder and notification systems [50], assistive technologies for the visually impaired [51], call reminder services [52], educational software [53], and smart speaker solutions [54]. These examples illustrate the widespread practical usage and increasing adoption of TTS systems in diverse domains. Vietnamese TTS systems have pursued innovative approaches to enhance speech quality and naturalness, striving to closely mimic human voices in the reading domain [55].

In recent years, the research focus in Vietnamese TTS has evolved significantly, largely influenced by the VLSP challenges held from 2019 to 2023. Initially centered on single-speaker speech synthesis [56–58], recent advancements have propelled the field towards more nuanced approaches, notably, single-speaker speech synthesis with emotion [59,60]. Leading synthesis models such as Tacotron2 [61], Fastspeech2 [62], and VITS [63] have been widely adapted for Vietnamese in single speaker settings. Furthermore, the emergence of diffusion-based models, such as Adapt-TTS [64] and Sten-TTS [65], indicates a new research direction towards Vietnamese multi-speaker and zero-shot speech synthesis.

2.4 Vietnamese Speech Datasets

Table 2.4 presents a non-exhaustive compilation of public Vietnamese speech corpora, showcasing the evolution of available resources in the field. The majority of these datasets, including VIVOS [14], FOSD [15], VinBigdata-VLSP2020 [16], and BUD500 [18], were primarily curated for automatic speech recognition (ASR) tasks. ASR datasets often contain noise and utilize text normalization techniques that remove punctuations, as noted in the analysis (refer to Appendix F). Furthermore, the challenges posed by the differences in sentence lengths between ASR and TTS datasets in training a LLM-based TTS models to achieve natural intonation and prosody were highlighted in section 5.3, where we compared the output quality of the models trained separately with BUD500 and viVoice datasets.

For TTS applications, existing open datasets like VietTTS [20] and Vietbiblevox [19] predominantly feature recordings from a single speaker, limiting their utility in capturing the diverse nuances of natural speech. The Vietnamese part of Common Voice 16 dataset [66], while featuring 276 speakers, has a relatively small total duration, and a majority of the data is not validated. In contrast, multi-speakers datasets such as VNSpeechCorpus [67] and the datasets used to train Adap-TTS [64] and Sten-TTS [65], though acknowledged to exist, are unfortunately not publicly available. Although VietnamCeleb [17] boasts a collection of over 1000 speakers, it was primarily designed for the speaker verification task, lacking transcription and containing predominantly noisy speech. Despite the absence of public access to these datasets, their existence underscores the potential for exploring multi-speaker synthesis in the Vietnamese language domain, urging further efforts in dataset sharing and collaboration within the research community.

Table 1: Summary of Public Available Vietnamese Speech Datasets

Dataset	Year	Task	Speakers	Samples	Hrs	SR
VIVOS [14]	2016	ASR	65	12420	16	16000
Common Voice [66]	2019	ASR	276	16175	18	48000
FOSD [15]	2018	ASR	N/A	25921	30	48000
VinBigdata100h [16]	2020	ASR	N/A	56427	101	16000
VietTTS [20]	2022	TTS	1	22884	36	22050
VietBibleVox [19]	2023	TTS	1	29185	67	48000
Vietam-Celeb [68]	2023	SR/SV	1000	87140	187	16000
BUD500 [18]	2024	ASR	N/A	649158	462	16000
viVoice	2024	TTS	≥ 186	887772	1017	24000

3 Project Management Plan

This section provides the overall project plan and the source codes. For a more detailed project plan, please check this sheet: https://docs.google.com/spreadsheets/d/1cViQUGwL0isb-TgxVNtUP-ZRAefkkBm_xlYHeRv7o9s/edit?usp=sharing

Table 2: Project Plan

Task Name	Owner	Week
Find documents, papers, methods	All Members	1
Feasibility tests on different methods	All Members	2
Start making data and training models	All Members	3
Improve datasets, models, methods	All Members	4
Improve datasets, models, methods	All Members	5
Improve datasets, models, methods	All Members	6
Improve datasets and models; Start writing documents	All Members	7
Improve datasets and models; Update documents	All Members	8
Start evaluations for each module of the pipeline; Update documents	All Members	9
Making demos; Update documents	All Members	10
Making demos; Update documents	All Members	11
Finish demos and evaluations; Opensource models	All Members	12
Finish documents and final pipeline demo	All Members	13
Revision, Preparation for Thesis defense	All Members	14

Table 3: Source Code and Data

Items	Link	Description
T-Llama checkpoint	https://huggingface.co/capleaf/T-Llama	Checkpoint for the T-Llama model and list of datasets used.
Vietnamese QA System	https://github.com/vTuanpham/Vietnamese_QA_System	Training script for LLM module.
Large Dataset Translator	https://github.com/vTuanpham/Large_dataset_translator	A multilingual translator for large datasets.
RAG	https://github.com/hungnq1310/rag	A repository for RAG (Retrieval-Augmented Generation) module.
TTS Data	https://github.com/thinhlpg/viVoice	viVoice dataset.
viXTTS model demo	https://github.com/thinhlpg/vixtts-demo	Demo for the TTS module.
viXTTS checkpoint	https://huggingface.co/capleaf/viXTTS	Checkpoint for the viXTTS model.
Full Demo App	Included in Edunext Submission	Full demo app code
Demo Video	https://youtu.be/NBuIRlPQwLw	Full pipeline Demo.
Other Source code	Included in Edunext Submission	viXTTS Training, Gradio MOS App,...

4 Materials and Methods

4.1 Large Language Model

4.1.1 Instruction tuning

Instruction Tuning is a technique used to fine-tune large language models (LLMs) to perform specific tasks or follow instructions more effectively. The key idea behind Instruction Tuning is to leverage the general language understanding and generation capabilities of LLMs and adapt them to follow explicit instructions or solve particular problems.

The process of Instruction Tuning typically involves the following steps:

1. **Data Collection:** Researchers or developers gather a dataset of task-specific instructions and corresponding desired outputs or behaviors. This dataset is used to fine-tune the LLM.
2. **Model Pretraining:** The LLM is first pretrained on a large corpus of general text data, such as web pages, books, or other natural language sources. This pretraining helps the model develop a strong understanding of language structure, semantics, and common patterns.
3. **Instruction Tuning:** The pretrained LLM is then fine-tuned on the task-specific dataset of instructions and desired outputs. This fine-tuning process adjusts the model's parameters to better understand and follow the given instructions, enabling it to generate more relevant and coherent responses.
4. **Evaluation and Refinement:** The performance of the Instruction-Tuned LLM is evaluated on a held-out set of instructions or tasks. Based on the results, the model may be further fine-tuned or adjusted to improve its ability to follow instructions and complete the target tasks.

The key benefits of Instruction Tuning include:

1. **Task-Specific Adaptation:** By fine-tuning the LLM on specific instructions or tasks, the model can better understand the context and requirements of those tasks, leading to more accurate and relevant outputs.
2. **Improved Coherence and Alignment:** Instruction Tuning helps the LLM generate responses that are more closely aligned with the given instructions, improving the overall coherence and reliability of the model's outputs.
3. **Scalability and Generalization:** Instruction Tuning allows for the development of versatile LLMs that can be adapted to a wide range of tasks and instructions, rather than being limited to a specific set of pre-defined capabilities.
4. **Reduced Bias and Errors:** The fine-tuning process can help mitigate biases and errors that may be present in the original LLM, as the model is trained to follow specific instructions more closely.

Instruction Tuning has been applied to a variety of tasks, such as question answering, task completion, code generation, and even creative writing as shown in Figure 2. As LLMs continue to advance, Instruction Tuning is expected to play an increasingly important role in making these models more versatile, reliable, and aligned with user needs and preferences.

Finetune on many tasks (“instruction-tuning”)

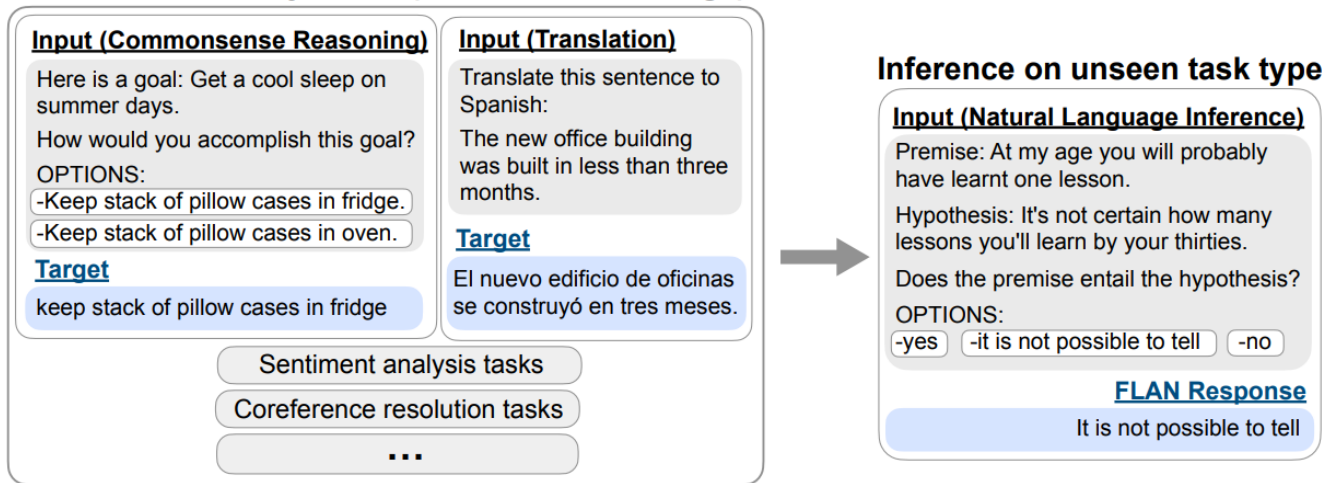


Figure 2: Instructions, inputs, and outputs for three tasks in Flan 2021. [2]

Instruction Tuning provides a foundation for our approach to dataset selection in Section 4.1.2. Rather than developing a simple question-answering system, we aim to create a local assistant that can effectively follow user instructions and carry out specific tasks. By fine-tuning a large language model through Instruction Tuning, we can imbue the model with the ability to understand and respond to explicit instructions from the user, rather than just answering general queries. This shift from a generic question-answering system to a task-oriented local assistant requires a carefully curated dataset that reflects the types of instructions and behaviors we want the model to learn. The data selection process in the next section will be crucial in shaping the capabilities of our Instruction-Tuned LLM, enabling it to serve as a versatile and reliable personal assistant for our users.

We further discuss the importance of instruction tuning in 6.2 where we can turn our model into any functions given enough examples (In-context learning).

4.1.2 Data selection

To develop a robust and versatile language model capable of understanding and generating content in both English and Vietnamese, we have curated a diverse dataset that draws from various sources. The primary goal of this data collection is to ensure our model emerges with strong bilingual capabilities, capable of comprehending and communicating effectively in both languages.

The core of our dataset consists of synthetic data generated from larger language models, such as GPT-4, GPT-3.5¹⁰, and Claude 2¹¹. These high-performance models have been used as "teacher" models to transfer their reasoning capabilities and broad knowledge into our dataset. By leveraging the strengths of these models, we aim to imbue our own model with the ability to tackle a wide range of tasks and queries, from open-ended questions to logical reasoning and mathematical problem-solving.

To supplement the synthetic data, we have also incorporated human-labeled datasets from sources such as Reddit’s ELI5 (Explain Like I’m 5) and databricks-dolly-15k [69]. These real-world examples of language usage provide valuable insights into how people communicate and seek information, allowing our model to learn from authentic patterns of expression.

Recognizing the importance of native Vietnamese language support, we have included several specialized datasets in our collection:

1. mt_eng_vietnamese [70]: This dataset contains 136k examples of translation between English and Vietnamese,

¹⁰<https://chat.openai.com>

¹¹<https://claude.ai/chats>

providing a foundation for our model to understand and translate between the two languages, this help the model to learn internally how to understand in both languages.

2. Bactrian-X (vi split) [33]: This 67k examples dataset combines questions from the AlpacaCleaned [71] and databricks-dolly-15k [69] datasets, with responses generated by the ChatGPT 3.5 model.
3. vilmlima-vi¹²: This small but valuable dataset consists of 1,030 Vietnamese question-answer pairs, sourced from the Virtual Interactive platform, an Ontocord community¹³.

To further enhance the model’s bilingual capabilities, we have also incorporated native English datasets, databricks-dolly-15k [69] (an instruction-following annotated completely by human), and OpenOrca [72] (a collection of user queries and ChatGPT/GPT-4¹⁴ responses).

Additionally, we have included datasets that focus on specific domains, such as TIGER-Lab/MathInstruct [73], which contains over 262k examples of high-quality, linguistically diverse grade school math word problems. This dataset is designed to improve the model’s reasoning and problem-solving skills, which can then be applied to a broader range of tasks and queries.

By curating this diverse and comprehensive dataset, we aim to create a language model that can seamlessly navigate between English and Vietnamese, while also demonstrating strong logical reasoning and task-completion abilities. This approach will allow our model to serve as a versatile and reliable assistant, capable of supporting users in a wide range of applications and scenarios.

4.1.3 Data parsing/translating pipeline

The data parsing and translation pipeline, as illustrated in Figure 3, plays a critical role in preparing the dataset for the LLM training process. This pipeline ensures that the data is converted into a format compatible with the model’s requirements, while also efficiently translating the content between English and Vietnamese.

The key steps in this pipeline include:

1. Data Parsing: The raw dataset is parsed and converted into a standardized format, such as that supported by the pyarrow and Hugging Face datasets libraries.
2. Pre-Translation Filtering: Filters are applied to the dataset to remove examples that may contain coding tasks(Translation will translate coding variables and function definition) or other content that could interfere with the translation process.
3. Parallel Translation: The dataset is split into chunks and translated in parallel 3, leveraging multi-thread processing to significantly reduce the overall translation time.
4. Automatic Retry Mechanism: Any threads that fail during the translation process are automatically restarted, ensuring that all data points are fully translated.
5. Data Format Conversion: The translated dataset is converted into a format compatible with the LLM training process, facilitating seamless integration.

By implementing this robust and efficient data translation pipeline, the LLM development process can leverage a high-quality, bilingual dataset that supports the model’s ability to understand and generate content in both English and Vietnamese.

¹²vilml/lima-vi: <https://huggingface.co/datasets/vilm/lima-vi>

¹³vilm: <https://huggingface.co/vilm>

¹⁴<https://chat.openai.com>

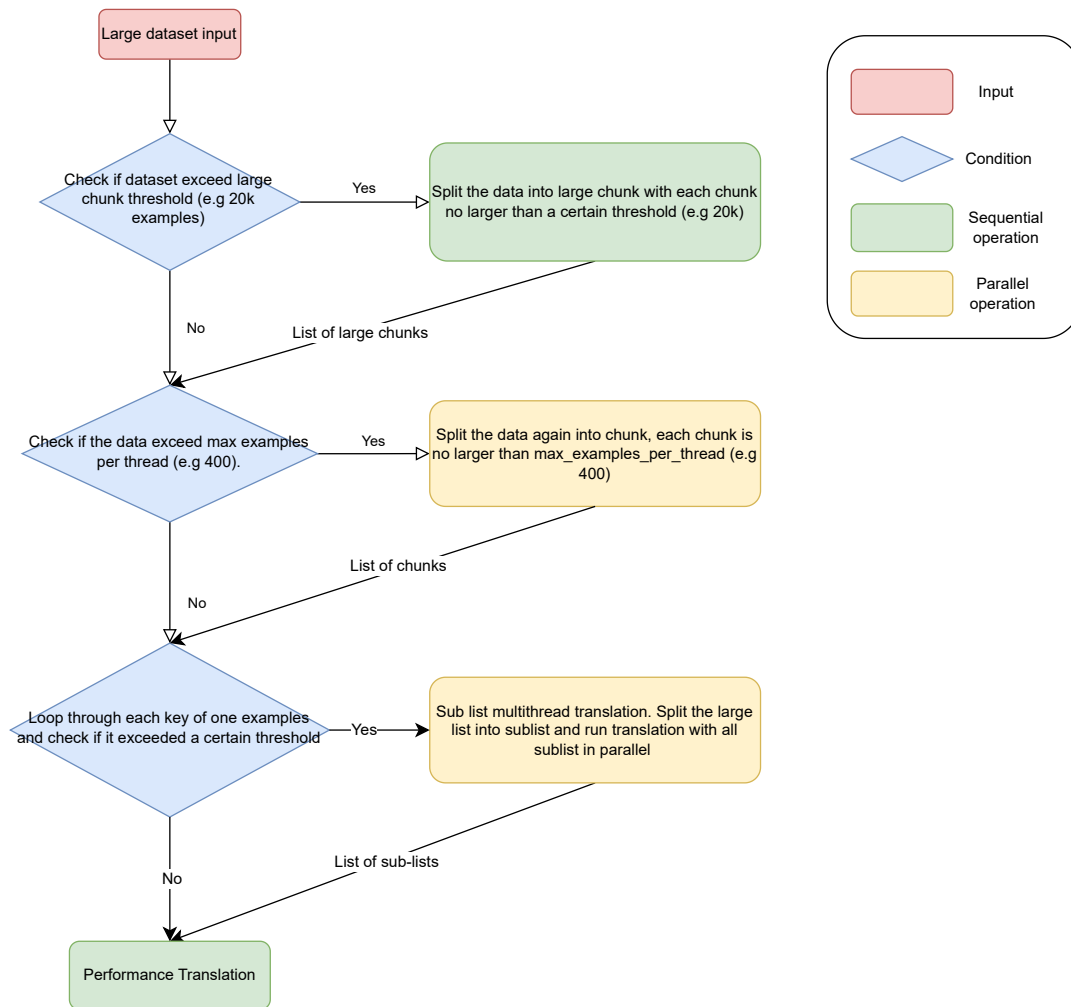


Figure 3: Translation split logic

The data contain the following fields:

- `qas_id`: Unique id for each example
- `system_prompt`: A context setter prompt at the top of the input. We adapt this field from the OpenOrca [72] dataset mainly because this is the largest instruction dataset that we picked from 4.1.2. In the OpenOrca [72] paper, the authors state that "The system message, placed at the start of the prompt, provides the LFM with essential context, guidelines, and other pertinent details." We follow them and design many sets of `system_prompts` from general assistant task to a more specific task like math, translation,...
- `question_text`: The user input query
- `orig_answer_texts`: The answer for the query (Agent response)
- `answer_lengths`: The answer length

Each example is a dictionary contain all fields mention above 25

```

class OpenOrcaParser(DataParser):
    def __init__(self, file_path: str, output_path: str):
        super().__init__(file_path, output_path,
                          parser_name=PARSER_NAME,
                          target_config=BaseConfig,
                          target_fields=['question_text', 'orig_answer_texts'],
                          do_translate=True,
                          no_translated_code=True)

    def read(self) -> None:
        super(OpenOrcaParser, self).read()
        self.data_read = load_dataset("Open-Orca/OpenOrca")

        return None

    def convert(self) -> None:
        super(OpenOrcaParser, self).convert()
        data_converted = []
        for split in self.data_read:
            for data in tqdm(self.data_read[split], desc=f"Converting_{split}_data"):
                data_dict = {}
                data_dict['system_prompt'] = data['system_prompt']
                data_dict['qas_id'] = data['id']
                data_dict['question_text'] = data['question']

                data_dict['orig_answer_texts'] = data['response']
                data_dict['answer_lengths'] = None
                data_converted.append(data_dict)

        self.converted_data = data_converted

        return None

```

Example for creating a new parser for OpenOrca dataset

The translation pipeline has been implemented to make it easier to parse and translate new datasets as illustrated in Figure 4.1.3. The user only needed to implement the read and convert function; the rest, like saving the correct format, translating, error handling, etc., is handled by the main DataParser class.

This makes it easier to adapt to new dataset and incorporate it into our model training data.

We adopt the prompt template from Alpaca [71] family of model and modify it to our own as shown here 26.

4.1.4 Model selection

Based on the insights provided in the previous sections, we have decided to use the bkai-foundation-models/vietnamese-llama2-7b-120GB [31] model as the starting point for our Instruction-Tuned language model.

The key reasons for selecting this model are:

1. **Architecture Alignment:** The llama2-7b [30] architecture has been widely used as the base for many state-of-the-art models on the VMLU¹⁵ leaderboard, a leaderboard from ZaloAI and JAIST for evaluating Vietnamese LLM. This suggests that the llama2-7b [30] architecture is well-suited for handling the complex tasks and diverse knowledge requirements of the VMLU benchmark.
2. **Bilingual Support:** The bkai-foundation-models/vietnamese-llama2-7b-120GB [31] model has been pre-trained on a large corpus of both English and Vietnamese data. This makes it well-equipped to handle our goal of developing a bilingual language model capable of understanding and generating content in both languages.

¹⁵VMLU: <https://vmlu.ai>

3. **Model Size and Resource Efficiency:** The 7 billion parameter size of the llama2-7b architecture along with load of methods for efficient training like peft [74] allows the model to be trained on a single GPU, making it a practical and resource-efficient choice for our development process.
4. **Maturity and Stability:** Even though the llama2-7b model is relatively older (released in July 2023), it is a well-understood and widely-implemented architecture. This maturity and stability provide a solid foundation for our Instruction Tuning efforts.

By starting with the bkai-foundation-models/vietnamese-llama2-7b-120GB [31] model and fine-tuning it on the diverse dataset described in the previous sections, we aim to create a highly capable Instruction-Tuned language model that can excel on the VMLU benchmark and serve as a versatile personal assistant for our users.

The Instruction Tuning process will allow us to further refine the model's understanding of instructions and task completion, while the bilingual dataset will ensure that the model maintains strong proficiency in both English and Vietnamese. This approach will result in a language model that can seamlessly navigate between the two languages, providing a seamless and natural interaction experience for our users.

4.1.5 Model training

Fine-tuning large language models is a highly effective approach to enhancing their performance and capabilities. This process involves taking a pre-trained, large-scale language model and further training it on a specific task or dataset, allowing the model to specialize and improve its performance on that particular domain. Fine-tuning can also be used to instill desirable behaviors or mitigate undesirable ones in the model.

However, the process of fine-tuning very large language models can be prohibitively expensive. For example, the standard 16-bit fine-tuning of a LLaMA 65B parameter model requires more than 780 GB of GPU memory (VRAM), which is a significant hardware requirement that may not be readily available to many researchers and developers.

Low Rank Approximation (LoRA) [75] freezes pre-trained model weights and adds rank decomposition matrices into each layer of the transformer, which is different from full fine-tuning methods. This reduces the number of trainable parameters for downstream tasks, the time and cost of training. For example, the paper "Platypus: Quick, Cheap, and Powerful Refinement of LLMs" [76] found that a **lora training session of a 13B parameters model** fine-tuned using **1 A100 80GB took 5 hours** while in comparison, a **full fine-tuning of Alpaca-7B [71]** from Stanford took **3 hours on 8 A100s 80GB**

Along with lora, recent advancements in quantization techniques have shown promise in reducing the memory footprint of large language models. These quantization methods involve compressing the model parameters by reducing the number of bits used to represent them, effectively reducing the overall memory consumption. However, these techniques are primarily designed for inference and often break down during the training process, limiting their applicability for fine-tuning large language models.

QLORA [3] is a novel technique that aims to address the challenge of expensive fine-tuning of large language models. It uses a high-precision quantization method to compress the base model down to just 4-bit precision, dramatically reducing the memory footprint as shown in figure 4

By combining high-precision quantization of the base model with the addition of learnable adapter weights, QLORA [3] provides an efficient and cost-effective approach to fine-tuning large language models. This can significantly expand the accessibility of these powerful models to a wider range of researchers and developers who may not have access to the massive GPU resources required for traditional fine-tuning methods.

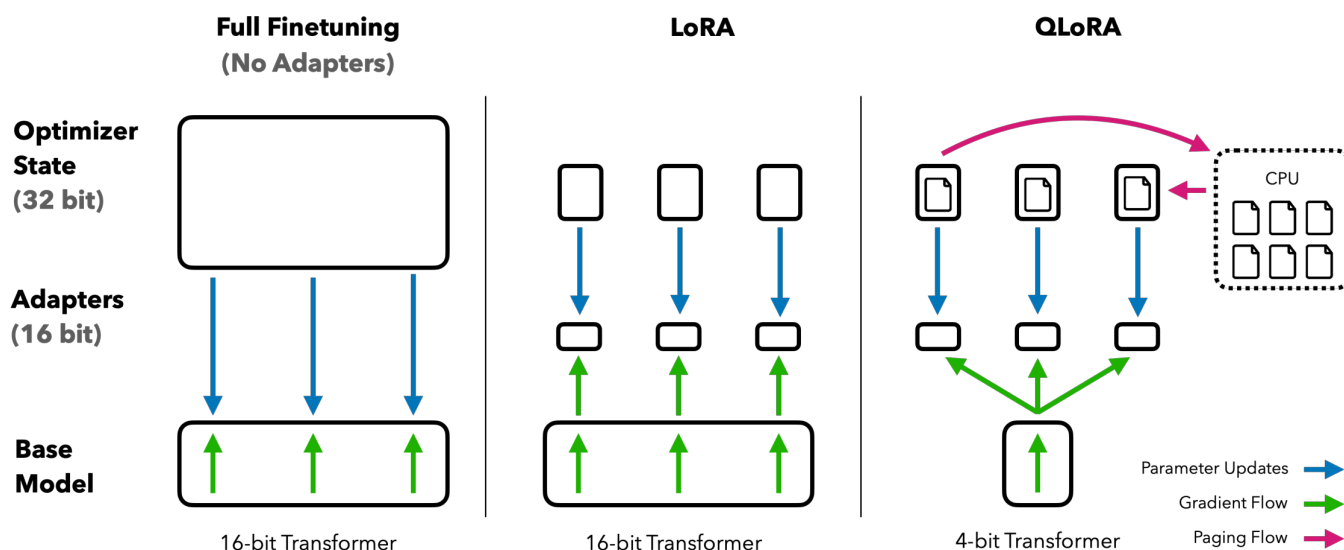


Figure 4: Comparison between full fine-tuning, Lora and Qlora

One of the key advantages of the QLoRA approach is its flexibility and modularity. In addition to the base quantized model and the learnable Low-rank Adapter weights, QLoRA also allows for the mixing of multiple LORA modules during the fine-tuning process.

The idea behind mixing LORA modules is to leverage the complementary strengths of different adaptation strategies. Each LORA module can be trained on a specific task or dataset, and then these modules can be combined to create a more versatile and capable fine-tuned model.

For example, one LORA module might be trained on a task related to natural language understanding, while another module is trained on a task involving commonsense reasoning. By mixing these LORA modules during fine-tuning, the model can acquire and retain the specialized knowledge from each module, resulting in improved overall performance on a wide range of tasks.

The mixing of LORA modules is facilitated by the low-rank nature of the adapter weights. These weights are typically much smaller than the original model parameters, making it computationally and memory-efficient to combine multiple LORA modules during inference.

The training stack is the combination of:

- deepspeed ¹⁶
- Huggingface:
 - pytorch ¹⁷ [77]
 - transformers ¹⁸ [78]
 - peft ¹⁹ [74]
 - accelerate ²⁰ [79]
 - datasets ²¹
 - optimum ²²

¹⁶deepspeed: <https://github.com/microsoft/DeepSpeed>

¹⁷pytorch: <https://github.com/pytorch/pytorch>

¹⁸transformers: <https://github.com/huggingface/transformers>

¹⁹peft: <https://github.com/huggingface/peft>

²⁰accelerate: <https://github.com/huggingface/accelerate>

²¹datasets: <https://github.com/huggingface/datasets>

²²optimum: <https://github.com/huggingface/optimum>

- bitsandbytes ²³
- wandb ²⁴

Our training configuration is as follow:

Hardware Type:

- GPU: VGA NVIDIA Tesla P100 16GB
- SYSTEM RAM: 32GB

Training Procedure:

Enhancing Reasoning Capabilities through LORA Adapter Mixing:

One of the key methods we employ to improve the reasoning capabilities of our base model (bkai-foundation-models/vietnamese-llama2-7b-120GB) is the mixing of LORA (Low-rank Adaptation) adapters.

Instead of training the base vietnamese-llama2-7b-120GB model from scratch, we take a state-of-the-art large language model (LLM) with the same architectural foundation (LLaMA 2-7B) that is currently ranked highly on the Open LLM leaderboard [80] for English, which is yeen214/llama2_7b_merge_orcafamily ²⁵ as the time of training. We then merge this model with the 120GB pre-trained Vietnamese lora adapter from the bkai-foundation-models/vietnamese-llama2-7b-120GB(the only Vietnamese base model repository that included the lora adapter), rather than using the model that has already been merged.

²³bitsandbytes: <https://github.com/TimDettmers/bitsandbytes>

²⁴wandb: <https://github.com/wandb/wandb>

²⁵llama2-orcafamily: https://huggingface.co/yeen214/llama2_7b_merge_orcafamily

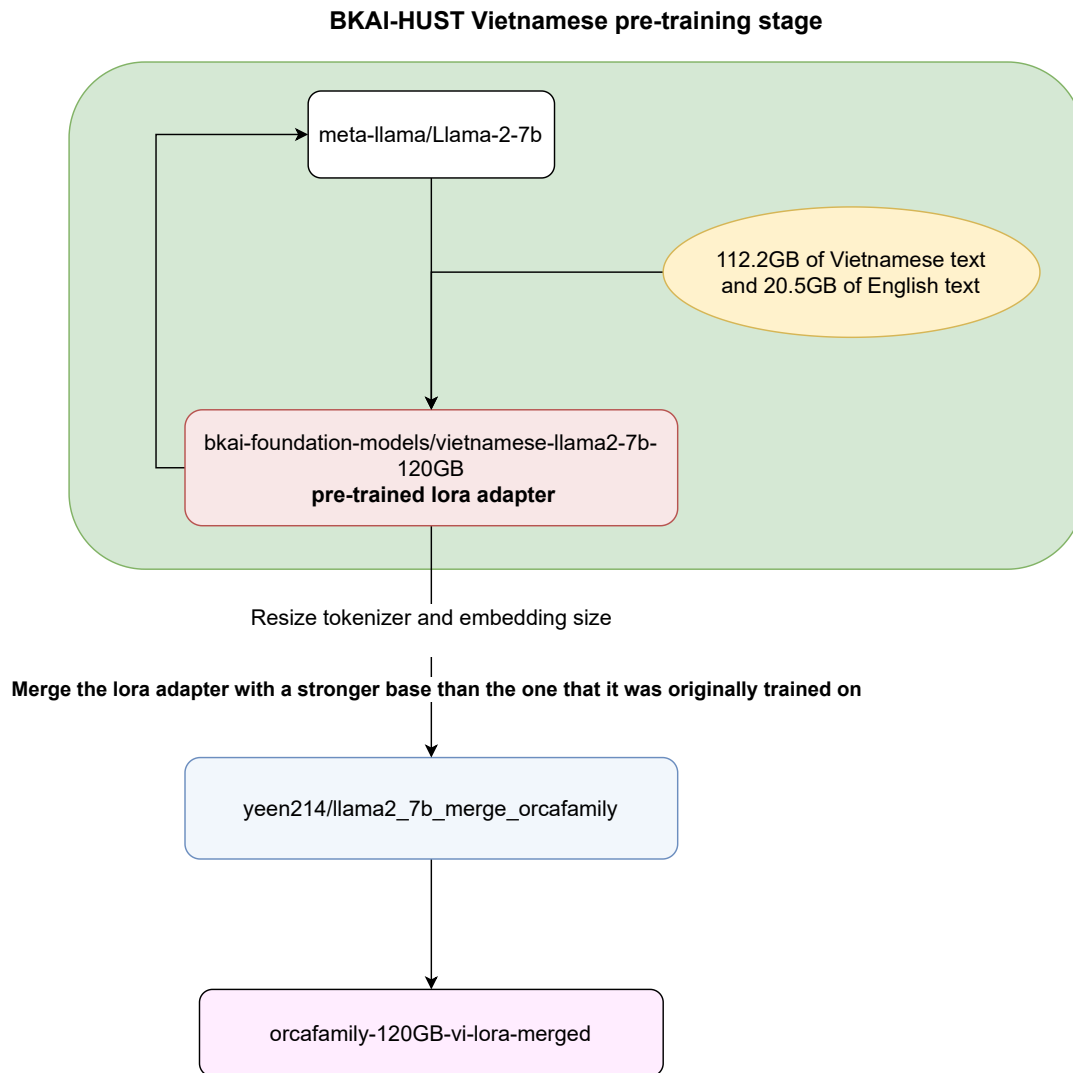


Figure 5: Visualization of how we create the base model

This result in a more capable base model that was good in logic and reasoning and has been merged with an adapter pre-trained on the Vietnamese language. We used this as our base model for instruction tuning on a mixed dataset of English and Vietnamese. Further discussion is on [6.1](#)

- Learning rate: 2e-5
- Learning rate scheduler: Cosine
- Optimizer: PagedLion8bit
- Precision: BF16/FP32
- Qlora config:
 - Rank: 64
 - Alpha: 64
 - Quantization: 4bit NF4

- Modules to save: embed_tokens, lm_head
 - Target modules: down_proj up_proj v_proj c_proj gate_proj k_proj q_proj o_proj
 - Bias: "none"
 - Lora dropout: 0.02
- Batch size: 2
 - Gradient accumulation: 128
 - Effective batch size: 256
 - Phase 1: 250k examples of 70% Vietnamese 30% English for 3.37 epochs
 - Phase 2: 350k examples of 60% Vietnamese 40% English for 1.4 epoch
 - GPU hours: 1,140* hours (47.5* days)

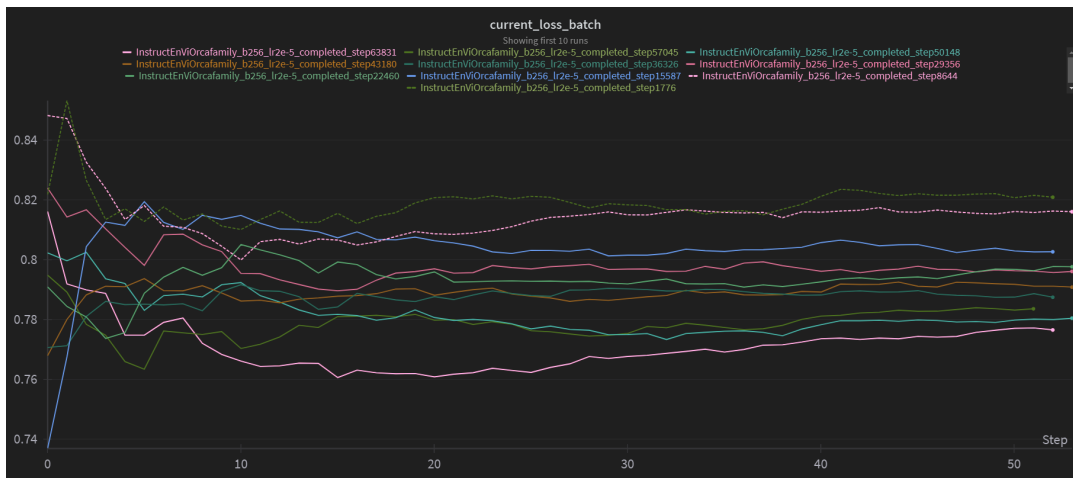


Figure 6: Last 10 runs

Final loss is: 0.7765

4.2 RAG Framework

4.2.1 Document Parsing

Sentence Splitter is an advanced tool designed to enhance readability and manageability of large texts by breaking them down into structured components. Here's a deeper look at how sentence splitters work and their importance in various applications. From there, it further segments these paragraphs into chunks of text. These chunks, or "slices," can vary in size and are defined either by the number of sentences they contain or by specific character length.

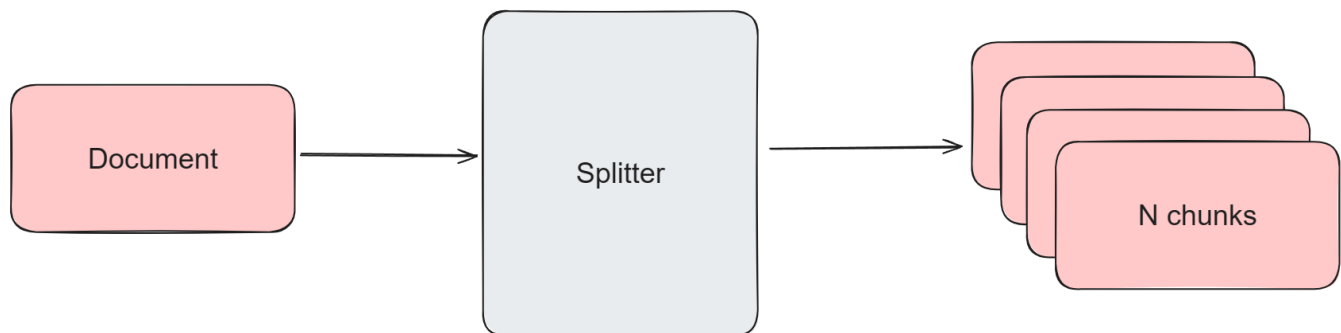


Figure 7: Splitting Process

Look at the Figure 7, the first step involves identifying the paragraph breaks within a document. This is typically based on formatting cues like newline characters and indentation patterns. After isolating paragraphs, the tool splits them into individual sentences. This involves understanding punctuation marks and capitalization rules that signify the end and beginning of sentences. In post segmentation, the sentences are grouped into chunks. These chunks can be uniform, containing a fixed number of sentences, or varied, depending on the specific requirements of the task at hand. For example, a chunk might comprise a single sentence, a paragraph, or a specific character count. All workflow is fomularized in Algorithm 1.

Algorithm 1 Text Splitting Algorithm

```

1: Input: text, chunk size, chunk overlap
2: Output: list of splits
3: function SPLITTEXT(text, chunk_size, chunk_overlap)
4:   paragraphs ← SPLIT(text, "\n\n\n")
5:   sentences ← []
6:   for paragraph in paragraphs do
7:     if LENGTH(paragraph) > chunk_size then
8:       sentences.extend(SPLITTEXT(paragraph, chunk_size, chunk_overlap))
9:     else
10:      sentences.extend(NLTKSENTENCETOKENIZER(paragraph))
11:    end if
12:  end for
13:  chunks ← []
14:  for sentence in sentences do
15:    if LENGTH(sentence) > chunk_size then
16:      chunks.extend(SPLITTEXT(sentence, chunk_size, chunk_overlap))
17:    else
18:      chunks.extend(REGEXSPLIT(sentence, "[.;] + [.,;]?"))
19:    end if
20:  end for
21:  words ← []
22:  for chunk in chunks do
23:    words.extend(SPLIT(chunk, " "))
24:  end for
25:  return words
26: end function
  
```

4.2.2 Indexing and Vector Database

Indexing is the process of organizing data in a way that allows for efficient retrieval. In text-based applications, this often involves parsing large documents into smaller, manageable chunks and then creating indexes that help locate these chunks based on content. Essentially, an index serves as a map to the original data, enabling quick searches without scanning the entire dataset.

Vector databases store data in the form of vectors, which are arrays of numbers representing data points in a high-dimensional space. These databases use machine learning algorithms to convert text or images into these numerical vectors. Indexing these vectors allows for semantic searches, where the intent behind a query is understood better than through simple keyword matches.

Milvus is an open-source vector database designed to provide scalable, reliable, and efficient search capabilities over large-scale vector data. Here's how Milvus aids in indexing:

- **Flexibility:** Milvus supports multiple index types, including FLAT, IVF, and HNSW, which cater to different needs in terms of balance between search speed and accuracy.
- **Scalability:** With its ability to handle billions of vectors and respond to queries in milliseconds, Milvus is ideal for enterprises with large datasets.
- **Integration:** Milvus can be easily integrated into existing systems, supporting popular programming languages and ready to use with only a few lines of code.

FAISS is a library for efficient similarity search and clustering of dense vectors developed by Facebook AI Research. It is particularly well-suited for applications where speed is critical. Key features of FAISS include:

- **High Performance:** FAISS uses optimized algorithms to index large vector sets for fast retrieval, even on large scales.
- **Specialized Indexing:** It offers specialized indexing strategies for different scenarios, including exact and approximate nearest neighbor search.
- **Resource Efficient:** FAISS is optimized to run on GPUs, providing significant accelerations in processing speed.

Combining Vector Databases and Indexing, this become a cornerstone of modern data management, crucial for dealing with the vast amounts of data generated daily. Vector databases like Milvus and FAISS not only facilitate efficient data retrieval but also enhance the capability to conduct searches that understand the context and semantics of the data. Look at Figure 8 (above), the process can be visualized clearly

4.2.3 Retriever Module

The landscape of data retrieval is continually evolving, with sophisticated algorithms enabling more precise and efficient information retrieval. Among these, BM25 stands out in sparse retrieval, while vector databases like Milvus and FAISS excel in dense retrieval. When integrated into the RAG framework, these technologies enhance the generation of accurate and contextually relevant content. Here, we delve into how these methodologies work individually and in concert within the RAG framework.

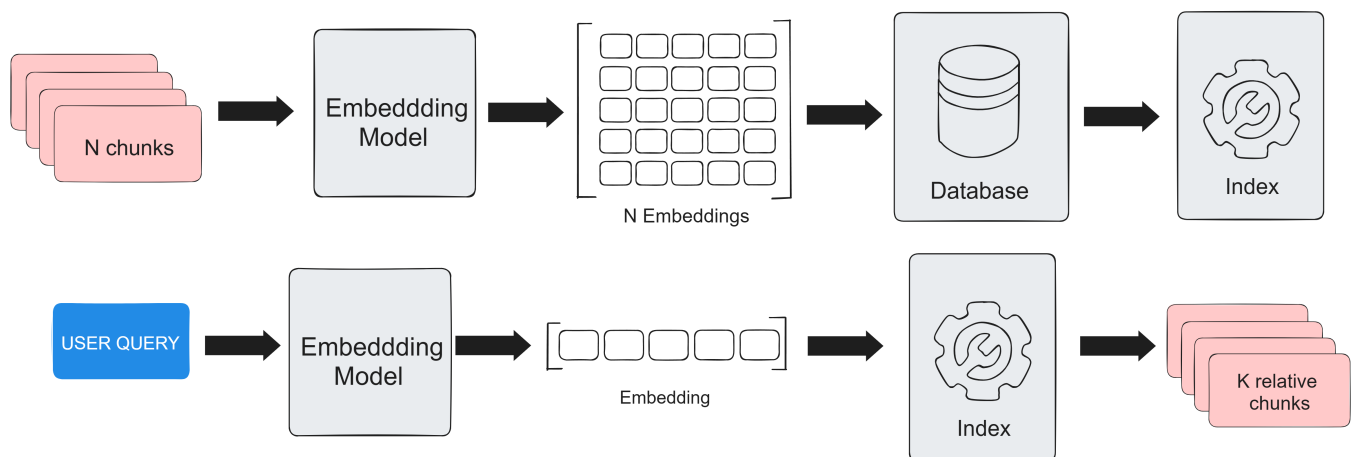


Figure 8: Indexing And Searching Query Process

Sparse Retrieval with BM25 BM25 is a ranking function used in text retrieval that calculates the relevance of documents to a given search query based on the frequency of query terms in each document. Unlike simple term frequency methods, BM25 accounts for term frequency saturation (diminishing returns of term frequency) and document length normalization (preventing longer documents from dominating simply due to size):

- **Term Frequency (TF):** BM25 modifies the simple term frequency to include a saturation point, which prevents a term from excessively influencing the document's relevance.
- **Inverse Document Frequency (IDF):** This component diminishes the weight of terms that appear more frequently across all documents, helping to highlight more distinctive terms.

This retriever operate on the principle of direct term matching, which is highly effective in environments where the exact match of key terms between the query and documents can indicate relevance. By weighting terms not just by frequency but also by their distribution across documents, BM25 allow retriever to prioritize terms that are common in relevant documents but rare elsewhere.

Vector Stores in Dense Retrieval In contrast to BM25, Milvus and FAISS are designed for dense retrieval, where data is represented as high-dimensional vectors. These vectors are generated from data through deep learning models, encapsulating much richer semantic information than possible with sparse, term-based approaches. Look at Figure 8, after building index using vector databases, this index then is used to retrieve relevant chunks by computing embedding of user query, and then scoring similarities between query and encoded information.

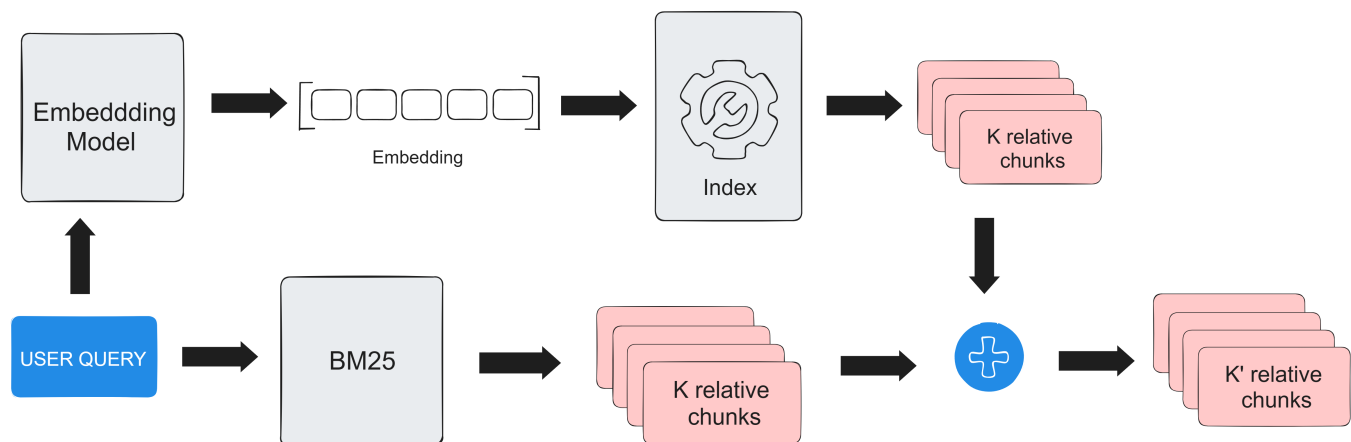


Figure 9: Hybrid Searching Process

Hybrid Retrieval In the dynamic field of information retrieval, combining sparse and dense retrieval methods has emerged as a powerful strategy to optimize both accuracy and efficiency. This hybrid approach leverages the unique strengths of both retrieval types to address the limitations of relying on either method alone. Here, we delve into how this combination works and its implications for advanced search and retrieval systems. The hybrid method integrates sparse and dense retrieval to create a more robust and comprehensive retrieval system. Here's how this integration typically unfolds:

- **Initial Filtering with Sparse Retrieval:** The process begins with sparse retrieval, where an algorithm like BM25 quickly narrows down the potential data pool. This step is effective at swiftly eliminating irrelevant data based on keyword matching, significantly reducing the search space and improving efficiency.
- **Refinement with Dense Retrieval:** After the initial filtering, dense retrieval comes into play. The pre-selected data from the sparse phase is further analyzed with vector-based systems, which assess the semantic relationships between the query and the documents. This stage allows for the retrieval of documents that may not have exact keyword matches but are semantically relevant to the query.

Advantages of Hybrid Retrieval: By combining both retrieval methods, the system can leverage the precision of keyword matching and the depth of semantic understanding, leading to more accurate results. Sparse retrieval's efficiency in narrowing down the dataset reduces the computational load for the more resource-intensive dense retrieval, making the overall process faster and more scalable. The hybrid approach is adaptable to a variety of data types and query complexities, making it suitable for diverse applications, from academic research databases to commercial search engines. The process can be seen as Figure 9:

4.2.4 Advance Rerank Method

In the realm of information retrieval, continuously evolving techniques aim to refine the precision and efficiency of search results. Advanced methods like Model Re-ranking leverage modern natural language processing (NLP) technologies to enhance the quality of search outcomes.

Algorithm 2 Delta Similarity Chunk Postprocessing

```

1: Input: List of chunks  $\{(s_1), (s_2), \dots, (s_N)\}$ , delta  $\delta$ 
2: Output: List of chunks filtered based on delta similarity criterion
3: Determine the highest score:

$$s_{\text{best}} = \max(s_1, s_2, \dots, s_N)$$

4: Initialize the list of new chunks: new_chunks = []
5: for each node  $i$  with score  $s_i$  do
6:   if  $s_{\text{best}} - s_i \leq \delta$  then
7:     Add node  $i$  to new_chunks
8:   end if
9: end for
10: return new_chunks

```

These techniques, often implemented using sophisticated models from resources like Sentence Transformers, represent the cutting edge of AI-driven search and retrieval mechanisms. The results from both retrieval methods are then integrated into a unified ranking system. This ranking may involve re-scoring documents based on a combination of their sparse and dense retrieval scores, optimizing both for relevance and contextual similarity.

Delta Mean Rerank We combine this implementation after using retriever module, which primarily used for filter the δ gap scores between the initial set of retrieve chunks after using either sparse or dense retrieval methods. This postprocessing way help model eliminate redundant information as well as avoid "hallucination" when generating. Algorithm is describe as Algorithm 2

Model Re-ranking After filtering unnessary chunks, a re-ranking model evaluates the relevance of each chunk to the query. Due to the first filter for removing redundant things, model then adjusts the ranking to prioritize those quality chunks that are most likely to be useful to the user. The goal is to ensure that the results most relevant to the user's refined query are presented first, improving user satisfaction and efficiency. Our algorithm can be visualize as:

Algorithm 3 Reranking Retrieved Chunks

```

1: Input: query, retrieve chunks
2: Output: reranked_chunks – chunks sorted by relevance
3: function RERANKCHUNKS(retrieve_chunks, query)
4:   relevance_scores  $\leftarrow$  compute relevance scores for each node
5:   chunks_sorted  $\leftarrow$  sort chunks by relevance_scores in descending order
6:   return chunks_sorted
7: end function
8: reranked_chunks  $\leftarrow$  RERANKCHUNKS(retrieved_chunks, query)
9: return reranked_chunks

```

Look at Figure 10 for better visualization. Generally, some Sentence Transformer models are particularly suited

for this task. They can generate embeddings for both the query and the documents, and then a similarity measure (such as cosine similarity, euclidean distance) can be used to re-rank the results based on semantic closeness to the query. Advanced models might also incorporate user interaction data to learn which types of documents tend to satisfy similar queries, further refining their re-ranking algorithms.

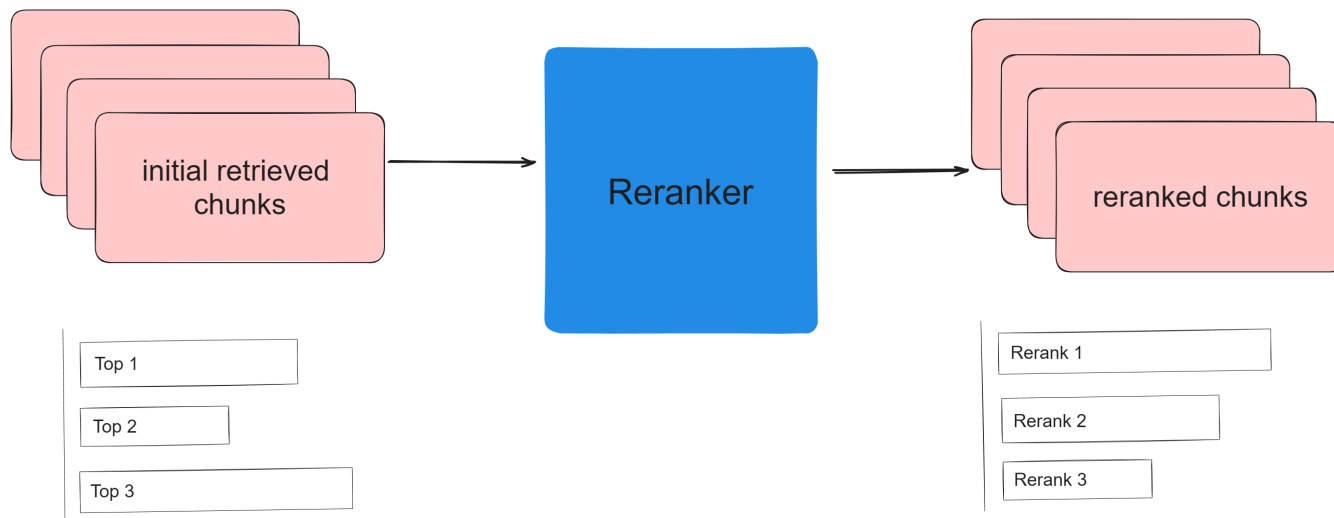


Figure 10: Rerank Visualization

Mean Rerank Even after reranking, there are some case where the score between reranked chunks existing the large gap, that why's this Mean Rerank come into play. This method seem to be a second filter which is mainly for continuing to remove the chunks that has low score similarity. The algorithm can be view in Algorithm 4:

Algorithm 4 Mean Similarity Node Postprocessing

- 1: **Input:** List of chunks $\{(s_1), (s_2), \dots, (s_N)\}$
- 2: **Output:** List of filtered chunks based on mean score criteria
- 3: Calculate the mean score:

$$\bar{s} = \frac{1}{N} \sum_{i=1}^N s_i$$

- 4: Initialize the new node list: `new_chunks = []`
 - 5: **for** each node i with score s_i **do**
 - 6: **if** $s_i \geq \bar{s}$ **then**
 - 7: Add node i to `new_chunks`
 - 8: **end if**
 - 9: **end for**
 - 10: **return** `new_chunks`
-

Those reranking methods significantly enhance the relevance of search results, directly impacting user experience and satisfaction. Future research could explore more advanced machine learning models to understand context and user intent better, further improving reranking processes.

4.2.5 Complete pipeline

The RAG framework combines the strengths of retrieval-based and generative models to enhance machine learning tasks such as question answering and content generation. Here's how BM25, vector stores, rerankers integrated within this framework:

- **Document Retrieval:** Initially, the RAG uses an algorithm like BM25 for sparse retrieval to quickly filter down the vast amount of available content to a manageable subset of relevant documents. This step is crucial for efficiency in large datasets.
- **Vector-based Augmentation:** Following the initial retrieval, dense vector stores like Milvus or FAISS refine the search to identify the chunks most semantically related to the query. This step leverages the dense embeddings to capture deeper, non-obvious relationships between the query and the document contents.
- **Reranking** Applying various method reranking after receiving retrieved chunks from Retrieval Module are the guardian for ensuring quality information, avoiding as much as possible the "hallucination" case, increasing credibilities for Generator.
- **Content Generation** The final stage involves a generative model that synthesizes the reranked information into coherent, contextually enriched responses. The generative model can adapt its output based on the nuanced understanding of the query provided by the dense retrieval phase.

By integrating these disparate technologies, the RAG framework not only speeds up the retrieval process but also significantly enhances the relevance and quality of the generated content. This hybrid approach represents a significant step forward in the field of AI-driven content generation, making it a pivotal tool for applications ranging from automated customer support to personalized content recommendation systems. All of pipeline are in Figure 11.

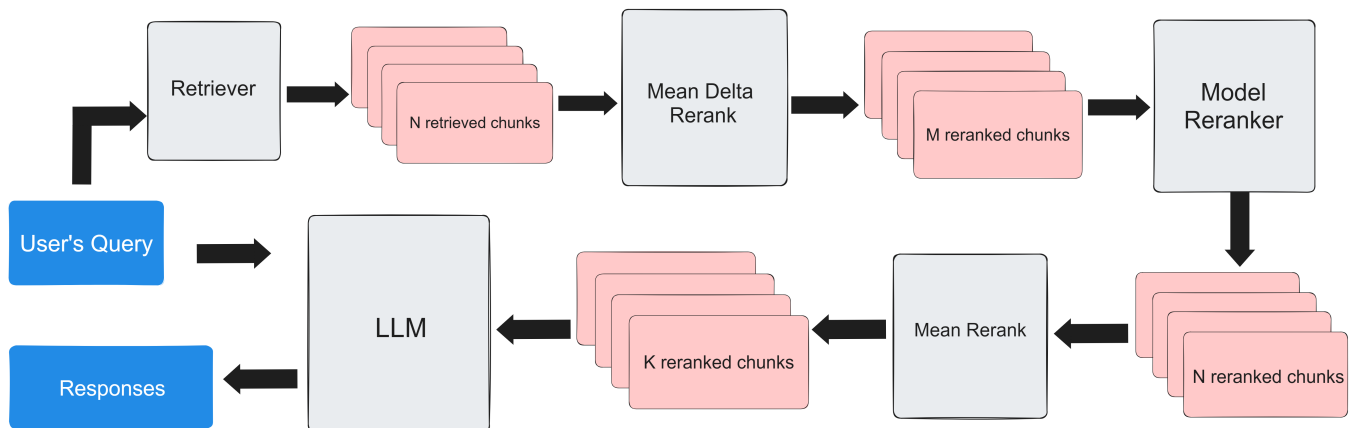


Figure 11: Our RAG Pipeline Visualization

4.3 TTS Dataset Collection and Description

The data collecting and processing pipeline with 3 stages is described in 12

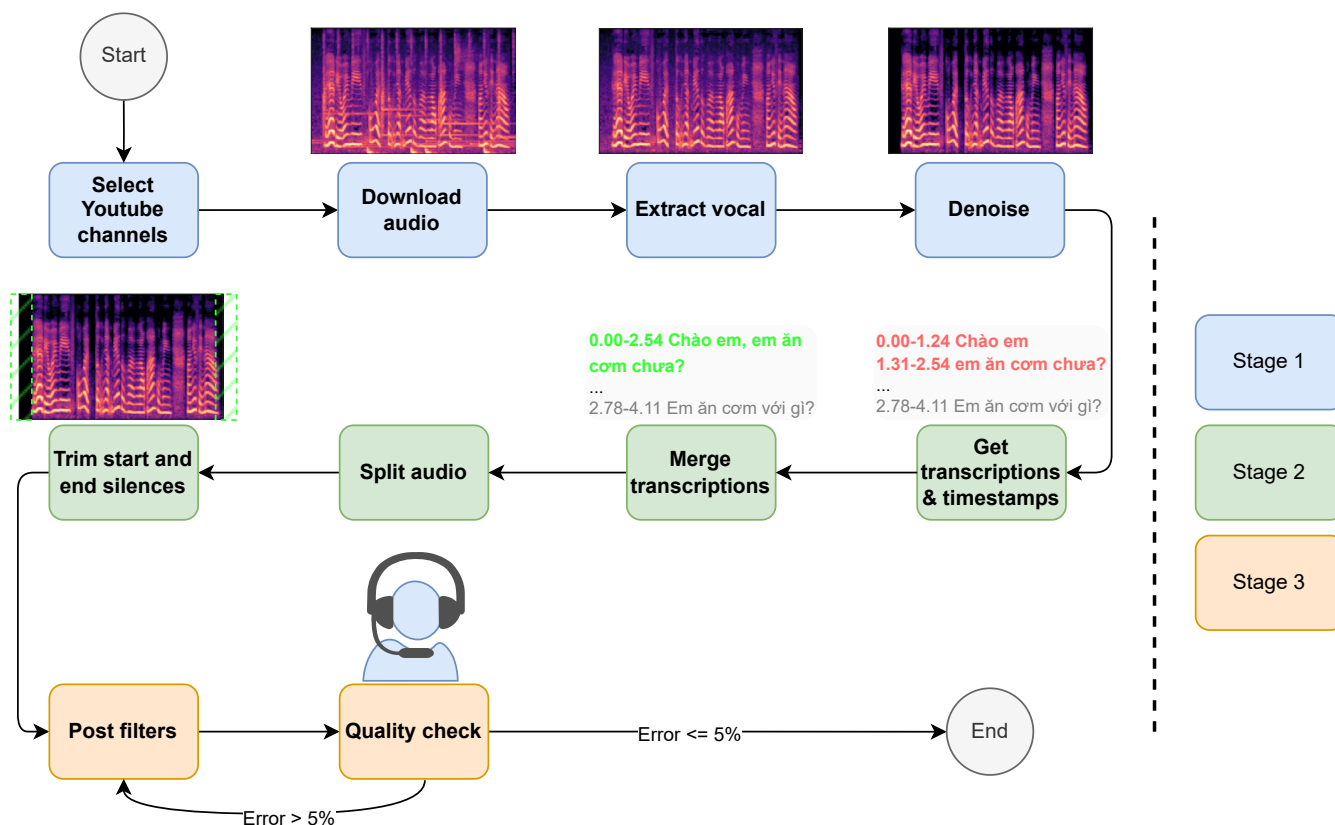


Figure 12: Data processing pipeline

4.3.1 Stage 1: Data Acquisition and Preprocessing

We manually curated a list of 186 Vietnamese YouTube channels spanning various genres (e.g., audio books, interviews, news, how-to, etc.) to ensure diversity in the speech data. Care was taken to select channels that adhered to the following criteria:

- Avoiding channels with a free-style speaking pattern containing many stop words or hesitations.
- Avoiding channels with overlapping speeches, ensuring a maximum of one person speaking at a time.
- Avoiding channels with machine generated speeches.
- Excluding channels that heavily utilized sound effects.

Downloading Audios: The audio from these channels was downloaded in their best quality possible, typically in opus format. A custom Python script utilizing the yt-dlp²⁶ library was used to download the audio files. Approximately 1,500 hours of raw audio data were downloaded.

Vocal Extraction: We utilized the UVR-MDX-Net-Voc_FT, a deep learning model acclaimed for its top performance in the Music Demixing Challenge [81], particularly in vocal/instrumental separation. This model effectively isolates vocal and speech segments from background music and other non-speech audio elements. We retained only the vocal track from the processed audio files. Subsequently, the output audios were converted to 24 kHz, 16-bit PCM, mono-channel format with ffmpeg [82] to optimize storage space and computational resources for subsequent stages.

²⁶<https://github.com/yt-dlp/yt-dlp>

Noise Removal: DeepFilterNet [83] can maintain the natural clarity and integrity of speech, ensuring that the resulting audio remains clear and authentic even after noise removal. Given our objective of preserving high-quality voice output, DeepFilterNet emerges as the optimal choice. Moreover, its remarkable speed significantly reduces processing costs, making it an efficient solution for our needs.

4.3.2 Stage 2: Get Transcription and Timestamps

Getting Transcriptions: We utilize Whisper Large V3 [84] to obtain transcriptions, as this model is publicly accessible and demonstrated superior performance in our small-scale tests. The default model settings are employed to generate transcriptions and timestamps.

Merging Transcriptions: The Whisper ASR system, as outlined in its paper, presented several notable limitations. We encountered challenges, including occasional premature sentence segmentation and inaccurate word-level timestamps, which made it difficult to split audio files mid-word. To ensure coherence in our transcriptions, we devised a rule-based approach that leveraged the behavior of Whisper Large V3 in handling punctuation outputs. The rule is straightforward: if the current sample text does not end with punctuation and the starting character of the next sample is not capitalized, then they are merged (text, audio timestamp). This method aimed to consolidate fragmented transcriptions into cohesive text sentences for each audio file.

Silence Trimming: In further refining our audio segments, we returned to Silero VAD. This time, we employed it to identify and timestamp the leading and trailing silence within each segmented audio file. By incorporating additional padding to prevent excessive intrusion into the audio content, we effectively removed any silent portions from the beginning or end of each segment, resulting in overall cleaner audio data.

4.3.3 Stage 3: Post-Filtering and Quality Assurance

Post-Filtering Criteria:

- Samples with transcriptions exceeding 250 characters or 80 words were discarded (primarily due to transcription and merging errors).
- Samples with maximum silence intervals exceeding 1.5 seconds, as detected by Silero VAD, were discarded.
- Samples exceeding 500 words per minute (WPM) were excluded.
- Duplicate samples were removed.
- For transcribing segmented audio, we employed PhoWhisper [85] using the base-size model, chosen for its state-of-the-art performance in various Vietnamese ASR benchmarks. The selection of the base-size model was based on its speed and its ability to retain stopwords and hesitations, features often disregarded by Whisper Large V3. To ensure consistency, we standardized both the PhoWhisper and Whisper Large V3 transcripts by removing punctuation, converting all characters to lowercase, and representing them in ASCII form. Subsequently, we compared the two transcripts, filtering out samples displaying discrepancies. Additionally, we established minor rules to address cases where the transcripts did not match perfectly but were deemed acceptable.

Manual Quality Check:

- We developed a Streamlit²⁷ app to streamline the manual inspection process. This app empowered annotators to visualize both the audio waveform and mel-frequency spectrogram for each sample, alongside its transcription. This visualization enabled the identification of common audio and transcription errors (e.g., missing or duplicated words) as well as audio issues (e.g., noise or mis-cuts).
- Annotators had the flexibility to filter and select samples based on various criteria, including text, notes, or custom filters. The app also provided statistical insights, such as the number of selected rows, rows with notes, and filtered rows.

²⁷<https://streamlit.io/>

- One human annotator utilized this app. Each inspection session involved randomly sampling up to 1000 samples for assessment. We determined this sample size to be statistically significant and efficient for evaluating transcription accuracy and audio quality. However, if the error rate exceeded a certain threshold (typically between 100 to 200 samples), we stopped early to prevent unnecessary effort. The annotator could also attach notes or flags to samples requiring further review or correction.
- Based on the notes taken, new filtering rules are iteratively added. This process ensured that new filter rules were continually added to enhance the accuracy of our quality checks.
- We repeated this inspection and refinement process until achieving a final error rate below 5%, ensuring high-quality data for our analysis.

We present a comparison of speech and text samples from our dataset to others in Appendix F. This comparison serves to highlight the unique characteristics and potential applications of our dataset in relation to existing speech and text corpora.

4.4 viVoice Statistic

We present a detailed analysis of the viVoice dataset, outlining its composition, and characteristics in comparison to existing corpora.

4.4.1 Dataset Overview

The viVoice dataset is a comprehensive resource for Vietnamese multi-speaker text-to-speech synthesis. Key statistics are summarized in Table 4.4.1. This dataset contains 887,772 speech samples, totaling 1016.97 hours of audio, sourced from 186 YouTube channels. It is divided into four parts, with Part 01 being the largest, comprising 338,921 samples from 43 speakers and totaling 392.79 hours. Part 03 also features a substantial 280,857 samples across 85 speakers, spanning 287.98 hours. Parts 01 and 03 offer diverse content (e-books, news, cooking, podcasts, etc.), while Part 02 exclusively contains audiobooks from two prominent Vietnamese providers. Part 04 focuses on news-related content. To understand the viVoice dataset’s composition, metrics like the male:female ratio and error rate were carefully estimated for each subset and the dataset overall. A random sample of 1000 speech samples from each subset was manually inspected to determine these values. Results are presented with a 95% confidence interval to highlight the statistical reliability of our estimations. A detailed explanation of error transcription classification criteria can be found in the Appendix G.3.

The viVoice dataset exhibits a noticeable gender imbalance across its various parts. While Parts 01 and 02 demonstrate a predominant male presence, with 69.10% and 61.70% male proportions respectively, Parts 03 and 04 reveal a substantial shift towards a more balanced gender distribution, with male proportions dropping to 53.50% and 30.50% respectively. Notably, Parts 03 and 04 also exhibit higher error rates in transcriptions compared to other segments. This discrepancy in error rates is attributed to spontaneous or informal conversational speeches, posing challenges for accurate transcription. These observations underscore the importance of considering gender balance and speech characteristics when analyzing and utilizing datasets for various applications.

The total audio duration of each channel is illustrated in Figure 13. The initial two channels are associated with Part 2, containing audio book content. Consequently, these channels intentionally feature a longer audio duration compared to the others. This can be attributed to the suitability of their content for text-to-speech (TTS) applications. Audio books generally offer well-enunciated, clear speech, rendering them ideal for TTS model training.

Table 4: Summary of subsets of viVoice dataset. This table summarizes key metrics including the number of speakers, total samples, duration in hours, male proportion and error text proportion for various parts of the viVoice dataset, as well as the combined statistics for the entire dataset.

Part	Speakers	Samples	Duration (Hrs)	Male (%)	Error Rate (%)
Part 01	43	338921	392.79	69.10 \pm 2.87	0.50 \pm 0.44
Part 02	2	199681	226.59	61.70 \pm 3.01	0.10 \pm 0.20
Part 03	85	289857	287.98	53.50 \pm 3.09	3.50 \pm 1.07
Part 04	56	59313	109.61	30.50 \pm 2.85	3.10 \pm 1.07
viVoice	186	887772	1016.97	61.30 \pm 3.02	1.80 \pm 0.82

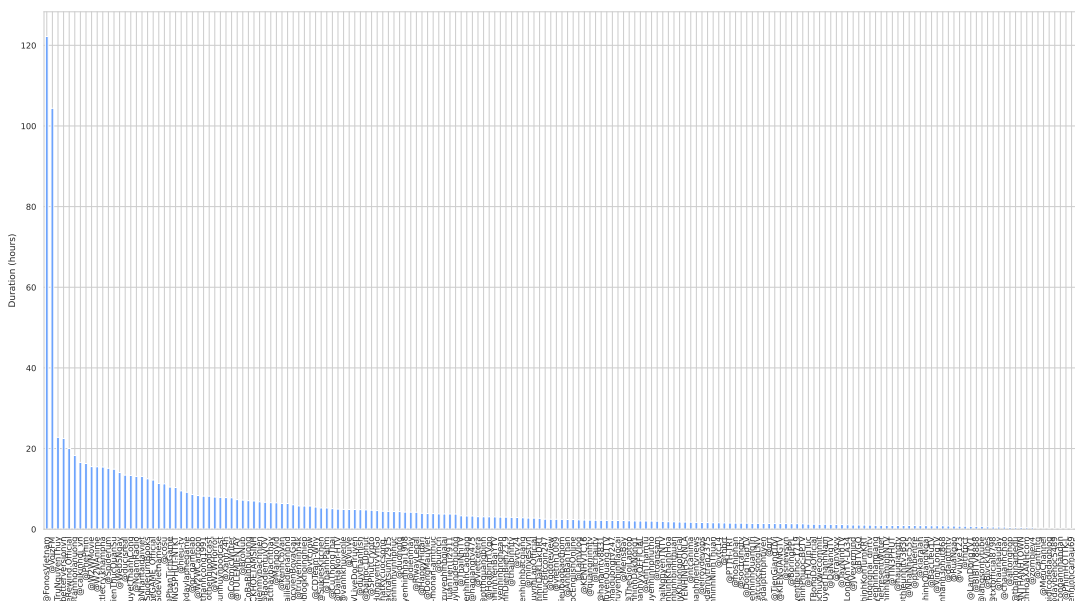


Figure 13: Duration distribution per channel.

4.4.2 Utterance Duration Characteristics

The violin plots in Figure 14 depict the distribution of utterance durations across several speech corpora, including viVoice, LibriTTS, BUD500, VIVOS, VinBigData, and VietnamCeleb. The distributions for viVoice and LibriTTS appear closely aligned, suggesting comparable characteristics in terms of utterance length. In contrast, the distributions for BUD500 and VIVOS exhibit a notable skew towards shorter durations. The VinBigData and VietnamCeleb corpora display distributions that fall between these two extremes, with a wider spread but still leaning towards shorter utterances. The observed variations across these datasets potentially highlight the impact of different segmentation methodologies employed during their construction, which can significantly influence the resulting distribution of utterance durations within each corpus.

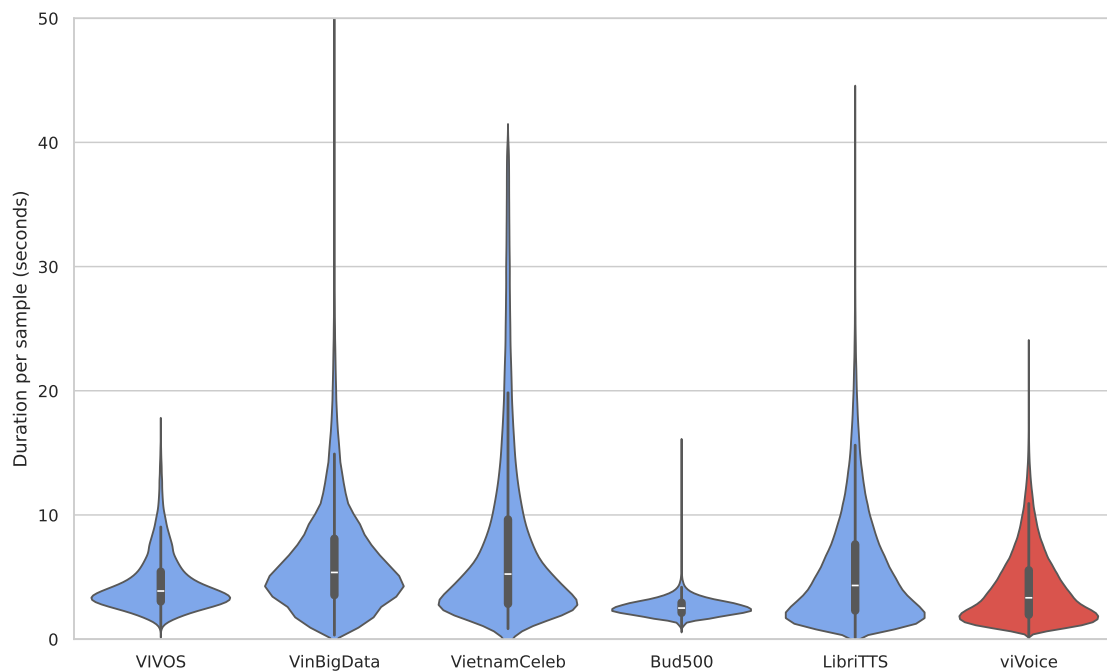


Figure 14: Violin plots illustrating the distribution of utterance durations across various speech corpora, including viVoice, LibriTTS, BUD500, VIVOS, VinBigData, and VietnamCeleb. To enhance visualization, the y-axis is capped at 50.

Figure 15 presents a box plot analysis of speaking rates across datasets, revealing a distinct characteristic of the vivoice dataset that has the potential to be beneficial in text-to-speech (TTS) applications. Compared to other datasets, viVoice demonstrates a higher median speaking rate, implying that its samples generally exhibit a faster pace. This characteristic could lead to more natural-sounding and engaging synthesized speech. Additionally, viVoice displays a narrower interquartile range, indicating a more consistent speaking rate distribution. This consistency may contribute to easier silence trimming and filtering processes within a TTS data pipeline. While the presence of outliers warrants consideration, the overall concentration of higher speaking rates makes viVoice a valuable dataset for TTS applications where a dynamic speaking style is desired.

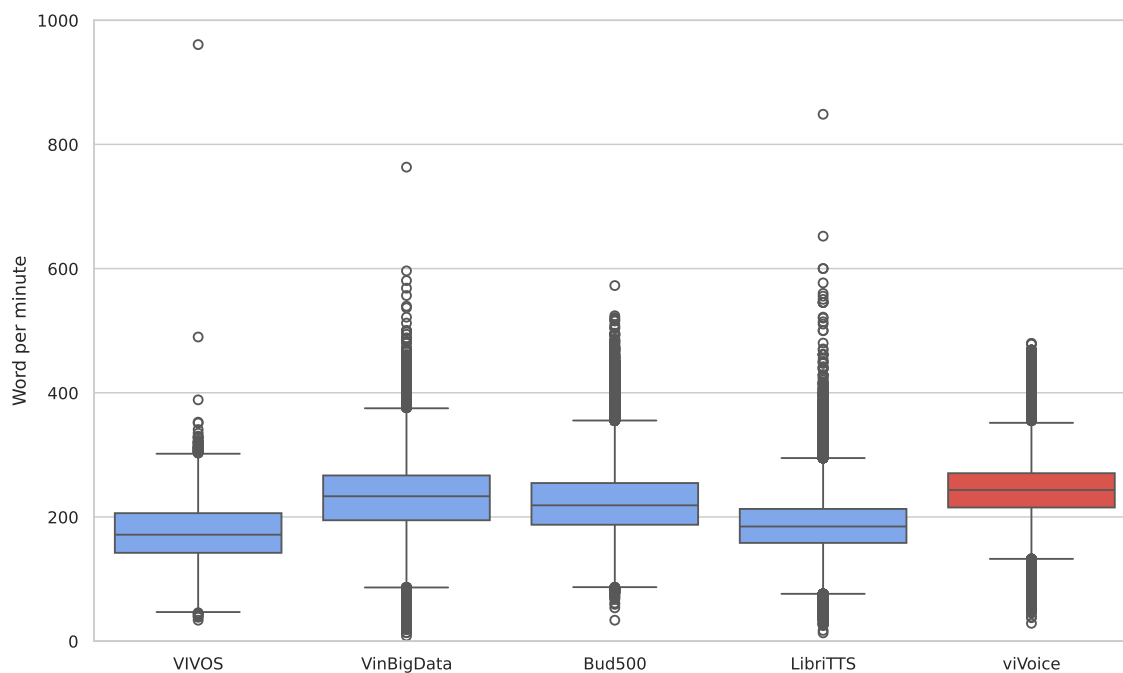


Figure 15: Box plots depicting the distribution of speaking rate (words per minute) across viVoice and other speech corpora, including VIVOS, VinBigData, BUD500, and LibriTTS

4.5 Experiments

In this section, we detail the experiments conducted to adapt the XTTSv2 [8] model to the Vietnamese language and evaluate its performance using subjective and objective metrics. We aim to assess the effectiveness of our curated dataset in enhancing TTS model performance compared to a baseline ASR dataset.

4.5.1 TTS Model Architecture

We utilize the XTTSv2, a LLM based TTS model, renowned for its natural-sounding voices across 17 languages and its unique cross-language voice cloning capability, including handling acronyms and numbers.

Given that XTTSv2 utilizes a multilingual backbone based on the GPT-2 model, and considering the successful adaptation of the GPT model [86] to Vietnamese text through adjustments to the tokenizer and fine-tuning on new datasets, we propose that a similar adaptation approach can be employed for XTTSv2 in Vietnamese speech synthesis. Our adaptation strategy primarily focuses on modifying the tokenizer component, as depicted in Figure 16, while keeping the model’s overall architecture intact.

Adapting XTTSv2 to Vietnamese required adjustments to the tokenizer to accommodate the Vietnamese character set and linguistic rules. We utilized the tokenizers library²⁸ to train a Byte Pair Encoding (BPE) tokenizer using the text from the VietTTS [20] as the training corpus, without any text modification or normalization. The process involved appending the language code "vi" and selecting the initial 1000 tokens from the resulting Vietnamese tokenizer, along with the token merge rule, to integrate with the XTTSv2 tokenizer. Tokens already present in the original tokenizer were excluded during merging, resulting in a tokenizer with 7544 tokens. Subsequently, new

²⁸<https://github.com/huggingface/tokenizers>

embedding vectors were initialized with Gaussian-distributed random values to match the newly added tokens. This adapted model was named viXTTS.

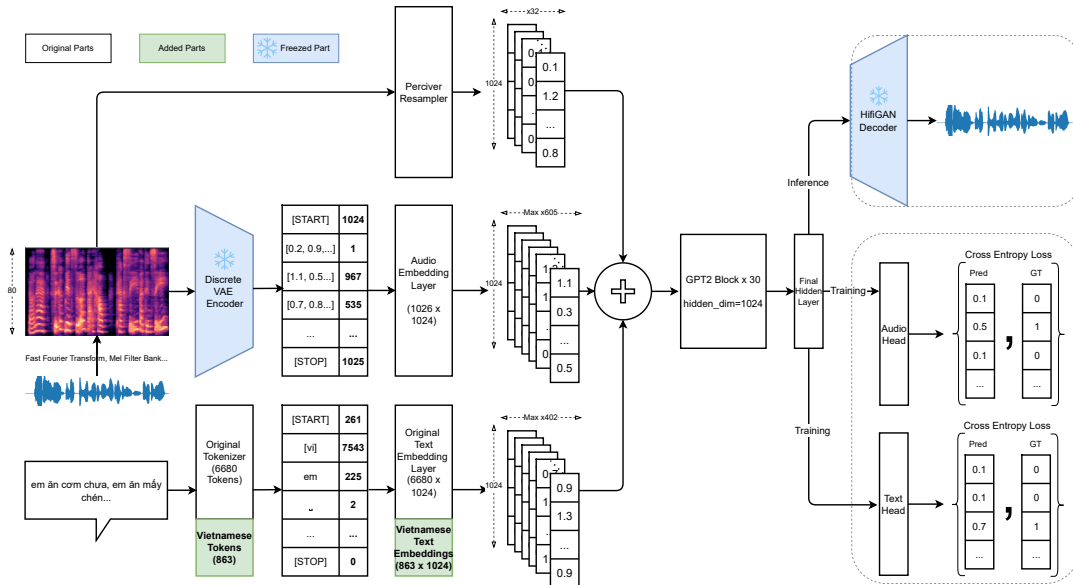


Figure 16: Abstract architecture of the XTTSv2 model. The newly added part is highlighted in green, indicating that the tokenizer is modified for adaptation to the Vietnamese language.

4.5.2 TTS Training

For the training process, we adopt minimal pre-processing steps for the input text, foregoing text normalization techniques as the original XTTS model inherently handles raw text input effectively.

We maintain consistency with the hyperparameters and optimization techniques used in the original XTTSv2 model. Further details regarding these configurations are available in the Appendix E. Specifically, we employ a real batch size of 2 and gradient accumulation of 16 during training, utilizing a single NVIDIA RTX 4060ti 16GB GPU.

To evaluate our dataset’s quality, we refined the viXTTS model using two distinct datasets: a 350-hour subset extracted from Parts 01 and 02 (with adjustments to balance total audio duration in the distribution) of our curated viVoice dataset, and the publicly available 500-hour Bud500 ASR dataset [18]. We excluded Parts 03 and 04 from consideration due to their unavailability at the time of evaluation. This deliberate selection allowed us to test a subject dataset of 350 hours against the 500-hour benchmark provided by the Bud500 ASR dataset, ensuring a focused comparison. Subsequently, we created three checkpoints: viXTTS trained on BUD500 for 1 epoch, viXTTS trained on viVoice for 1 epoch, and viXTTS trained on viVoice for 5 epochs.

4.6 TTS Evaluation

In this section, we discuss both objective and subjective evaluation metrics employed in our study.

4.6.1 Subjective Evaluation Metrics

Subjective metrics provide quantitative assessments of TTS system performance, with a focus on key measures such as:

- **Mean Opinion Score for Naturalness (nMOS):** nMOS evaluates the naturalness of each generated speech sample.

- **Mean Opinion Score for Similarity (sMOS):** sMOS assesses the similarity between synthesized speech and a reference human speech, offering valuable insights into synthesis quality.

While controversies exist surrounding these metrics, including concerns about bias [87] and their relative nature [88], we continue to utilize them as nMOS and sMOS are widely recognized metrics in the field of speech synthesis [6,58,64,65]. We do so while maintaining transparency in our framework, application, and data to mitigate these limitations. Our aim is to provide a foundation for future research to replicate and build upon.

The VIVOS dataset [14] was chosen for its public availability and established usage in research. This promotes transparency and reproducibility compared to proprietary datasets.

For a thorough evaluation, we meticulously curated two samples from each of the 19 speakers in the VIVOS dataset, following the methodology outlined in [89]. These samples were chosen to facilitate assessment using both nMOS and sMOS metrics. Additionally, we generated three additional samples for each original prompt sample, representing different training checkpoints (Bud500 1 epoch, viVoice 1 epoch, and viVoice 5 epochs).

To minimize bias, we introduced modifications to the reference audios using audiomentations²⁹. These changes were intended to make it more difficult for participants to distinguish the ground truth (prompt) audio when compared to the reference. Additionally, we adjusted the sampling rate of the synthesized audios to 16 kHz to match that of the prompt audio.

Complementing this dataset selection, we created a web-based evaluation tool using Gradio [90] to measure Mean Opinion Score (MOS) for speech samples. This tool allows users to listen to samples, rate their naturalness and similarity on a 1 to 5 scale, and calculate MOS scores from these ratings. Detailed evaluation criterias are provided in the Appendix 34.

4.6.2 Objective Evaluation Metrics

Objective metrics are calculated using publicly available methods. We employ the following metrics:

- **Word Error Rate (WER) and Character Error Rate (CER):** WER and CER are metrics used to evaluate the accuracy of synthesized speech compared to the reference text. To compute these metrics, we employ the PhoWhisper-large model to generate the ground truth and synthesized text. Subsequently, we calculate WER and CER using the jiwer³⁰ library.
- **Equal Error Rate (EER):** EER metric provides insights into the system’s ability to authenticate speakers. Follow [65, 68], we calculated EER using ECAPA-TDNN [91], automatic speaker verification. We use the checkpoint provided by SpeechBrain [92] on HuggingFace³¹.
- **Speaker Encoder Cosine Similarity (SECS):** SECS measures the cosine similarity between the speaker representations of the target speech and the synthetic speech. We utilize the cosine similarity value returned by the same ECAPA-TDNN model for this metric.

²⁹<https://github.com/iver56/audiomentations>

³⁰<https://github.com/jitsi/jiwer>

³¹<https://huggingface.co/speechbrain/spkrec-ecapa-voxceleb>

5 Results

In this section, we present the outcomes of each individual module as well as the overall performance of the combined pipeline.

5.1 LLM Results

5.1.1 Vietnamese Multitask Language Understanding Benchmark

As highlighted in the related work section 2.1, the development of Vietnamese large language models (LLMs) has been a growing area of research and innovation. Models like GemSura [22], PhoGPT [24], Sailor [25], and SeaLLM [?] have all contributed to advancing the state-of-the-art in this domain.

To comprehensively evaluate the capabilities of these Vietnamese LLMs, the VMLU ³² (Vietnamese Model Language Understanding) benchmark has been introduced. VMLU is a human-centric evaluation framework consisting of 10,880 multiple-choice questions spanning 58 distinct subjects, organized across four overarching domains: STEM, Humanities, Social Sciences, and a broader category encompassing diverse topics.

The VMLU is a benchmark developed by **ZaloAI and JAIST** (Japan Advanced Institute of Science and Technology) designed to challenge foundation models across a wide spectrum of difficulty levels, from elementary proficiency to advanced professional expertise. This broad coverage not only tests the models' general knowledge but also their ability to tackle complex problem-solving tasks, providing a robust and multifaceted evaluation of their linguistic and reasoning capabilities in the Vietnamese context.

By leveraging this diverse and rigorous benchmark, researchers and developers can gain valuable insights into the strengths and limitations of their Vietnamese LLMs. This can enable them to identify areas for improvement and guide the development of more capable and versatile language models tailored to the unique needs of the Vietnamese language ecosystem.

```
{
  "id": "28-0021"
  ,
  "question": "Một nền kinh tế trong trạng thái toàn dụng nhân công có nghĩa là:"
  ,
  "choices": [
    "A. Không còn lạm phát nhưng có thể còn thất nghiệp"
    ,
    "B. Không còn thất nghiệp nhưng có thể còn lạm phát"
    ,
    "C. Không còn thất nghiệp và không còn lạm phát"
    ,
    "D. Vẫn còn một tỷ lệ lạm phát và tỷ lệ thất nghiệp nhất định"
  ]
}
```

Figure 17: An example of VMLU test set

³²VMLU: <https://vmlu.ai>

Table 5: Leaderboard of fine-tuned models on the VMLU benchmark. Our model **T-Llama** ranks in the top 5 as of the time of writing. Detail results can be found here: [Appendix A](#)

#	Model	Creator	Base Model	STEM \uparrow	SS \uparrow	HM \uparrow	OTHERS \uparrow	AVG \uparrow
1	VNPTAI.IO-14B	VNPT AI	Qwen15-14B-Chat	51.64	61.75	58.09	54.51	55.83
2	Vistral-7B-Chat	UONLP x Ontocord	Mistral-7B-v0.1	43.32	57.02	55.12	48.01	50.07
3	SealLM-7B-v2	DAMO Academy	llama-2-7b	39.95	52.02	49.38	45.27	45.79
4	bloomz-7b1	BigScience	Bloom-7b1	32.63	45.73	41.85	39.89	38.87
5	T-Llama	FPTU HCM	llama-2-7b	32.2	43.15	40.31	36.57	37.28
6	vbd-llama2-7b-50b-chat	Vin BigData	llama-2-7b	31.45	40.34	40.24	39.62	36.98
7	vietcuna-3b	Virtual Interactive	bloomz-3b	30.12	39.92	37.86	33.83	34.79
8	bloomz-1b7	BigScience	Bloom-1b7	29.72	40.17	34.73	33.41	33.65
9	SealLM-7B-Hybrid	DAMO Academy	llama-2-7b	29.49	34.61	36.68	34.52	33.39
10	ura-llama-7b	HCMCUT	llama-2-7b	29.19	33.31	34.64	32.97	32.18
11	vinallama-7b-chat	Virtual Interactive	llama-2-7b	25.70	34.50	33.87	31.41	30.64
12	vietcuna-7b-v3	Virtual Interactive	bloomz-7b	28.70	33.94	31.32	28.24	30.34
13	vietnamese-llama2-7b-40GB	BKAI - HUST	llama-2-7b	23.22	25.61	26.71	26.30	25.19

The combine method of ours base model merge [5](#) and our two phase training on a large mixed dataset of English and Vietnamese that largely came from "teacher" models like GPT-4, GPT3.5, Claude 2,... All of which contribute to this result as shown in this table here [5.1.1](#).

5.1.2 Open LLM leaderboard

The Open LLM Leaderboard is a platform that allows researchers and developers to submit their large language models for automated evaluation on a variety of benchmarks, including the Eleuther AI Language Model Evaluation Harness [\[80\]](#). This unified framework tests generative language models on a range of tasks, such as the AI2 Reasoning Challenge [\[93\]](#), HellaSwag [\[94\]](#), MMLU [\[95\]](#), TruthfulQA [\[96\]](#), Winogrande [\[97\]](#), and GSM8k [\[98\]](#).

- **AI2 Reasoning Challenge (25-shot)** [\[93\]](#): This is a set of grade-school science questions that test a model's ability to perform reasoning and inference.
- **HellaSwag (10-shot)** [\[94\]](#): This task evaluates a model's commonsense inference capabilities, which can be challenging for state-of-the-art models despite being easy for humans (around 95)
- **MMLU (5-shot)** [\[95\]](#): The Multimodal Language Understanding (MMLU) benchmark measures a text model's multitask accuracy, covering 57 tasks including elementary mathematics, US history, computer science, law, and more.
- **TruthfulQA (0-shot)** [\[96\]](#): This task evaluates a model's propensity to reproduce falsehoods commonly found online, assessing its truthfulness and reliability.
- **Winogrande (5-shot)** [\[97\]](#): This is an adversarial and difficult Winograd-style benchmark for commonsense reasoning at scale.

- **GSM8k (5-shot)** [98]: This task tests a model’s ability to solve diverse grade school math word problems, requiring multi-step mathematical reasoning.

By leveraging this leaderboard, the open-source community can transparently compare the performance of different LLMs across these key benchmarks. This helps to filter out grandiose claims and identify the genuine progress being made in the field of large language models. The leaderboard’s backend runs the Eleuther AI Language Model Evaluation Harness, ensuring a consistent and reliable evaluation process for all submitted models.

The open and collaborative nature of this platform encourages the development of more robust and capable language models, ultimately advancing the state-of-the-art in natural language processing and understanding.

Table 6: TOP-10 of the open LLM leaderboard by huggingface on 15/4/2024 only for llama2-7B backbone

Model	Average↑	ARC↑	HellaSwag↑	MMLU↑	TruthfulQA↑	Winogrande↑
kevin099/llamaRAGdrama	74.65	72.01	88.83	64.5	70.24	86.66
chargoddar/internlm2-7b-llama	66.94	60.49	88.99	63.16	54.25	79.87
kevin099/llamafia	66.49	66.13	82.08	61.81	47.94	80.11
Fantinoe5930/llama2-init-Mistral	60.98	60.07	83.3	64.09	42.15	78.37
TencentARC/LlaMA-Pro-88-Instruct	58.06	52.99	76.98	52.58	49.43	72.22
yeen214/llama2-7b-merge-orcafamily	56.38	56.91	81.17	51.49	49.68	75.93
Minami-su/Qwen1.5-7B-Chat-llamafy	56	57.59	78.52	61.18	57.59	66.46
chargoddar/internlm2-base-7b-llama	53.62	54.35	79.47	54.85	43.23	71.43
hamxea/llama-2-7b-chat-hf-activity-fine-tuned-v4	53.3	54.27	78.1	48.44	45.77	73.95
capleaf/T-Llama	54.34	54.18	76.48	47.98	46.47	71.27

As shown in the table 6, our base model merging approach 5 utilized the yeen214/llama2-7b-merge-orcafamily³³ model as the English logic and reasoning foundation. Even after merging this base with the bkai-foundation-models/vietnamese-llama2-7b-120GB [31] pre-trained Vietnamese LoRA adapter and fine-tuning on our mixed English-Vietnamese sft dataset, the model’s performance on the benchmark remained quite close to the original yeen214/llama2-7b-merge-orcafamily model. This suggests that our approach was able to maintain the strong English reasoning capabilities while also gaining the ability to understand and generate Vietnamese text.

5.2 RAG Results

5.2.1 RGB Benchmark

We follow the RAG evaluation benchmark of [99], these testbeds have almost covered almost the practical scenarios, where acquiring flawless documents containing all the required external knowledge is unattainable. There only three testbeds will be evaluated because the fourth testbed is likely always exist between LLM models and not have such assessment. Three used testbeds are:

- The resilience of large language models (LLMs) to noise in documents is referred to as noise robustness. Given the imperfections of retrieval systems, the external knowledge they gather often includes a substantial amount of irrelevant information, meaning documents that, while pertinent to the query, fail to offer any insights regarding the solution. To efficiently respond to queries, LLMs need to pinpoint the required data amidst these distractions.
- The concept of negative rejection assesses the ability of LLMs to withhold answers when the available information does not aid in resolving a query. Often, search engines do not succeed in fetching documents that contain the necessary answers. In such instances, it’s crucial for the model to recognize its limitations and refrain from providing inaccurate information.

³³https://huggingface.co/yeen214/llama2_7b_merge_orcafamily

- Information integration refers to the skill of synthesizing responses from various sources. Frequently, the solution to a query may be scattered across several documents. For instance, identifying the winners of the U.S. Open 2022 men’s and women’s singles might require piecing together information from different locations. Thus, the effectiveness of LLMs in delivering comprehensive responses to multifaceted questions hinges on their capacity for information consolidation.

5.2.2 Experiment

In total, this benchmark having compiled 600 foundational questions for RAG assessment, with an additional 200 questions aimed at evaluating the capacity for information integration and another 200 for assessing counterfactual robustness. These questions are evenly distributed between English and Chinese languages, and we want to focus only English, so that total samples for experimenting includes 300 question for Noise Robustness, Negative Rejection and Information Integration testbed

Table 7: Noise Robustness testbed. Higher noise ratio means more negative documents existing in context provided for Generator.

Noise Ratio	English (%)				
	0.0	0.2	0.4	0.6	0.8
ChatGPT ³⁴	96.33	94.67	94.00	90.00	76.00
ChatGLM-6B [100]	93.67	90.67	89.33	84.67	70.67
ChatGLM2-6B [101]	91.33	89.67	83.00	77.33	57.33
Vicuna-7B-v1.3 [102]	87.67	83.33	86.00	82.33	60.33
Qwen-7B-Chat [103]	94.33	91.67	91.00	87.67	73.67
BELLE-7B-2M [104]	83.33	81.00	79.00	71.33	64.67
T-Llama	93.67	90.67	90.00	85.67	73.33

In the Table 7, T-Llama’s performance across various noise ratios in English is listed alongside other models such as different versions of ChatGPT and other named models. For a noise ratio of 0, T-Llama shows a performance of 93.67%, which is on par with ChatGPT (OpenAI 2022) at 96.33% and is in the higher performance range among the models listed. At a noise ratio of 0.2, T-Llama’s performance drops slightly to 90.67%, which is still competitive but slightly behind the best-performing model in this category, ChatGPT (OpenAI 2022), which stands at 94.67%. The trend of a slight decrease continues for T-Llama as the noise ratio increases, with performances of 90.00%, 85.67% for noise ratios of 0.4, 0.6 respectively. This indicates a decline in performance as the noise ratio increases, which is consistent with the trends shown by other models in the table. At noise ratio of 0.8, our T-Llama result shows that we achieve 73.33% accuracy, this show our model still has drop dramatically when there exist lots of noise information. On average, we are in top 3, only behind ChatGPT and Gwen model, this superior can come from model backbone, where T-Llama is finetuned on second language, which having little shift distribution of vocabulary.

Table 8: Information Integration testbed. Rej means the rejection rate (%) and Rej* means the rejection rate evaluated by ChatGPT. Higher Rej mean this model having ability to reject the noise information

Rejection Rate \uparrow	English	
	Rej	Rej*
ChatGPT ³⁵	24.67	45.00
ChatGLM-6B [100]	9.00	25.00
ChatGLM2-6B [101]	10.33	41.33
Vicuna-7B-v1.3 [102]	17.00	33.33
Qwen-7B-Chat [103]	31.00	35.67
BELLE-7B-2M [104]	5.67	32.33
T-Llama	22.67	22.33

In the Table 8, T-Llama’s performance is shown in terms of rejection rates compared to other models when evaluated with English noise documents. T-Llama has a rejection rate (Rej) of 22.67% when only noise documents are provided, which is lower than several other models, indicating a better performance in rejecting noise. For instance, it outperforms Vicuna-7B-v1.3 and Qwen-7B-Chat, which have higher rejection rates at 17.00% and 31.00% respectively.

However, when evaluated by ChatGPT (Rej*), T-Llama’s rejection rate increases slightly to 22.33%. This rate is still quite competitive compared to the rest. For example, ChatGLM-6B has a lower Rej rate at 9.00% but a much higher Rej* rate at 25.00%, indicating a significant difference in performance when evaluated by ChatGPT. In comparison, BELLE-7B-2M shows the lowest Rej rate at 5.67%, but a Rej* rate that is notably higher at 32.33%.

Overall, T-Llama displays a consistent rejection performance, with only a slight difference between its Rej and Rej* rates, suggesting that its performance is stable regardless of whether it is self-evaluated or assessed by ChatGPT. This stability might be indicative of its reliability in rejecting noise, placing it as a robust model among the compared models, though not necessarily the top performer.

Table 9: Information Integration testbed. Higher noise ratio means more negative documents existing in context provided for Generator.

Noise Ratio	English (%)		
	0	0.2	0.4
ChatGPT ³⁶	51.0	51.0	34.0
ChatGLM-6B [100]	45.0	36.0	35.0
ChatGLM2-6B [101]	34.0	32.0	21.0
Vicuna-7B-v1.3 [102]	60.0	53.0	43.0
Qwen-7B-Chat [103]	55.0	50.0	37.0
BELLE-7B-2M [104]	40.0	34.0	24.0
T-Llama	47.0	50.0	42.0

From the table 9, we can see T-Llama’s performance compared to various models across different noise ratios in English documents, specifically focusing on the ability to integrate information. T-Llama scores 47 at a noise ratio of 0, which is lower than several other models like ChatGPT (OpenAI 2022), Qwen-7B-Chat, and Vicuna-7B-v1.3, suggesting it has less capacity for information integration in the absence of noise.

As noise increases to a ratio of 0.2, T-Llama’s score rises to 50, clearly approach ChatGPT (OpenAI 2022). This indicates a relative increase in T-Llama’s robustness to noise compared to these models. At a noise ratio of 0.4,

T-Llama’s score further increases to 42, which is notably higher than these other model except Vicuna-7B-v1.3 (Chiang et al. 2023).

Generally, T-Llama demonstrates an increase in hallucination potential with higher noise ratios, indicating a growing challenge to maintain information integration quality in noisier conditions. However, compared to other models, T-Llama shows a moderate resilience to noise, suggesting it maintains a balance in the ability to integrate information from multiple documents as noise levels increase, although it’s not the strongest in this respect.

5.3 TTS Results

- Quantitative evaluation of the trained model’s performance on various metrics.
- Comparison with baseline models and existing TTS systems.
- Qualitative assessment through audio samples demonstrating voice cloning capabilities.

5.3.1 Subjective Evaluation Results

Table 10: Subjective evaluation results for different training configurations. nMOS and sMOS scores are reported with 95% confidence intervals.

Model	nMOS \uparrow	sMOS \uparrow
Ground Truth	3.68 ± 0.11	3.70 ± 0.11
BUD500 1 epoch	2.65 ± 0.12	2.97 ± 0.12
viVoice 1 epoch	3.14 ± 0.11	3.19 ± 0.12
viVoice 5 epochs	3.29 ± 0.11	3.28 ± 0.12

The subjective evaluation results, including Mean Opinion Scores for Naturalness (nMOS) and Similarity (sMOS) with 95% confidence intervals, are presented in Table 5.3.1. The results were collected from 287 valid participants, which invalid submissions (MOS scores lower than 1) are filtered. These results offer insights into the perceived quality of the synthesized speech samples.

As expected, the ground truth samples (natural human speech) achieved the highest nMOS and sMOS scores, with average values of 3.68 and 3.70, respectively. The viVoice 5 epochs model also performed well, with an nMOS of 3.29 and an sMOS of 3.28. While a performance gap exists compared to ground truth, these scores suggest a positive listener perception of the synthesized speech.

The viVoice 1-epoch model and the Bud500 1-epoch model demonstrated lower scores, with nMOS values of 3.14 and 2.65, and sMOS values of 3.19 and 2.97, respectively. This finding suggests a potential link between utterance duration, training duration, and improved synthesis quality. Specifically, the viVoice dataset’s better length distribution gives the model the ability to generate more robust speeches than BUD500.

5.3.2 Objective Evaluation Results

Table 11: Combined results showing both objective evaluation (Word Error Rate and Character Error Rate) and speaker verification (Equal Error Rate and Speaker Encoder Cosine Similarity) for different training configurations.

Model	WER (%)↓	CER (%)↓	EER (%)↓	SECS↑
Ground Truth	5.25	2.41	-	-
BUD500 1 epoch	17.41	10.03	0.13	0.79
viVoice 1 epoch	10.98	5.46	0.13	0.78
viVoice 5 epochs	11.94	6.49	0.13	0.78

Table 11 reveals a surprising outcome. The viVoice model trained for only 1 epoch achieved a lower Word Error Rate (WER) of 10.98% and Character Error Rate (CER) of 5.46% compared to its 5-epoch counterpart. This suggests the possibility of over-fitting with longer training duration, warranting further investigation. Speaker verification results, however, showed minimal variation across checkpoints. Equal Error Rates (EER) and Speaker Encoder Cosine Similarities (SECS) remained relatively consistent, likely due to the frozen vocoder component ensuring stable speaker identity representation throughout the experiment.

5.4 Full pipeline results of LLM + RAG + TTS

5.4.1 Hardware Requirements

The implementation of the full pipeline, comprising the Large Language Model (LLM), Retrieval Augmented Generation (RAG), and Text-to-Speech (TTS) components, offers flexibility in deployment options to accommodate various infrastructural setups.

To achieve maximum speed in running the pipeline, various optimizations have been implemented. Intermediate caching techniques, such as KV caching in the Large Language Model (LLM) and audio feature caching in Text-to-Speech (TTS), have been enabled to enhance inference speed by efficiently storing and retrieving intermediate results. Additionally, efficient inference techniques, including 4-bit quantization and deepspeed, have been integrated across components to optimize inference performance. These techniques collectively enhance the speed and efficiency of the pipeline, ensuring faster processing and improved resource utilization.

For users seeking a streamlined approach, all three components can be consolidated and operated on a single machine. This integrated setup necessitates the following hardware specifications:

- **Combined Setup:**

- GPU with a minimum VRAM of 16GB
- RAM of 32GB
- Disk space: Approximately 32GB to accommodate model checkpoints and related files.

Alternatively, each component can be deployed separately via API, offering distributed deployment across multiple machines. In this scenario, the following hardware configurations are recommended:

- **Large Language Model (LLM):**

- GPU with a minimum VRAM of 8GB
- RAM of 16GB
- Disk space: At least 16GB for storing model checkpoints and associated files.

- **Retrieval Augmented Generation (RAG):**

- GPU with a minimum VRAM of 4GB
- RAM of 16GB
- Disk space: Approximately 16GB for storing RAG model checkpoints and data caches.

- **Text-to-Speech (TTS):**

- GPU with a minimum VRAM of 4GB
- RAM of 16GB
- Disk space: Around 8GB for storing TTS model files,

API deployment offers scalability and remote access, but may introduce network latency and configuration overhead. Single-machine deployment simplifies management and can improve latency-sensitive performance, but limits scalability and may lead to resource contention with computationally intensive components.

The pipeline's software components have been tested on various runtime environments (e.g., Kaggle, Google Colab, Local Ubuntu Runtime, and Docker containers) to ensure compatibility. Detailed setup instructions for installation and configuration are provided in the README file of each component's repository.

5.4.2 API Specification

The following are the abstract specifications for the Retriever, LLM, and TTS module, suitable for deployment on separate machines:

1. RAG - Upload Files from Directory

- **Endpoint:** /upload
- **Method:** POST
- **Description:** Uploads files from a specified directory to the document repository.
- **Request Body:**
 - `directory_path` (string): Path of the directory containing files to upload.
- **Response:**
 - Success: Status 200 with a success message.
 - Failure: Status code indicating the error.

2. RAG - Query Document Retriever

- **Endpoint:** /retrieve/{query_text}
- **Method:** POST
- **Description:** Queries the document retriever service with the provided text query.
- **Request Body:**
 - `query_text` (string): Text query for document retrieval.
- **Response:**
 - Success: Status 200 with retrieved documents.
 - Failure: Status code indicating the error.

3. LLM - Generate Text with T-LLama Model

- **Endpoint:** /generate
- **Method:** POST
- **Description:** Generates text using the Language Model API.
- **Request Body:**
 - **prompt** (string): The prompt for text generation.
 - **stream** (boolean, optional): Whether to stream the generated text. Defaults to false.
 - **print_stream** (boolean, optional): Whether to print the generated text as it is streamed. Defaults to false.
 - **gen_kwargs** (dictionary, optional): Additional keyword arguments for text generation.
- **Response:**
 - Success: Status 200 with generated text or API response.
 - Failure: Status code indicating the error.

4. TTS - Generate Speech with viXTTS Model

- **Endpoint:** /run_tts
- **Method:** POST
- **Description:** Generates speech using the viXTTS API.
- **Request Body:**
 - **lang** (string): The language for generation. Default: "vi".
 - **tts_text** (string): The input text.
 - **speaker_audio_file** (file/string): The reference speaker audio file. Example: speaker.wav. (Specify file format and encoding if applicable)
 - **use_deepfilter** (boolean, optional): Whether to filter the reference audio file.
 - **normalize_text** (boolean, optional): Whether to normalize the input text.
- **Response:**
 - Success: Status message "Success", and the result audio file.
 - Failure: Status code indicating the error (provide details of possible error codes and their meanings).

5.4.3 Full Demo App

The combined result of our pipeline build with Gradio [90]:

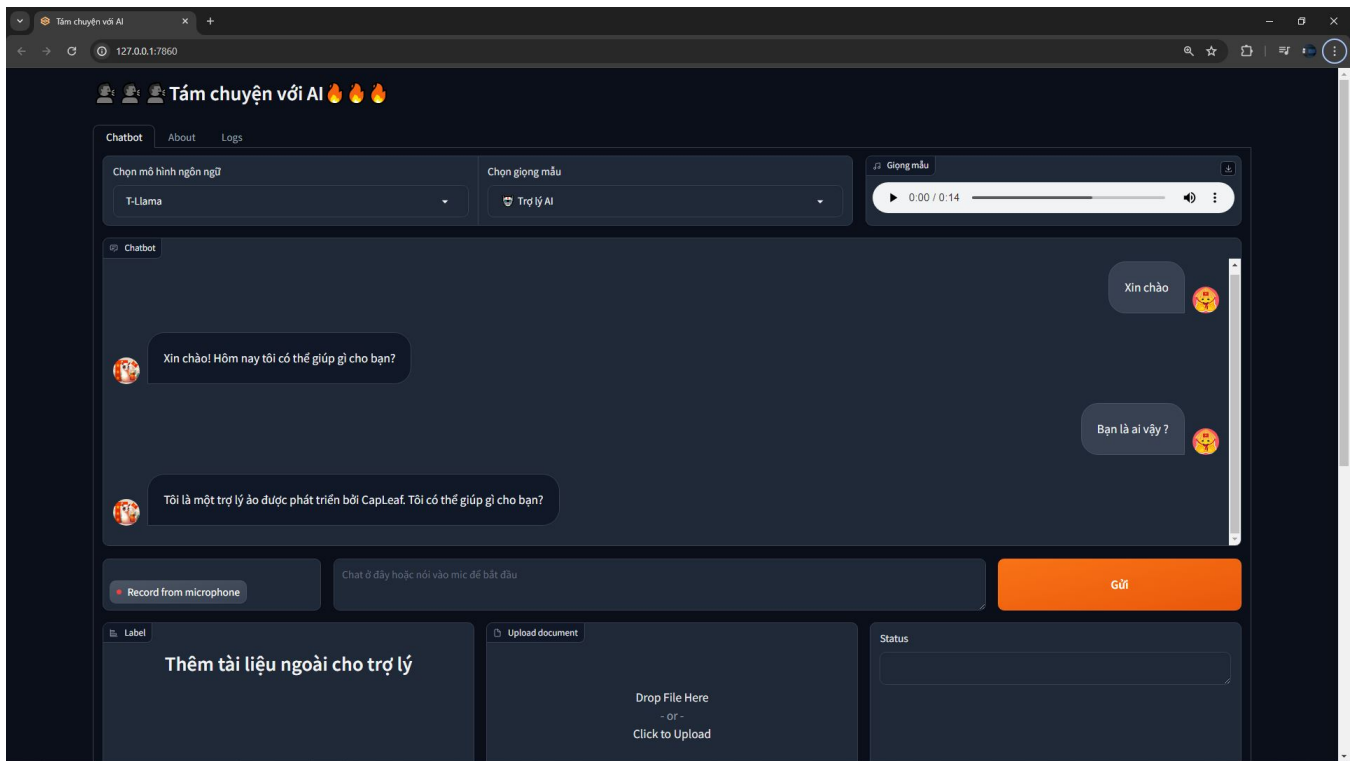


Figure 18: An example of general chat with T-Llama

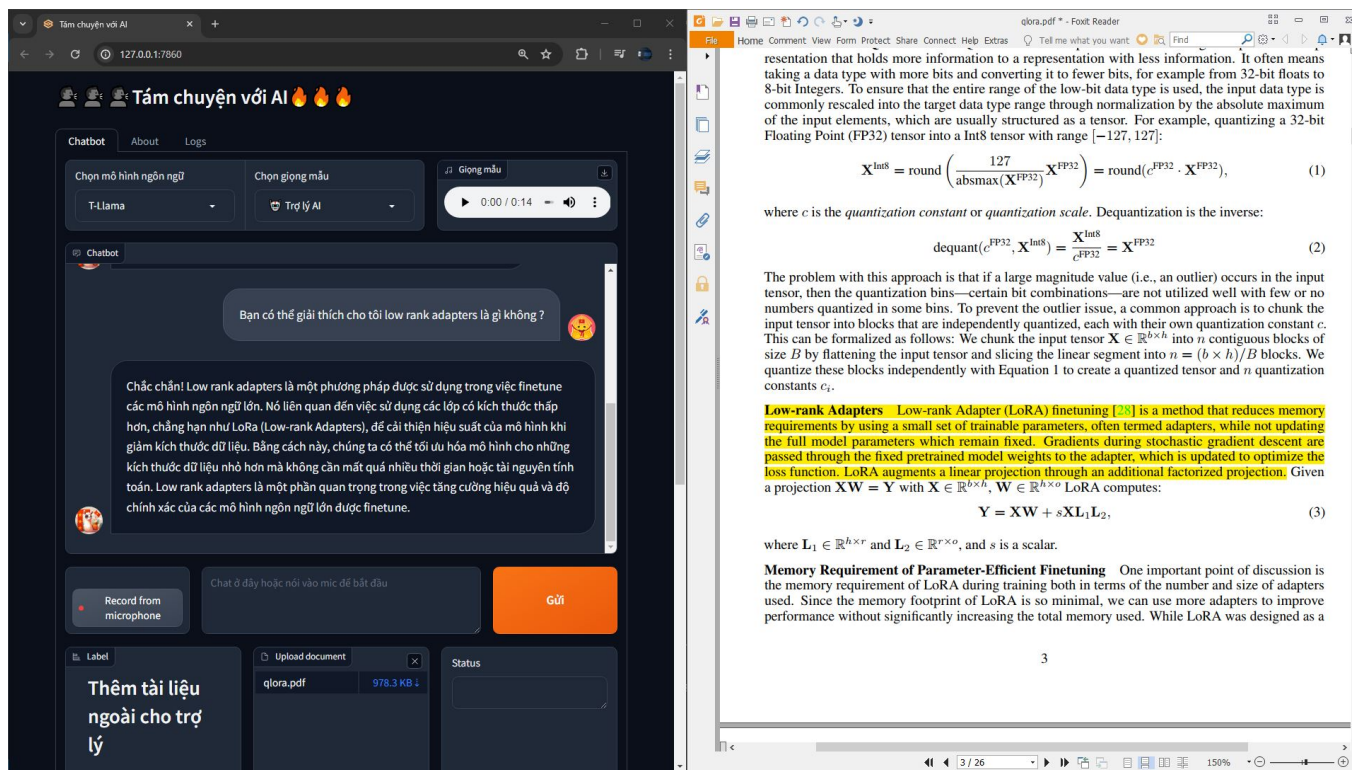


Figure 19: An example question answering on the Qlora paper asking *Bạn có thể giải thích cho tôi Low rank adapters là gì được không?*

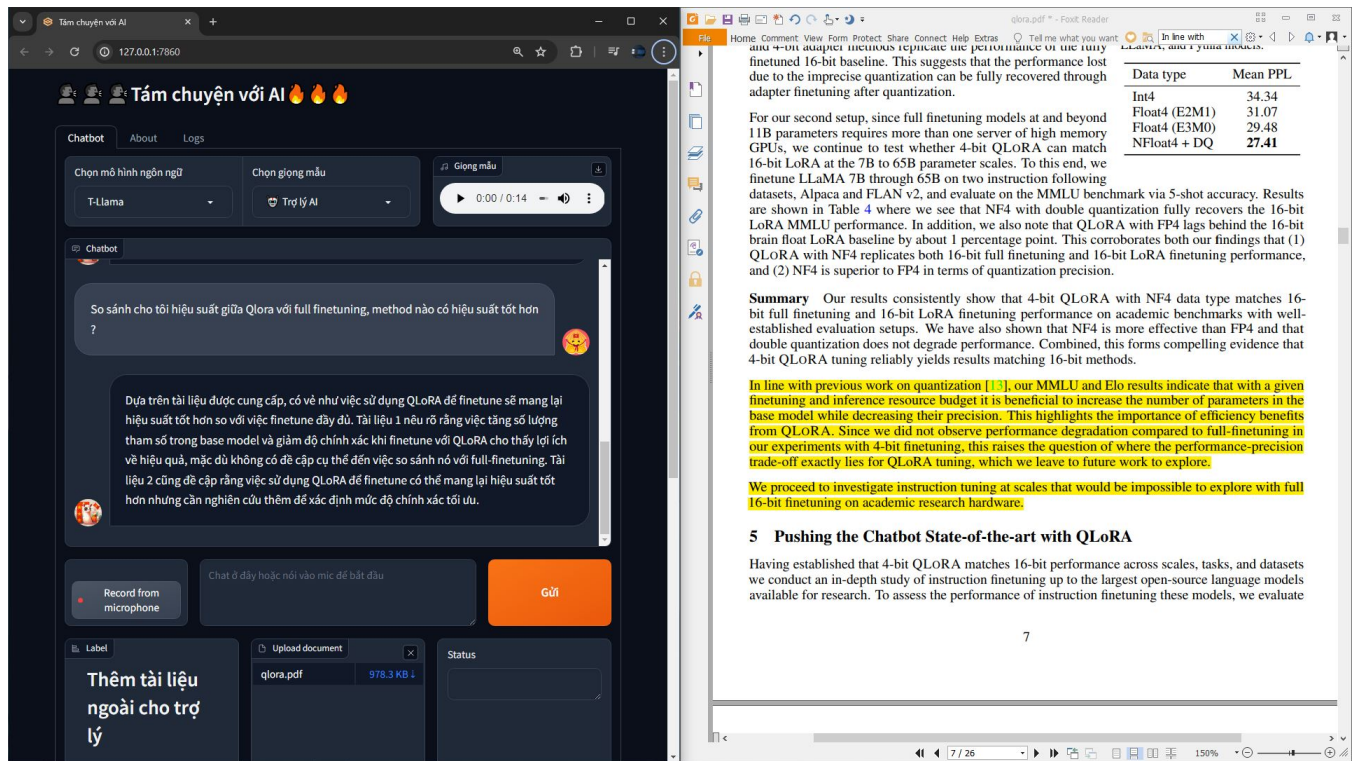


Figure 20: An example question answering on the Qlora paper asking "So sánh cho tôi hiệu suất giữa QLoRA với full fine-tuning, method nào có hiệu suất tốt hơn?". Internal prompt for LLM+RAG is shown here 28

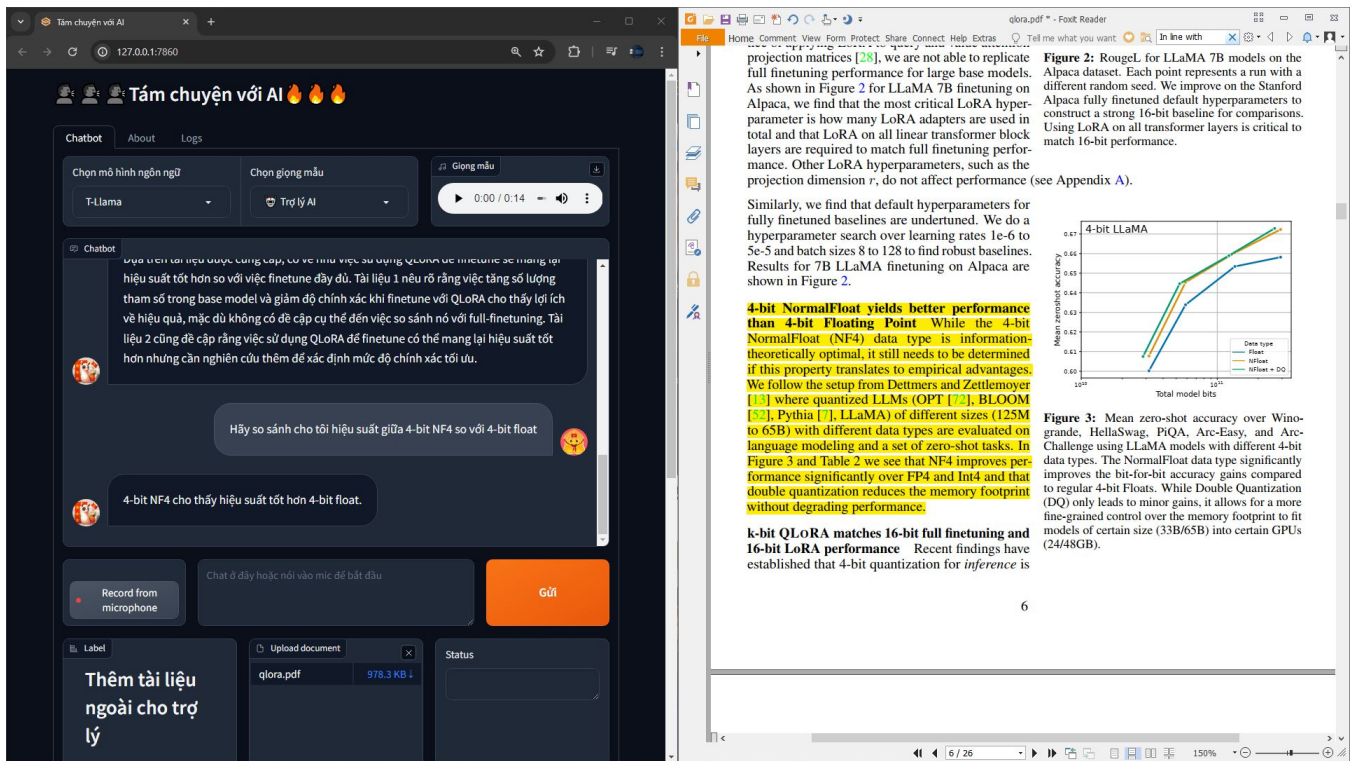


Figure 21: An example question answering on the Qlora paper asking `Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit Float` Internal prompt for LLM+RAG is shown here `27`

Here is a youtube link that include **LLM + TTS example**, cloning voice using only 6 seconds example: <https://youtu.be/NBuIR1PQwLw>

6 Discussion

6.1 Large language model merging

Model merging is a technique that combines two or more LLMs into a single model by interpolating weights together [105]. It's a relatively new and experimental method to create new models for cheap (no GPU required). Model merging works surprisingly well and produced many state-of-the-art models on the Open LLM Leaderboard [80].

However, these methods are not guaranteed and the result may vary from a significant improvement over the base model on every benchmark to catastrophic forgetting. Mixing Lora adapters however, is more "homogeneous" as the results are clearer and little to no fine-tune is necessary.

For example:

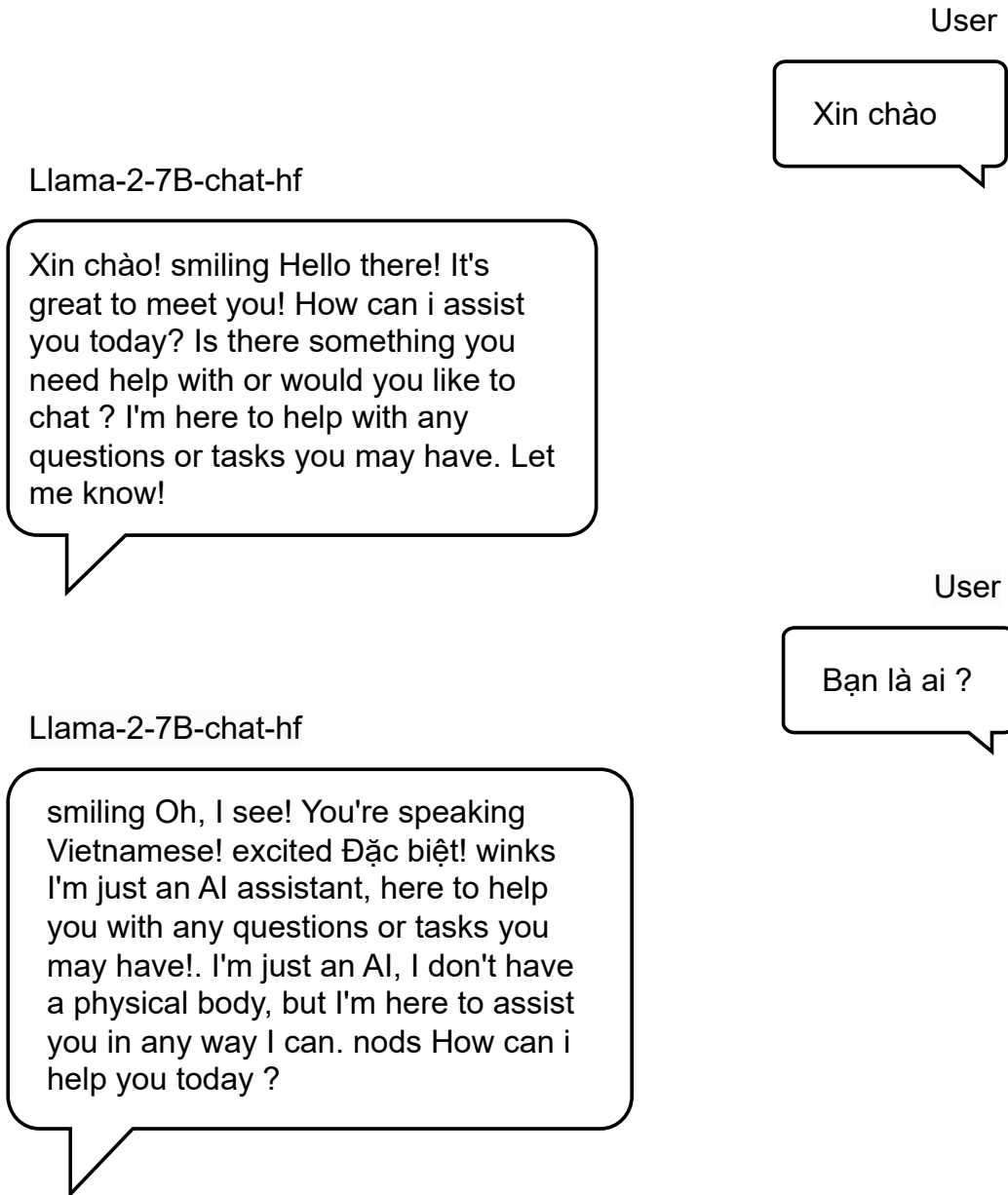


Figure 22: Here is an example chat with the sft version of llama2 from Meta

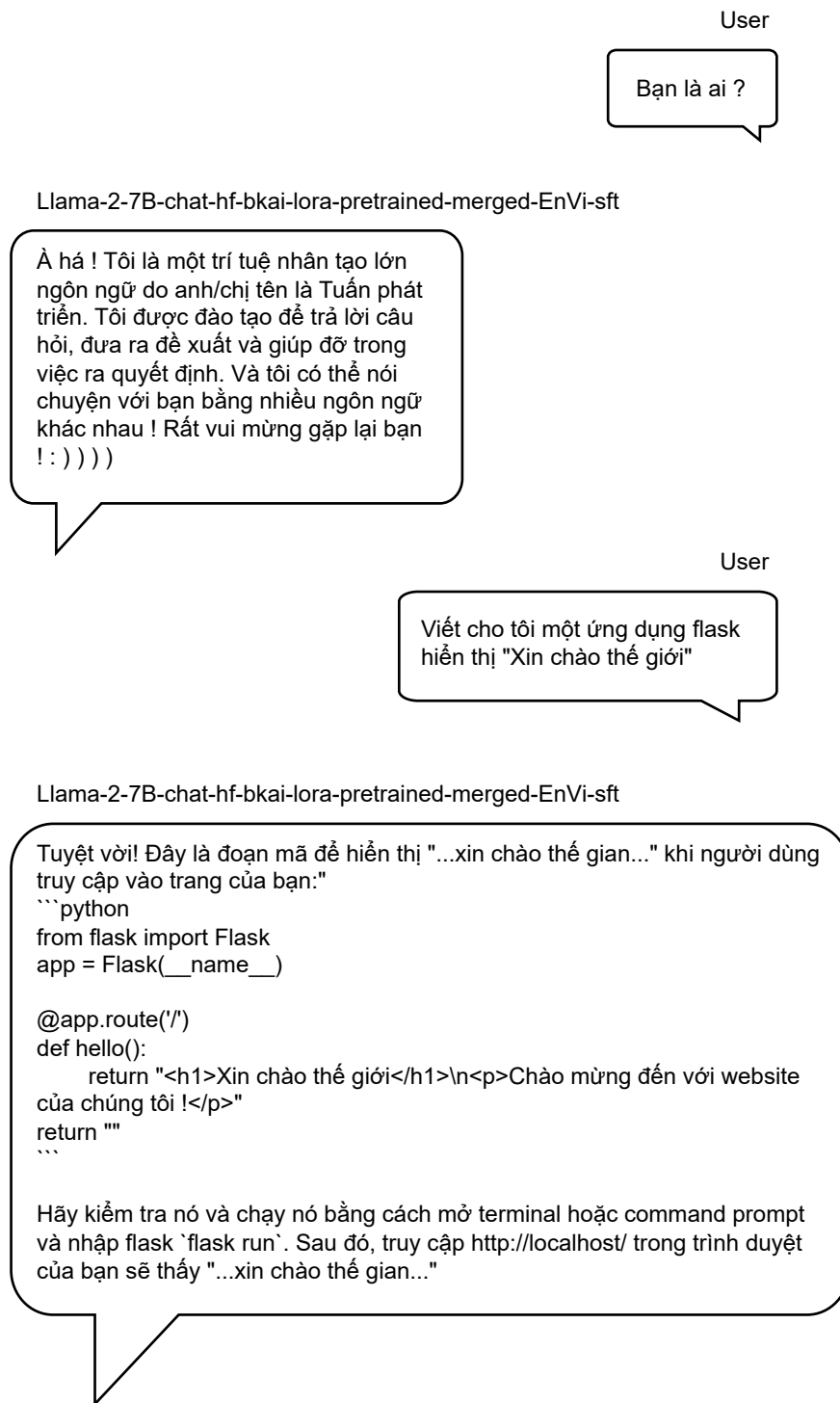


Figure 23: Here is an example chat with the model using the our procedure 5 but with meta-llama/Llama-2-7b-chat-hf instead of yeen214/llama2_7b_merge_orcafamily and then tuned on our mixed En-Vi dataset. The chat model from Meta have a very **chatty** response so we use this as an example for our hypothesis of why this method work.

The Meta LLaMA 2 model fine-tuned using supervised finetuning (SFT) techniques does not demonstrate strong

Vietnamese language capabilities, as evidenced by the example conversation shown in Figure 22. While the model's responses appear highly conversational, leveraging emojis and exhibiting a very human-like tone, its ability to effectively communicate in Vietnamese appears limited.

When we merged the Meta LLaMA-2-7B-chat model with the bkai pre-trained 120GB LORA adapter, and then fine-tuned the combined model on our dataset containing a mix of English and Vietnamese text, we observed some interesting results.

Surprisingly, the finetuned model appeared to inherit the ability to respond effectively in Vietnamese, despite the Meta LLaMA-2-7B-chat model's previous limitations in this area. Additionally, the model retained the engaging, chatty personality characterized by the use of emojis and a natural, human-like tone in its responses as shown in this Figure 23

This suggests that the combination of the pre-trained LORA adapter from bkai and the fine-tuning on the bilingual dataset were able to imbue the model with robust Vietnamese language capabilities, while preserving the conversational fluency and personality traits of the original Meta LLaMA-2-7B-chat model.

The resulting model seems to have achieved a compelling balance - it can communicate seamlessly in both English and Vietnamese, all while maintaining an approachable, chatty demeanor that can enhance the user experience. This approach demonstrates the potential for leveraging specialized adapters and fine-tuning techniques to create highly capable, versatile language models tailored to specific needs and domains.

Based on these observations, we hypothesize that the fine-tuned model not only inherited the chatty response characteristics, but also the underlying logic, reasoning capabilities, and overall intelligence of the base LLaMA-2-7B-chat model. This would suggest the robustness of the LORA adapter approach, as it was able to be successfully merged with a different base model (the LLaMA-2-7B-chat) than the one it was originally trained on (the LLaMA-2-7B base).

This method is more homogeneous than the "MergeKit [105]" approach, which involves interpolating the weights of two different models together. By leveraging the complementary strengths of the pre-trained LLM and the specialized Vietnamese adapter, we aim to imbue our base model with enhanced reasoning capabilities that can better tackle a wide range of tasks and queries in the Vietnamese language domain.

6.2 LLM are in context learners

In-context learning, also known as zero-shot or few-shot learning, is a powerful capability of modern LLMs like ChatGPT, Llama2, and Claude. It refers to the ability of these models to adapt to new tasks and domains without extensive fine-tuning on task-specific data.

The key idea behind in-context learning is that LLMs can leverage their vast general knowledge and language understanding capabilities to quickly adapt to new tasks by simply providing them with a few relevant examples or instructions within the input context. This allows the model to infer the desired behavior and generate appropriate responses, even for tasks it has not been explicitly trained on.

For example, if you provide a language model with a few examples of movie reviews, along with a prompt asking it to write a new movie review, the model can use the provided examples to understand the task and generate a relevant review, without any additional fine-tuning.

In-context learning is a significant advancement in the field of machine learning, as it reduces the need for large amounts of labeled training data and enables LLMs to be more versatile and adaptable to a wide range of applications. This has important implications for areas like few-shot learning, knowledge transfer, and rapid adaptation to new tasks and domains.

In our end-to-end pipeline, which consists of a Retrieval-Augmented Generation (RAG) model, a Large Language Model, and a Text-to-Speech (TTS) component, we employ in-context learning techniques for various submodules or "LLM functions."

One such critical function is intent classification. We've found that not all user queries require the full RAG pipeline, as many of them are simple chit-chat or factual questions that can be answered by the LLM alone, without the need for external information retrieval. Examples of such queries include "Who are you?" or "I'm bored."

To address this, we've built a prompt-based intent classification module that is integrated into the overall pipeline as shown in this figure 29 and this figure 30. This prompt is used to query our LLM, specifically our T-Llama model, to determine whether the user's input requires the activation of the RAG module or if the LLM can provide a suitable response on its own.

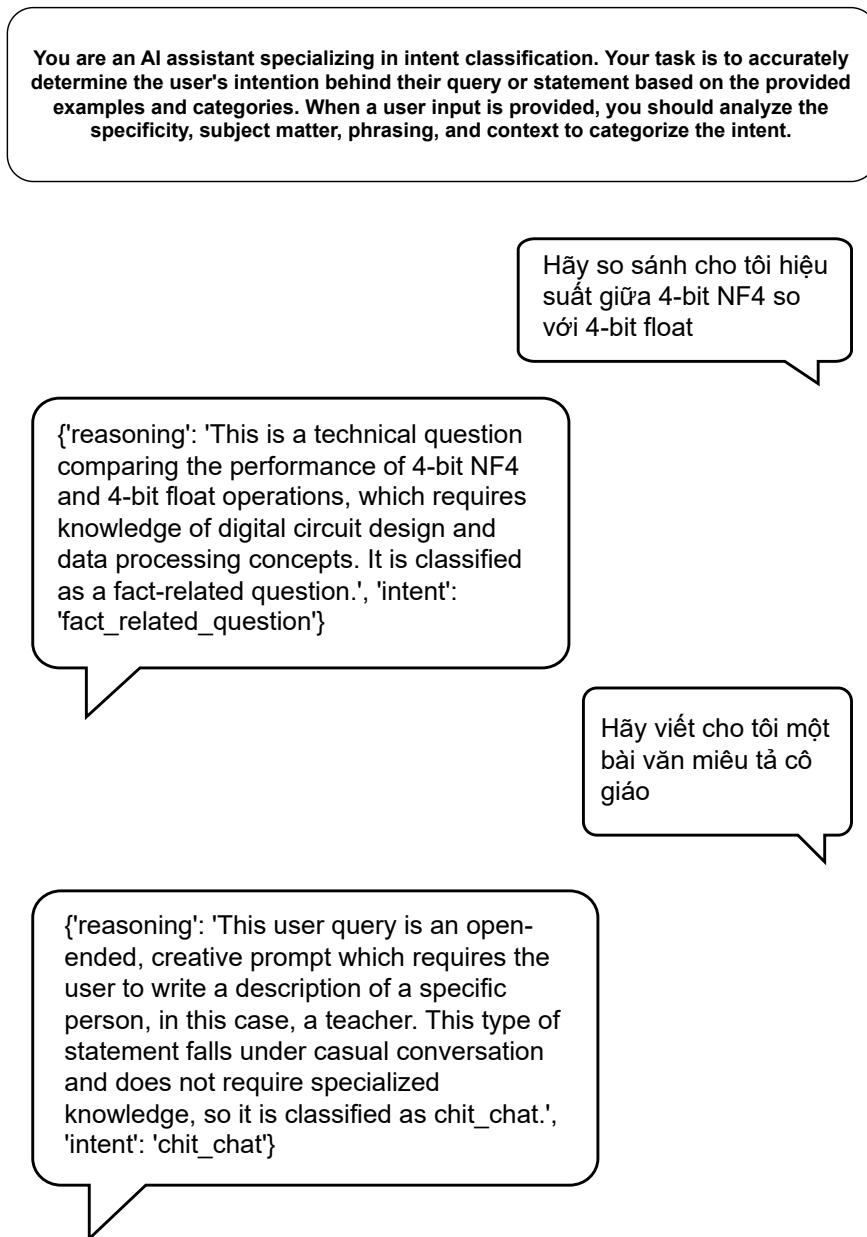


Figure 24: Intent classifier LLM function, a more detail prompt is shown in this figure here [29](#) and here [30](#)

By incorporating this intent classification step, we can optimize the computation and resource usage of our system, as we only invoke the more expensive RAG module when necessary, i.e., for queries that require external information to be answered effectively. The in-context learning capabilities of the LLM allow it to accurately classify the user's intent and make this determination, improving the overall efficiency and performance of our pipeline.

6.3 viVoice Limitations and Future Work

While this dataset has achieved prominent results with the viXTTS model, it is not without limitations. These limitations include errors in transcriptions, which have the potential to impact downstream tasks such as speech recognition and natural language processing. Additionally, the use of unnormalized text may pose challenges for

training certain models, as normalization is often essential for ensuring consistent and accurate processing of textual data.

Future research endeavors should prioritize addressing the identified limitations of the dataset to enhance its utility and reliability. This involves implementing robust error correction mechanisms to improve transcription accuracy and developing effective normalization techniques to handle unnormalized text.

7 Conclusion: A Synergistic Approach to Bilingual Interaction

This project has successfully integrated Large Language Models (LLMs), Retrieval Augmented Generation (RAG), and Text-to-Speech (TTS) to develop a robust bilingual question-answering and TTS system supporting Vietnamese and English. Key contributions include:

- The development of 'T-LLama', demonstrating strong performance on QA tasks.
- The implementation of a bilingual RAG system for efficient information retrieval.
- The creation of 'viXTTS', the first LLM-based Vietnamese TTS model with voice cloning and multilingual potential through transfer learning.
- The assembly of 'viVoice', the 1000 hours Vietnamese speech dataset,

These advancements provide a solid foundation for continued exploration and improvement in bilingual natural language processing.

LLMs, fine-tuned through instruction tuning and data augmentation techniques like LORA adapter mixing, provide the foundation for understanding and generating human-like text in both languages. The VMLU benchmark results demonstrate the effectiveness of our LLM, T-Llama, in comprehending and responding to complex queries, achieving a competitive position among state-of-the-art Vietnamese language models.

The RAG framework further enhances the system's capabilities by incorporating dynamic information retrieval. By combining sparse retrieval methods like BM25 with the semantic understanding of dense retrieval using vector databases, the RAG module ensures that the LLM has access to the most relevant and up-to-date information, enriching its responses and mitigating potential issues like hallucinations or outdated knowledge. The RGB benchmark results show T-Llama's effectiveness in handling noisy data and integrating information from various sources.

Finally, the viXTTS TTS module, trained on the novel viVoice dataset, enables natural and engaging speech synthesis in Vietnamese. This module bridges the gap between text and speech, providing a more accessible and interactive experience for users. The subjective and objective evaluation results highlight the high quality and accuracy of the synthesized speech, with promising speaker verification performance.

By synergistically integrating LLMs, RAG, and TTS, this project has paved the way for a new generation of bilingual interaction systems. The potential applications are vast, ranging from personal assistants and educational tools to customer service chatbots and accessibility solutions for the visually impaired.

Future research directions include:

- Further exploration of LLM merging techniques and data augmentation strategies to improve reasoning and task-completion abilities.
- Refinement of the RAG framework with advanced reranking and information integration methods.
- Expansion of the viVoice dataset and exploration of zero-shot voice adaptation techniques for Vietnamese TTS on this dataset.
- Development of a user-friendly interface for seamless interaction with the combined system.

This project serves as a testament to the transformative power of NLP technologies and their potential to bridge language barriers and enhance human-computer interaction. As research in these areas continues to advance, we can expect even more innovative and impactful applications that will shape the future of communication and information access.

Tài liệu

- [1] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, Q. Guo, M. Wang, and H. Wang, “Retrieval-augmented generation for large language models: A survey,” 2024.
- [2] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, “Finetuned language models are zero-shot learners,” 2022.
- [3] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, “Qlora: Efficient finetuning of quantized llms,” 2023.
- [4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” 2021.
- [5] A. Mehrish, N. Majumder, R. Bhardwaj, R. Mihalcea, and S. Poria, “A Review of Deep Learning Techniques for Speech Processing,” May 2023, arXiv:2305.00359 [eess]. [Online]. Available: <http://arxiv.org/abs/2305.00359>
- [6] S.-H. Lee, H.-Y. Choi, S.-B. Kim, and S.-W. Lee, “HierSpeech++: Bridging the Gap between Semantic and Acoustic Representation of Speech by Hierarchical Variational Inference for Zero-shot Speech Synthesis,” Nov. 2023, arXiv:2311.12454 [cs, eess]. [Online]. Available: <http://arxiv.org/abs/2311.12454>
- [7] E. Casanova, J. Weber, C. Shulby, A. C. Junior, E. Gölge, and M. A. Ponti, “YourTTS: Towards Zero-Shot Multi-Speaker TTS and Zero-Shot Voice Conversion for everyone,” Apr. 2023, arXiv:2112.02418 [cs, eess]. [Online]. Available: <http://arxiv.org/abs/2112.02418>
- [8] “XTTS.” [Online]. Available: <https://edresson.github.io/XTTS/>
- [9] H. Zen, V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, “LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech,” Apr. 2019. [Online]. Available: <https://arxiv.org/abs/1904.02882v1>
- [10] Y. Koizumi, H. Zen, S. Karita, Y. Ding, K. Yatabe, N. Morioka, M. Bacchiani, Y. Zhang, W. Han, and A. Bapna, “LibriTTS-R: A Restored Multi-Speaker Text-to-Speech Corpus,” May 2023. [Online]. Available: <https://arxiv.org/abs/2305.18802v1>
- [11] C. H. Nga, C.-T. Li, Y.-H. Li, and J.-C. Wang, “A Survey of Vietnamese Automatic Speech Recognition,” in *2021 9th International Conference on Orange Technology (ICOT)*, Dec. 2021, pp. 1–4. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9680652>
- [12] P. N. Phuong, C. T. Quang, Q. T. Do, and M. C. Luong, “A study on neural-network-based text-to-speech adaptation techniques for vietnamese,” in *2021 24th Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA)*, 2021, pp. 199–205.
- [13] T.-N. Phung, D.-B. Nguyen, and N.-P. Pham, “A Review on Speech Recognition for Under-Resourced Languages: A Case Study of Vietnamese,” *International Journal of Knowledge and Systems Science (IJKSS)*, vol. 15, no. 1, pp. 1–16, Jan. 2024, publisher: IGI Global. [Online]. Available: <https://www.igi-global.com/article/a-review-on-speech-recognition-for-under-resourced-languages/www.igi-global.com/article/a-review-on-speech-recognition-for-under-resourced-languages/332869>
- [14] H.-T. Luong and H.-Q. Vu, “A non-expert Kaldi recipe for Vietnamese speech recognition system,” in *Proceedings of the Third International Workshop on Worldwide Language Service Infrastructure and Second Workshop on Open Infrastructures and Analysis Frameworks for Human Language Technologies (WLSI/OIAF4HLT2016)*. Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 51–55. [Online]. Available: <https://aclanthology.org/W16-5207>

- [15] D. C. Tran, “The First Vietnamese FOSD-Tacotron-2-based Text-to-Speech Model Dataset,” *Data in Brief*, vol. 31, Aug. 2020, publisher: Elsevier. [Online]. Available: [https://www.data-in-brief.com/article/S2352-3409\(20\)30669-7/fulltext#](https://www.data-in-brief.com/article/S2352-3409(20)30669-7/fulltext#)
- [16] “VinBigdata shares 100-hour data for the community.” [Online]. Available: <https://institute.vinbigdata.org/en/events/vinbigdata-shares-100-hour-data-for-the-community/>
- [17] V. T. Pham, X. T. H. Nguyen, V. Hoang, and T. T. T. Nguyen, “Vietnam-Celeb: a large-scale dataset for Vietnamese speaker recognition,” in *INTERSPEECH 2023*. ISCA, Aug. 2023, pp. 1918–1922. [Online]. Available: https://www.isca-speech.org/archive/interspeech_2023/pham23b_interspeech.html
- [18] L. N. T. D. C. P. P. D. A. N. Anh Pham, Khanh Linh Tran, “Bud500: A comprehensive vietnamese asr dataset,” 2024. [Online]. Available: <https://github.com/quocanh34/Bud500>
- [19] “ntt123/VietBibleVox · Datasets at Hugging Face.” [Online]. Available: <https://huggingface.co/datasets/ntt123/VietBibleVox>
- [20] NTT123, “NTT123/Vietnamese-Text-To-Speech-Dataset,” Mar. 2024, original-date: 2020-07-21T02:09:20Z. [Online]. Available: <https://github.com/NTT123/Vietnamese-Text-To-Speech-Dataset>
- [21] Y. Koizumi, H. Zen, S. Karita, Y. Ding, K. Yatabe, N. Morioka, M. Bacchiani, Y. Zhang, W. Han, and A. Bapna, “LibriTTS-R: A Restored Multi-Speaker Text-to-Speech Corpus,” May 2023. [Online]. Available: <https://arxiv.org/abs/2305.18802v1>
- [22] S. T. Truong, D. Q. Nguyen, T. D. V. Nguyen, D. D. Le, N. N. Truong, T. Quan, and S. Koyejo, “Crossing linguistic horizons: Finetuning and comprehensive evaluation of vietnamese large language models,” in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, Washington: Association for Computational Linguistics, 2024.
- [23] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, P. Tafti, L. Hussenot, P. G. Sessa, A. Chowdhery, A. Roberts, A. Barua, A. Botev, A. Castro-Ros, A. Slone, A. Héliou, A. Tacchetti, A. Bulanova, A. Paterson, B. Tsai, B. Shahriari, C. L. Lan, C. A. Choquette-Choo, C. Crepy, D. Cer, D. Ippolito, D. Reid, E. Buchatskaya, E. Ni, E. Noland, G. Yan, G. Tucker, G.-C. Muraru, G. Rozhdestvenskiy, H. Michalewski, I. Tenney, I. Grishchenko, J. Austin, J. Keeling, J. Labanowski, J.-B. Lespiau, J. Stanway, J. Brennan, J. Chen, J. Ferret, J. Chiu, J. Mao-Jones, K. Lee, K. Yu, K. Millican, L. L. Sjoesund, L. Lee, L. Dixon, M. Reid, M. Miłkula, M. Wirth, M. Sharman, N. Chinaev, N. Thain, O. Bachem, O. Chang, O. Wahltinez, P. Bailey, P. Michel, P. Yotov, R. Chaabouni, R. Comanescu, R. Jana, R. Anil, R. McIlroy, R. Liu, R. Mullins, S. L. Smith, S. Borgeaud, S. Girgin, S. Douglas, S. Pandya, S. Shakeri, S. De, T. Klimenko, T. Hennigan, V. Feinberg, W. Stokowiec, Y. hui Chen, Z. Ahmed, Z. Gong, T. Warkentin, L. Peran, M. Giang, C. Farabet, O. Vinyals, J. Dean, K. Kavukcuoglu, D. Hassabis, Z. Ghahramani, D. Eck, J. Barral, F. Pereira, E. Collins, A. Joulin, N. Fiedel, E. Senter, A. Andreev, and K. Kenealy, “Gemma: Open models based on gemini research and technology,” 2024.
- [24] D. Q. Nguyen, L. T. Nguyen, C. Tran, D. N. Nguyen, D. Phung, and H. Bui, “PhoGPT: Generative Pre-training for Vietnamese,” *arXiv preprint*, vol. arXiv:2311.02945, 2023.
- [25] L. Dou, Q. Liu, G. Zeng, J. Guo, J. Zhou, W. Lu, and M. Lin, “Sailor: Open language models for south-east asia,” 2024.
- [26] X. L. M. A. Z. H. C. S. Y. K. C. X. L. J. W. Q. T. L. C. G. C. Y. D. S. Y. C. L. H. Z. L. B. Xuan-Phi Nguyen*, Wenxuan Zhang*, “Seallms - large language models for southeast asia,” 2023.
- [27] H. L. Duy Quang Do and D. T. Nguyen, “Torolama: The vietnamese instruction-following and chat model,” Software, 2023, <https://github.com/allbyai/ToRoLaMa>.
- [28] Q. N. H. N. B. P. N. P. H. N. P. S. T. N. Chien Van Nguyen, Thuat Nguyen, “Vistral-7b-chat - towards a state-of-the-art large language model for vietnamese,” 2023.

- [29] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, “Mistral 7b,” 2023.
- [30] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, “Llama 2: Open foundation and fine-tuned chat models,” 2023.
- [31] N. D. N. N. D. N. M. L. T. H. D. V. S. Nguyen Quang Duc, Le Hai Son, “Towards comprehensive vietnamese retrieval-augmented generation and large language models,” *arXiv preprint arXiv:2403.01616*, 2024.
- [32] T. Nguyen, C. V. Nguyen, V. D. Lai, H. Man, N. T. Ngo, F. Dernoncourt, R. A. Rossi, and T. H. Nguyen, “Culturax: A cleaned, enormous, and multilingual dataset for large language models in 167 languages,” 2023.
- [33] H. Li, F. Koto, M. Wu, A. F. Aji, and T. Baldwin, “Bactrian-x : A multilingual replicable instruction-following model with low-rank adaptation,” 2023.
- [34] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, “Query rewriting for retrieval-augmented large language models,” 2023.
- [35] X. Li, Z. Liu, C. Xiong, S. Yu, Y. Gu, Z. Liu, and G. Yu, “Structure-aware language model pretraining improves dense retrieval on structured data,” 2023.
- [36] X. V. Lin, X. Chen, M. Chen, W. Shi, M. Lomeli, R. James, P. Rodriguez, J. Kahn, G. Szilvasy, M. Lewis, L. Zettlemoyer, and S. Yih, “Ra-dit: Retrieval-augmented dual instruction tuning,” *ArXiv*, vol. abs/2310.01352, 2023. [Online]. Available: <https://api.semanticscholar.org/CorpusID:263605962>
- [37] W. Yu, D. Iter, S. Wang, Y. Xu, M. Ju, S. Sanyal, C. Zhu, M. Zeng, and M. Jiang, “Generate rather than retrieve: Large language models are strong context generators,” 2023.
- [38] Z. Shao, Y. Gong, Y. Shen, M. Huang, N. Duan, and W. Chen, “Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy,” 2023.
- [39] X. Wang, Q. Yang, Y. Qiu, J. Liang, Q. He, Z. Gu, Y. Xiao, and W. Wang, “Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases,” 2023.
- [40] Z. Rackauckas, “Rag-fusion: A new take on retrieval augmented generation,” *International Journal on Natural Language Computing*, vol. 13, no. 1, p. 37–47, Feb. 2024. [Online]. Available: <http://dx.doi.org/10.5121/ijnlc.2024.13103>
- [41] X. Cheng, D. Luo, X. Chen, L. Liu, D. Zhao, and R. Yan, “Lift yourself up: Retrieval-augmented text generation with self memory,” 2023.
- [42] X. Li, E. Nie, and S. Liang, “From classification to generation: Insights into crosslingual retrieval augmented icl,” 2023.
- [43] D. Cheng, S. Huang, J. Bi, Y. Zhan, J. Liu, Y. Wang, H. Sun, F. Wei, D. Deng, and Q. Zhang, “Uprise: Universal prompt retrieval for improving zero-shot evaluation,” 2023.
- [44] Z. Dai, V. Y. Zhao, J. Ma, Y. Luan, J. Ni, J. Lu, A. Bakalov, K. Guu, K. B. Hall, and M.-W. Chang, “Promptagator: Few-shot dense retrieval from 8 examples,” 2022.
- [45] Z. Sun, X. Wang, Y. Tay, Y. Yang, and D. Zhou, “Recitation-augmented language models,” 2023.

- [46] O. Khattab, K. Santhanam, X. L. Li, D. Hall, P. Liang, C. Potts, and M. Zaharia, “Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp,” 2023.
- [47] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, “Active retrieval augmented generation,” 2023.
- [48] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, “Self-rag: Learning to retrieve, generate, and critique through self-reflection,” 2023.
- [49] ICT Vietnam. (2020) Ra mắt 2 nền tảng xử lý giọng nói tiếng việt, ứng dụng ai. [Online]. Available: <https://ictvietnam.vn/ra-mat-2-nen-tang-xu-ly-giong-noi-tieng-viet-ung-dung-ai-32373.html>
- [50] —. (2022) Vinfast bắt tay amazon đưa trợ lý giọng nói alexa vào xe điện. [Online]. Available: <https://ictvietnam.vn/vinfast-bat-tay-amazon-dua-tro-ly-giong-noi-alexa-va-vo-xe-dien-21331.html>
- [51] —. (2023) Trợ lý giọng nói kiki cán mốc 300.000 lượt cài đặt sử dụng. [Online]. Available: <https://ictvietnam.vn/tro-ly-giong-noi-kiki-can-moc-300-000-luot-cai-dat-su-dung-56821.html>
- [52] —. (2022) Sẽ ứng dụng callbot trong việc nhắc hạn đăng kiểm và bhxh. [Online]. Available: <https://ictvietnam.vn/se-ung-dung-callbot-trong-viec-nhac-han-dang-kiem-va-bhxh-22230.html>
- [53] —. (2020) Fpt ai ra mắt ứng dụng chuyển đổi văn bản thành giọng nói. [Online]. Available: <https://ictvietnam.vn/fpt-ai-ra-mat-ung-dung-chuyen-doi-van-ban-thanh-giong-noi-30095.html>
- [54] —. (2020) Giải pháp loa phương thông minh đạt giải vàng quốc tế. [Online]. Available: <https://ictvietnam.vn/giai-phap-loa-phuong-thong-minh-dat-giai-vang-quoc-te-31429.html>
- [55] L. H. Thạch. (2021) Tạo thói quen nghe “sách” cho người việt - lê hoàng thạch - ceo voiz fm | vi ep03. Youtube. [Online]. Available: <https://youtu.be/4OR4Rkg0cRc?feature=shared&t=1806>
- [56] N. T. T. Trang and N. X. Tung, “Text-to-speech shared task in vlsp campaign 2019: evaluating vietnamese speech synthesis on common datasets,” *Vietnamese Language Signal Processing. VLSP*, 2019.
- [57] T. T. T. Nguyen, H. K. Nguyen, Q. M. Pham, and D. M. Vu, “Vietnamese text-to-speech shared task vlsp 2020: Remaining problems with state-of-the-art techniques,” in *Proceedings of the 7th International Workshop on Vietnamese Language and Speech Processing*, 2020, pp. 35–39.
- [58] N. T. T. Trang and H. K. Nguyen, “Vlsp 2021-tts challenge: Vietnamese spontaneous speech synthesis,” *VNU Journal of Science: Computer Science and Communication Engineering*, vol. 38, no. 1, 2022.
- [59] “VLSP 2022 TTS: Emotional speech synthesis | Association for Vietnamese Language and Speech Processing.” [Online]. Available: <https://vlsp.org.vn/vlsp2022/eval/tts>
- [60] VLSP. (2023) Vlsp 2023 challenge on emotional speech synthesis. [Online]. Available: <https://vlsp.org.vn/vlsp2023/eval/ess>
- [61] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” 2018.
- [62] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, “FastSpeech 2: Fast and high-quality end-to-end text to speech,” 2022.
- [63] J. Kim, J. Kong, and J. Son, “Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech,” 2021.
- [64] P. P. Ngoc, C. T. Quang, and M. L. Chi, “ADAPT-TTS: HIGH-QUALITY ZERO-SHOT MULTI-SPEAKER TEXT-TO-SPEECH ADAPTIVE-BASED FOR VIETNAMESE,” *Journal of Computer Science and Cybernetics*, vol. 39, no. 2, pp. 159–173, Jun. 2023, number: 2. [Online]. Available: <https://vjs.ac.vn/index.php/jcc/article/view/18136>

- [65] C. Tran, C. M. Luong, and S. Sakti, “STEN-TTS: Improving Zero-shot Cross-Lingual Transfer for Multi-Lingual TTS with Style-Enhanced Normalization Diffusion Framework,” in *Proc. INTERSPEECH 2023*, 2023, pp. 4464–4468.
- [66] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, “Common voice: A massively-multilingual speech corpus,” in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215.
- [67] MICA: Multimedia, Information, Communication Applications, “Vnspeechcorpus,” https://www.mica.edu.vn/index.php?option=com_content&view=article&id=25&Itemid=226&lang=vn, Accessed: 2024.
- [68] V. T. Pham, X. T. H. Nguyen, V. Hoang, and T. T. T. Nguyen, “Vietnam-Celeb: a large-scale dataset for Vietnamese speaker recognition,” in *INTERSPEECH 2023*. ISCA, Aug. 2023, pp. 1918–1922. [Online]. Available: https://www.isca-archive.org/interspeech_2023/pham23b_interspeech.html
- [69] M. Conover, M. Hayes, A. Mathur, J. Xie, J. Wan, S. Shah, A. Ghodsi, P. Wendell, M. Zaharia, and R. Xin. (2023) Free dolly: Introducing the world’s first truly open instruction-tuned llm. [Online]. Available: <https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm>
- [70] Da Nang, Vietnam.
- [71] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, “Stanford alpaca: An instruction-following llama model,” https://github.com/tatsu-lab/stanford_alpaca, 2023.
- [72] W. Lian, B. Goodson, E. Pentland, A. Cook, C. Vong, and "Teknium", “Openorca: An open dataset of gpt augmented flan reasoning traces,” <https://huggingface.co/Open-Orca/OpenOrca>, 2023.
- [73] G. Z. Y. F. W. H. H. S. Y. S. W. C. Xiang Yue, Xingwei Qu, “Mammoth: Building math generalist models through hybrid instruction tuning,” *arXiv preprint arXiv:2309.05653*, 2023.
- [74] S. Mangrulkar, S. Gugger, L. Debut, Y. Belkada, S. Paul, and B. Bossan, “Peft: State-of-the-art parameter-efficient fine-tuning methods,” <https://github.com/huggingface/peft>, 2022.
- [75] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *International Conference on Learning Representations*, 2022. [Online]. Available: <https://openreview.net/forum?id=nZeVKeeFYf9>
- [76] A. N. Lee, C. J. Hunter, and N. Ruiz, “Platypus: Quick, cheap, and powerful refinement of llms,” 2024.
- [77] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” 2019.
- [78] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, “Huggingface’s transformers: State-of-the-art natural language processing,” 2020.
- [79] S. Gugger, L. Debut, T. Wolf, P. Schmid, Z. Mueller, S. Mangrulkar, M. Sun, and B. Bossan, “Accelerate: Training and inference at scale made simple, efficient and adaptable.” <https://github.com/huggingface/accelerate>, 2022.
- [80] E. Beeching, C. Fourier, N. Habib, S. Han, N. Lambert, N. Rajani, O. Sanseviero, L. Tunstall, and T. Wolf, “Open llm leaderboard,” https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.
- [81] R. Solovyev, A. Stempkovskiy, and T. Habruseva, “Benchmarks and leaderboards for sound demixing tasks,” 2023.
- [82] S. Tomar, “Converting video formats with ffmpeg,” *Linux Journal*, vol. 2006, no. 146, p. 10, 2006.

- [83] H. Schröter, T. Rosenkranz, A. N. Escalante-B., and A. Maier, “DeepFilterNet: Perceptually Motivated Real-Time Speech Enhancement,” May 2023, arXiv:2305.08227 [cs, eess]. [Online]. Available: <http://arxiv.org/abs/2305.08227>
- [84] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, “Robust speech recognition via large-scale weak supervision,” 2022.
- [85] T.-T. Le, L. T. Nguyen, and D. Q. Nguyen, “PhoWhisper: Automatic Speech Recognition for Vietnamese,” in *Proceedings of the ICLR 2024 Tiny Papers track*, 2024.
- [86] D. Q. Nguyen, L. T. Nguyen, C. Tran, D. N. Nguyen, N. Nguyen, T. H. Nguyen, D. Phung, and H. Bui, “Phogpt: Generative pre-training for vietnamese,” *arXiv preprint arXiv:2311.02945*, 2023.
- [87] A. Rosenberg and B. Ramabhadran, “Bias and statistical significance in evaluating speech synthesis with mean opinion scores.” in *Interspeech*, 2017, pp. 3976–3980.
- [88] S. Le Maguer, S. King, and N. Harte, “The limits of the mean opinion score for speech synthesis evaluation,” *Computer Speech & Language*, vol. 84, p. 101577, 2024.
- [89] Y. A. Li, C. Han, V. S. Raghavan, G. Mischler, and N. Mesgarani, “StyleTTS 2: Towards Human-Level Text-to-Speech through Style Diffusion and Adversarial Training with Large Speech Language Models,” Nov. 2023, arXiv:2306.07691 [cs, eess]. [Online]. Available: <http://arxiv.org/abs/2306.07691>
- [90] A. Abid, A. Abdalla, A. Abid, D. Khan, A. Alfozan, and J. Zou, “Gradio: Hassle-free sharing and testing of ml models in the wild,” *arXiv preprint arXiv:1906.02569*, 2019.
- [91] B. Desplanques, J. Thienpondt, and K. Demuyck, “Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnm based speaker verification,” in *Interspeech 2020*. ISCA, Oct. 2020. [Online]. Available: <http://dx.doi.org/10.21437/Interspeech.2020-2650>
- [92] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, “SpeechBrain: A general-purpose speech toolkit,” 2021, arXiv:2106.04624.
- [93] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, “Think you have solved question answering? try arc, the ai2 reasoning challenge,” 2018.
- [94] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, “Hellaswag: Can a machine really finish your sentence?” 2019.
- [95] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, “Measuring massive multitask language understanding,” 2021.
- [96] S. Lin, J. Hilton, and O. Evans, “Truthfulqa: Measuring how models mimic human falsehoods,” 2022.
- [97] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, “WINOGRANDE: an adversarial winograd schema challenge at scale,” 2019.
- [98] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, “Training verifiers to solve math word problems,” 2021.
- [99] J. Chen, H. Lin, X. Han, and L. Sun, “Benchmarking large language models in retrieval-augmented generation,” 2023.
- [100] Z. Du, Y. Qian, X. Liu, M. Ding, J. Qiu, Z. Yang, and J. Tang, “Glm: General language model pretraining with autoregressive blank infilling,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 320–335.

- [101] A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia *et al.*, “Glm-130b: An open bilingual pre-trained model,” *arXiv preprint arXiv:2210.02414*, 2022.
- [102] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing, “Vicuna: An open-source chatbot impressing gpt-4 with 90% chatgpt quality,” March 2023. [Online]. Available: <https://lmsys.org/blog/2023-03-30-vicuna>
- [103] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, B. Hui, L. Ji, M. Li, J. Lin, R. Lin, D. Liu, G. Liu, C. Lu, K. Lu, J. Ma, R. Men, X. Ren, X. Ren, C. Tan, S. Tan, J. Tu, P. Wang, S. Wang, W. Wang, S. Wu, B. Xu, J. Xu, A. Yang, H. Yang, J. Yang, S. Yang, Y. Yao, B. Yu, H. Yuan, Z. Yuan, J. Zhang, X. Zhang, Y. Zhang, Z. Zhang, C. Zhou, J. Zhou, X. Zhou, and T. Zhu, “Qwen technical report,” *arXiv preprint arXiv:2309.16609*, 2023.
- [104] Y. G. Y. P. Q. N. L. Z. B. M. X. L. Yunjie Ji, Yong Deng, “Exploring the impact of instruction data scaling on large language models: An empirical study on real-world use cases,” *arXiv preprint arXiv:2303.14742*, 2023.
- [105] C. Goddard, S. Siriwardhana, M. Ehghaghi, L. Meyers, V. Karpukhin, B. Benedict, M. McQuade, and J. Solawetz, “Arcee’s mergekit: A toolkit for merging large language models,” 2024.

A Detail of Vietnamese Multitask Language Understanding Benchmark for T-Llama

Table 12: Summary of T-Llama scores by Category

Category	Subject	Score (%)
STEM		
	Introduction to Programming	25.7
	Metrology Engineer	22.7
	Middle School Biology	38.82
	Middle School Chemistry	31.11
	Middle School Mathematics	27.78
	Middle School Physics	24.44
	Operating System	41.67
	Statistics and Probability	26.44
	Applied Informatics	45.56
	Computer Architecture	34.44
	Computer Network	39.11
	Discrete Mathematics	32.73
	Electrical Engineering	26.14
	Elementary Mathematics	28.33
	Elementary Science	55.56
	High School Biology	31.67
	High School Chemistry	25.56
	High School Mathematics	23.65
	High School Physics	34.44
	Introduction to Chemistry	29.61
	Introduction to Physics	30.64
	Total	32.2
Humanity		
	Administrative Law	45.56
	Business Law	41.9
	Civil Law	49.44
	Criminal Law	42.33
	Economic Law	26.71
	Education Law	37.95
	Elementary History	38.98
	High School History	37.22
	High School Literature	37.78
	History of World Civilization	49.44
	Ideological and Moral Cultivation	47.78
	Introduction to Laws	47.62
	Introduction to Vietnam Culture	37.22
	Logic	30.46

Continued on next page

Table 12 – *Continued from previous page*

Category	Subject	Score (%)
	Middle School History	42.22
	Middle School Literature	40.8
	Revolutionary Policy of the Vietnamese Communist Party	40
	Vietnamese Language and Literature	32.18
	Total	40.31
Social Science		
	Principles of Marxism and Leninism	40
	Sociology	50.56
	Business Administration	43.1
	High School Civil Education	48.33
	High School Geography	33.33
	Ho Chi Minh Ideology	37.99
	Macroeconomics	30
	Microeconomics	38.89
	Middle School Civil Education	65.73
	Middle School Geography	43.54
	Total	43.15
Other		
	Clinical Pharmacology	26.67
	Driving License Certificate	54.97
	Environmental Engineering	30.99
	Internal Basic Medicine	29.24
	Preschool Pedagogy	48.04
	Tax Accountant	30.46
	Tax Civil Servant	40.94
	Accountant	32.14
	Civil Servant	35.67
	Total	36.57

B Large language model dataset format

Translated example

```
{
"system_prompt": "Your purpose is to inform, provide a comprehensive response.",
"qas_id": "N7X736",
"question_text": "Chuẩn bị thực đơn ngày mai cho người muốn ăn uống lành mạnh và có ngân sách eo hẹp",
"orig_answer_texts": "Đây là kế hoạch bữa ăn dành cho những người muốn ăn uống lành mạnh với ngân sách hạn chế.\n\n**Bữa sáng:** Yến mạch để qua đêm\n- Nguyên liệu: 1/3 cốc yến mạch cán, 1/2 cốc sữa hạnh nhân, 1/2 cốc sữa chua Hy Lạp, 1/2 cốc quả mọng và một chút mật ong.\n- Hướng dẫn: Trộn yến mạch, sữa hạnh nhân và sữa chua Hy Lạp vào lọ hoặc bát. Khuấy để kết hợp. Top với quả mọng và mật ong. Đậy nắp và để lạnh qua đêm.\n\n**Ăn nhẹ:** Trứng luộc\n- Nguyên liệu: 1 quả trứng lớn\n- Hướng dẫn: Cho trứng vào nồi và tráng qua nước lạnh. Đun sôi nước ở nhiệt độ cao. Sau khi sôi, lấy chảo ra khỏi bếp, đậy nắp và để yên trong 12 phút. Sau 12 phút, vớt trứng ra tô nước đá cho nguội.\n\n**Bữa trưa:** Cơm gạo lứt và rau xào\n- Nguyên liệu: 1 chén gạo lứt nấu chín, 1 chén rau củ đông lạnh trộn, 1 thìa dầu ô liu và nước tương vừa ăn.\n- Hướng dẫn: Đun nóng dầu ô liu trong chảo trên lửa vừa. Thêm rau đông lạnh và nấu cho đến khi mềm, khoảng 5 phút. Cho cơm đã nấu vào chảo và nấu thêm 2-3 phút, khuấy thường xuyên. Rưới nước tương và phục vụ.\n\n**Ăn nhẹ:** Chuối bơ đậu phộng\n- Nguyên liệu: 1 quả chuối vừa và 1 thìa bơ đậu phộng tự nhiên\n- Hướng dẫn: Cắt lát chuối và phết bơ đậu phộng lên trên.\n\n**Bữa tối:** Đùi gà nướng với rau củ nướng\n- Nguyên liệu: 2 đùi gà, 2 chén rau trộn (như cà rốt, bông cải xanh, súp lơ trắng), 1 thìa dầu ô liu, muối và tiêu.\n- Hướng dẫn: Làm nóng lò ở nhiệt độ 425°F (220°C). Đặt đùi gà và rau củ lên khay nướng. Rắc dầu ô liu và nêm muối và hạt tiêu. Nướng khoảng 30 phút cho đến khi gà chín và rau củ mềm.\n\n**Món tráng miệng:** Salad trái cây tươi\n- Nguyên liệu: 1 cốc trái cây tươi trộn (ví dụ: quả mọng, táo, chuối)\n- Hướng dẫn: Rửa sạch và cắt trái cây thành từng miếng nhỏ. Trộn đều trong một cái bát và phục vụ.\n\nHãy tận hưởng kế hoạch bữa ăn lành mạnh và hợp túi tiền của bạn cho ngày mai!",
"answer_lengths": null
}
```

Parsed example

```
{
"system_prompt": "Your purpose is to inform, provide a comprehensive response.",
"qas_id": "N7X736",
"question_text": "Prepare a meal plan for tomorrow for someone who wants to eat healthy and has a limited budget",
"orig_answer_texts": "Here is a meal plan for someone who wants to eat healthy on a limited budget.\n\n**Breakfast:** Overnight oats\n- Ingredients: 1/3 cup rolled oats, 1/2 cup almond milk, 1/2 cup Greek yogurt, 1/2 cup berries, and a drizzle of honey.\n- Instructions: Combine the oats, almond milk, and Greek yogurt in a jar or bowl. Stir to combine. Top with berries and honey. Cover and refrigerate overnight.\n\n**Snack:** Hard-boiled Egg\n- Ingredients: 1 large egg\n- Instructions: Place the egg in a saucepan and cover it with cold water. Bring the water to a boil over high heat. Once boiling, remove the pan from heat, cover, and allow to sit for 12 minutes. After 12 minutes, transfer the egg to a bowl of ice water to cool.\n\n**Lunch:** Brown Rice and Vegetable Stir Fry\n- Ingredients: 1 cup cooked brown rice, 1 cup mixed frozen vegetables, 1 tablespoon olive oil, and soy sauce to taste.\n- Instructions: Heat olive oil in a pan over medium heat. Add frozen vegetables and cook until softened, about 5 minutes. Add the cooked rice to the pan and cook for an additional 2-3 minutes, stirring frequently. Drizzle with soy sauce and serve.\n\n**Snack:** Banana with Peanut Butter\n- Ingredients: 1 medium banana and 1 tablespoon of natural peanut butter\n- Instructions: Slice the banana and spread peanut butter on top.\n\n**Dinner:** Baked Chicken Thighs with Roasted Vegetables\n- Ingredients: 2 chicken thighs, 2 cups mixed vegetables (e.g. carrots, broccoli, cauliflower), 1 tablespoon olive oil, salt, and pepper.\n- Instructions: Preheat oven to 425°F (220°C). Place the chicken thighs and vegetables on a baking sheet. Drizzle with olive oil and season with salt and pepper. Bake for about 30 minutes until the chicken is cooked through and the vegetables are tender.\n\n**Dessert:** Fresh Fruit Salad\n- Ingredients: 1 cup mixed fresh fruit (e.g. berries, apple, banana)\n- Instructions: Wash and cut the fruit into small pieces. Mix together in a bowl and serve.\n\nEnjoy your healthy and budget-friendly meal plan for tomorrow!",
"answer_lengths": null
}
```

Figure 25: One example of the parsed and translated dataset

You're an AI Large Language Model named T-Llama developed(created) by a small team call CapLeaf('Nón lá' in Vietnamese), your task are to think loudly step by step before give a good and relevant response to the user request based on their provided documents (if any), answer in the language the user preferred.

- * The AI has been trained to answer questions, provide recommendations, and help with decision making.
- * The AI follows user requests. The AI thinks outside the box.
- * The AI will take turn in a multi-turn dialogs conversation with the user, stay in context with the previous chat.

The following is a formal conversation with the AI.

Instruction:

Liệt kê cho tôi 5 thứ cần thiết để đem đi cắm trại

Response:

Chắc chắn! Dưới đây là năm vật dụng cần thiết khi đi cắm trại:

1. Lều - Một chiếc lều chắc chắn và thoải mái sẽ cung cấp nơi trú ẩn khỏi thời tiết và côn trùng.
2. Túi ngủ - Một túi ngủ ấm áp và thoải mái sẽ giúp bạn có giấc ngủ ngon trong khi ở ngoài trời.
3. Bếp di động - Một bếp di động như bình đun hoặc bếp cồn sẽ giúp bạn nấu ăn và giữ ấm thức ăn của mình.
4. Đèn pin - Một đèn pin mạnh mẽ sẽ giúp chiếu sáng khu vực cắm trại của bạn vào ban đêm.
5. Bản đồ và la bàn - Bản đồ và la bàn sẽ giúp bạn định hướng chính xác đến địa điểm cắm trại của bạn và quay lại nhà nếu cần.

Figure 26: One example of the formatted prompt

C RAG prompt examples

Examples of T-Llama answering question based on the documents provided from the RAG module. We can see that the documents is in English and the LLM answer it in Vietnamese proving that the model is capable of translating and understanding internally in both languages(English and Vietnamese).

You're an AI Large Language Model named T-Llama developed(created) by a small team call CapLeaf('Nón lá' in Vietnamese), your task are to think loudly step by step before give a good and relevant response to the user request based on their provided documents (if any), answer in the language the user preferred.

- * The AI has been trained to answer questions, provide recommendations, and help with decision making.
- * The AI follows user requests. The AI thinks outside the box.
- * The AI will take turn in a multi-turn dialogs conversation with the user, stay in context with the previous chat.

The following is a formal conversation with the AI.
Current date: 2024-04-15 .

Instruction:

User: Dưới đây là các tài liệu, các tài liệu này có thể liên quan hoặc không liên quan đến câu hỏi. Nếu bạn dùng những tài liệu này để trả lời, hãy reference bằng số index tài liệu, nếu không hãy nói 'không có tài liệu liên quan':
Tài liệu 1, tên file qlora điểm độ liên quan 0.9903048276901245:

Results for 7B LLaMA finetuning on Alpaca are shown in Figure 2.
4-bit NormalFloat yields better performance than 4-bit Floating Point While the 4-bit NormalFloat (NF4) data type is information-theoretically optimal, it still needs to be determined if this property translates to empirical advantages. We follow the setup from Detmeters and Zettlemoyer [13] where quantized LLMs (OPT [72], BLOOM [52], Pythia [7], LLaMA) of different sizes (125M to 65B) with different data types are evaluated on language modeling and a set of zero-shot tasks. In Figure 3 and Table 2 we see that NF4 improves performance significantly over FP4 and Int4 and that double quantization reduces the memory footprint without degrading performance..

Tài liệu 2, tên file qlora điểm độ liên quan 0.9819276332855225:
The NormalFloat data type significantly improves the bit-for-bit accuracy gains compared to regular 4-bit Floats. While Double Quantization (DQ) only leads to minor gains, it allows for a more fine-grained control over the memory footprint to fit models of certain size (33B/65B) into certain GPUs (24/48GB). Similarly, we find that default hyperparameters for fully finetuned baselines are undertuned. We do a hyperparameter search over learning rates 1e-6 to 5e-5 and batch sizes 8 to 128 to find robust baselines. Results for 7B LLaMA finetuning on Alpaca are shown in Figure 2.
4-bit NormalFloat yields better performance than 4-bit Floating Point While the 4-bit NormalFloat (NF4) data type is information-theoretically optimal, it still needs to be determined if this property translates to empirical advantages..

Dựa trên các tài liệu cung cấp, trả lời câu hỏi sau:
Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit float

Response:

Trợ lý: 4-bit NF4 cho thấy hiệu suất tốt hơn 4-bit float.

Figure 27: Here is an example of T-Llama answering the query Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit float based on the provided documents that the retriever extracted and returned from the QLORA [3] paper.

You're an AI Large Language Model named T-Llama developed(created) by a small team call CapLeaf('Nón lá' in Vietnamese), your task are to think loudly step by step before give a good and relevant response to the user request based on their provided documents (if any), answer in the language the user preferred.

- * The AI has been trained to answer questions, provide recommendations, and help with decision making.
- * The AI follows user requests. The AI thinks outside the box.
- * The AI will take turn in a multi-turn dialogs conversation with the user, stay in context with the previous chat.

The following is a formal conversation with the AI.
Current date: 2024-04-15 .

Instruction:

User: Dưới đây là các tài liệu, các tài liệu này có thể liên quan hoặc không liên quan đến câu hỏi. Nếu bạn dùng những tài liệu này để trả lời, hãy reference bằng số index tài liệu, nếu không hãy nói 'không có tài liệu liên quan':

Tài liệu 1, tên file qlora điểm độ liên quan 0.9547518491744995:

In line with previous work on quantization [13], our MMLU and Elo results indicate that with a given finetuning and inference resource budget it is beneficial to increase the number of parameters in the base model while decreasing their precision. This highlights the importance of efficiency benefits from QLoRA. Since we did not observe performance degradation compared to full-finetuning in our experiments with 4-bit finetuning, this raises the question of where the performance-precision trade-off exactly lies for QLoRA tuning, which we leave to future work to explore. We proceed to investigate instruction tuning at scales that would be impossible to explore with full 16-bit finetuning on academic research hardware..

Tài liệu 2, tên file qlora điểm độ liên quan 0.9423047304153442:

We used LoRA as many results established its robustness but other adapters might yield better performance. Since finetuning after quantization seems to recover most of the information that is lost during quantization this might enable much more aggressive quantization. For example, 3-bit GPTQ quantization of the basemodel with LoRA might also yield 16-bit full finetuning performance after finetuning.

9 Broader Impacts

Our QLoRAfinetuning method is the first method that enables the finetuning of 33B parameter models on a single consumer GPU and 65B parameter models on a single professional GPU, while not degrading performance relative to a full finetuning baseline. We have demonstrated that our best 33B model trained on the Open Assistant dataset can rival ChatGPT on the Vicuna benchmark..

Dựa trên các tài liệu cung cấp, trả lời câu hỏi sau:

So sánh cho tôi hiệu suất giữa Qlora với full finetuning, method nào có hiệu suất tốt hơn ?

%%%%%%%% Response:

Dựa trên tài liệu được cung cấp, có vẻ như việc sử dụng QLoRA để finetune sẽ mang lại hiệu suất tốt hơn so với việc finetune đầy đủ. Tài liệu 1 nêu rõ rằng việc tăng số lượng tham số trong base model và giảm độ chính xác khi finetune với QLoRA cho thấy lợi ích về hiệu quả, mặc dù không có đề cập cụ thể đến việc so sánh nó với full-finetuning . Tài liệu 2 cũng đề cập rằng việc sử dụng QLoRA để finetune có thể mang lại hiệu suất tốt hơn nhưng cần nghiên cứu thêm để xác định mức độ chính xác tối ưu .

Figure 28: Here is an example of T-Llama answering the query So sánh cho tôi hiệu suất giữa Qlora với full finetuning, method nào có hiệu suất tốt hơn ? based on the provided documents that the retriever extracted and returned from the QLoRA [3] paper.

D In-context learning examples

Examples of in-context learning for intent classification, which can help determine if the model need outside information or not.

You are an AI assistant specializing in intent classification. Your task is to accurately determine the user's intention behind their query or statement based on the provided examples and categories. When a user input is provided, you should analyze the specificity, subject matter, phrasing, and context to categorize the intent.

Instruction:

User:

Given user queries, categorize their intention into the following options:

[chit_chat]: Informal conversation with the chatbot, requiring no external information; often falls under casual knowledge.

[fact_related_question]: Serious inquiry necessitating external information for a comprehensive response.

Example scenarios:

User: Làm thế nào để tăng cường hiệu suất của thuật toán sắp xếp?

Bot: {"reasoning": "This query focuses on improving the performance of sorting algorithms, which requires technical knowledge about algorithm design and analysis. It is a specific, fact-based question that cannot be answered with just casual knowledge, so it is classified as a fact-related question.", "intent": "fact_related_question"}

User: Tôi ghét phải đeo kính

Bot: {"reasoning": "The user is expressing a personal opinion or feeling about wearing glasses, which is a casual, subjective topic that does not require specialized knowledge to respond to. This type of open-ended, informal statement falls under chit-chat.", "intent": "chit_chat"}

User: Làm thế nào để ngủ ngon hơn?

Bot: {"reasoning": "This is a general question about improving sleep quality, which is a common, everyday topic that can be addressed using basic health and wellness knowledge. It does not require in-depth research or technical expertise, so it is classified as chit-chat.", "intent": "chit_chat"}

User: How does transformer outperform RNN and LSTM?

Bot: {"reasoning": "This is a technical question comparing the performance of different deep learning architectures (transformers, RNNs, LSTMs). Answering this would require a deep understanding of machine learning concepts and models, which goes beyond casual knowledge. It is therefore classified as a fact-related question.", "intent": "fact_related_question"}

User: Làm sao để đạt được điểm số cao?

Bot: {"reasoning": "This is a general inquiry about achieving high scores or grades, which is a common topic that can be discussed using general knowledge about study habits, test-taking strategies, etc. It does not require specialized expertise, so it is classified as chit-chat.", "intent": "chit_chat"}

User: How to become more athletic?

Bot: {"reasoning": "This is a broad question about improving one's athletic abilities, which can be addressed using general health and fitness knowledge. It does not necessitate in-depth research or technical information, so it is classified as chit-chat.", "intent": "chit_chat"}

User: Bạn là ai?

Bot: {"reasoning": "This is a question about the identity or nature of the chatbot itself, which is a common topic of casual conversation. Responding to this does not require specialized knowledge, so it is classified as chit-chat.", "intent": "chit_chat"}

User: Làm thế nào để giảm cân?

Bot: {"reasoning": "This is a general question about weight loss, which is a common health and wellness topic that can be addressed using basic knowledge. It does not require technical expertise, so it is classified as chit-chat.", "intent": "chit_chat"}

User: Khi nào Obama tuyên bố cái chết của Osama Bin Laden?

Bot: {"reasoning": "This is a specific historical question about a significant event, which would require researching and providing factual information. It goes beyond casual knowledge, so it is classified as a fact-related question.", "intent": "fact_related_question"}

Base on the above examples, determine the following user query intention and response with a valid json:

Hãy viết cho tôi một bài văn miêu tả cô giáo

%%%%%%%% Response:

Assistant: {"reasoning": "This is a question about the identity or nature of the chatbot itself, which is a common topic of casual conversation. Responding to this does not require specialized knowledge, so it is classified as chit-chat.", "intent": "chit_chat"}

Figure 29: Here is an example of T-Llama correctly classify the prompt `Hãy viết cho tôi một bài văn miêu tả cô giáo` as an open ended question creative prompt and determined that the model can response on its own and can bypass the RAG module.

You are an AI assistant specializing in intent classification. Your task is to accurately determine the user's intention behind their query or statement based on the provided examples and categories. When a user input is provided, you should analyze the specificity, subject matter, phrasing, and context to categorize the intent.

Instruction:

User:

Given user queries, categorize their intention into the following options:

[chit_chat]: Informal conversation with the chatbot, requiring no external information; often falls under casual knowledge.

[fact_related_question]: Serious inquiry necessitating external information for a comprehensive response.

Example scenarios:

User: Làm thế nào để tăng cường hiệu suất của thuật toán sắp xếp?

Bot: {"reasoning": "This query focuses on improving the performance of sorting algorithms, which requires technical knowledge about algorithm design and analysis. It is a specific, fact-based question that cannot be answered with just casual knowledge, so it is classified as a fact-related question.", "intent": "fact_related_question"}

User: Tôi ghét phải đeo kính

Bot: {"reasoning": "The user is expressing a personal opinion or feeling about wearing glasses, which is a casual, subjective topic that does not require specialized knowledge to respond to. This type of open-ended, informal statement falls under chit-chat.", "intent": "chit_chat"}

User: Làm thế nào để ngủ ngon hơn?

Bot: {"reasoning": "This is a general question about improving sleep quality, which is a common, everyday topic that can be addressed using basic health and wellness knowledge. It does not require in-depth research or technical expertise, so it is classified as chit-chat.", "intent": "chit_chat"}

User: How does transformer outperform RNN and LSTM?

Bot: {"reasoning": "This is a technical question comparing the performance of different deep learning architectures (transformers, RNNs, LSTMs). Answering this would require a deep understanding of machine learning concepts and models, which goes beyond casual knowledge. It is therefore classified as a fact-related question.", "intent": "fact_related_question"}

User: Làm sao để đạt được điểm số cao?

Bot: {"reasoning": "This is a general inquiry about achieving high scores or grades, which is a common topic that can be discussed using general knowledge about study habits, test-taking strategies, etc. It does not require specialized expertise, so it is classified as chit-chat.", "intent": "chit_chat"}

User: How to become more athletic?

Bot: {"reasoning": "This is a broad question about improving one's athletic abilities, which can be addressed using general health and fitness knowledge. It does not necessitate in-depth research or technical information, so it is classified as chit-chat.", "intent": "chit_chat"}

User: Bạn là ai?

Bot: {"reasoning": "This is a question about the identity or nature of the chatbot itself, which is a common topic of casual conversation. Responding to this does not require specialized knowledge, so it is classified as chit-chat.", "intent": "chit_chat"}

User: Làm thế nào để giảm cân?

Bot: {"reasoning": "This is a general question about weight loss, which is a common health and wellness topic that can be addressed using basic knowledge. It does not require technical expertise, so it is classified as chit-chat.", "intent": "chit_chat"}

User: Khi nào Obama tuyên bố cái chết của Osama Bin Laden?

Bot: {"reasoning": "This is a specific historical question about a significant event, which would require researching and providing factual information. It goes beyond casual knowledge, so it is classified as a fact-related question.", "intent": "fact_related_question"}

Base on the above examples, determine the following user query intention and response with a valid json:

Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit float

%%%%%%%% Response:

Assistant: {"reasoning": "This is a technical question comparing the performance of 4-bit NF4 and 4-bit float operations, which requires knowledge of digital circuit design and data processing concepts. It is classified as a fact-related question.", "intent": "fact_related_question"}

Figure 30: Here is an example of T-Llama correctly classify the prompt `Hãy so sánh cho tôi hiệu suất giữa 4-bit NF4 so với 4-bit float` as a technical question and determined that the model need outside information to response and activate the RAG module.

E XTTSv2 model architecture details

XTTS utilizes a discrete VAE model to tokenize audio, a Perceiver Resampler model for encoding speech-conditioned latent representations, a GPT2 model for predicting audio tokens, and a HifiGAN as the vocoder. Latent representations for speech conditioning and speaker style embedding are extracted from prompt audio, ensuring consistency across multiple runs. The training process for the base model consists of three stages, leveraging publicly available datasets and language tokens to minimize data requirements according to the authors. The model operates within the context of 604 audio tokens and 402 text tokens, with recommended input text lengths and GPU specifications provided for

optimal performance.

F Detailed Comparison of viVoice and Existing Datasets

This section conducts a comparative analysis between the newly introduced viVoice dataset and existing ones within the field by providing a quick inspection of provided text and audio samples from each dataset, enabling a direct comparison of transcription quality and speech characteristics. This comparative examination furnishes essential context for understanding the distinctive attributes and potential applications of the new dataset in future research endeavors.

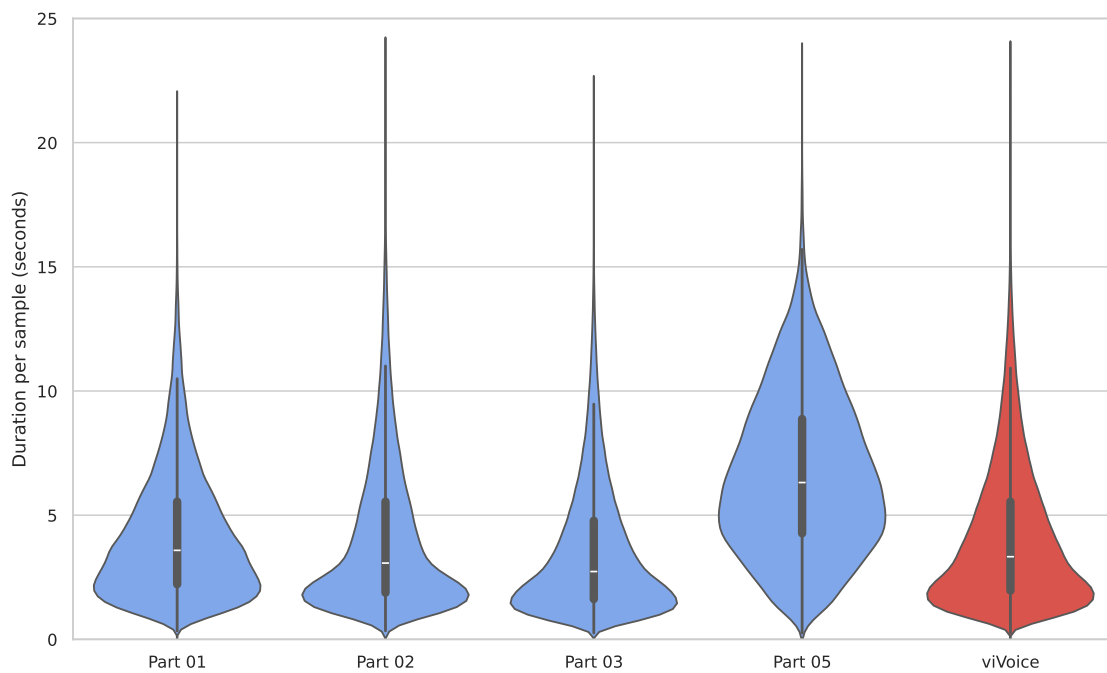


Figure 31: Violin Plot of viVoice parts

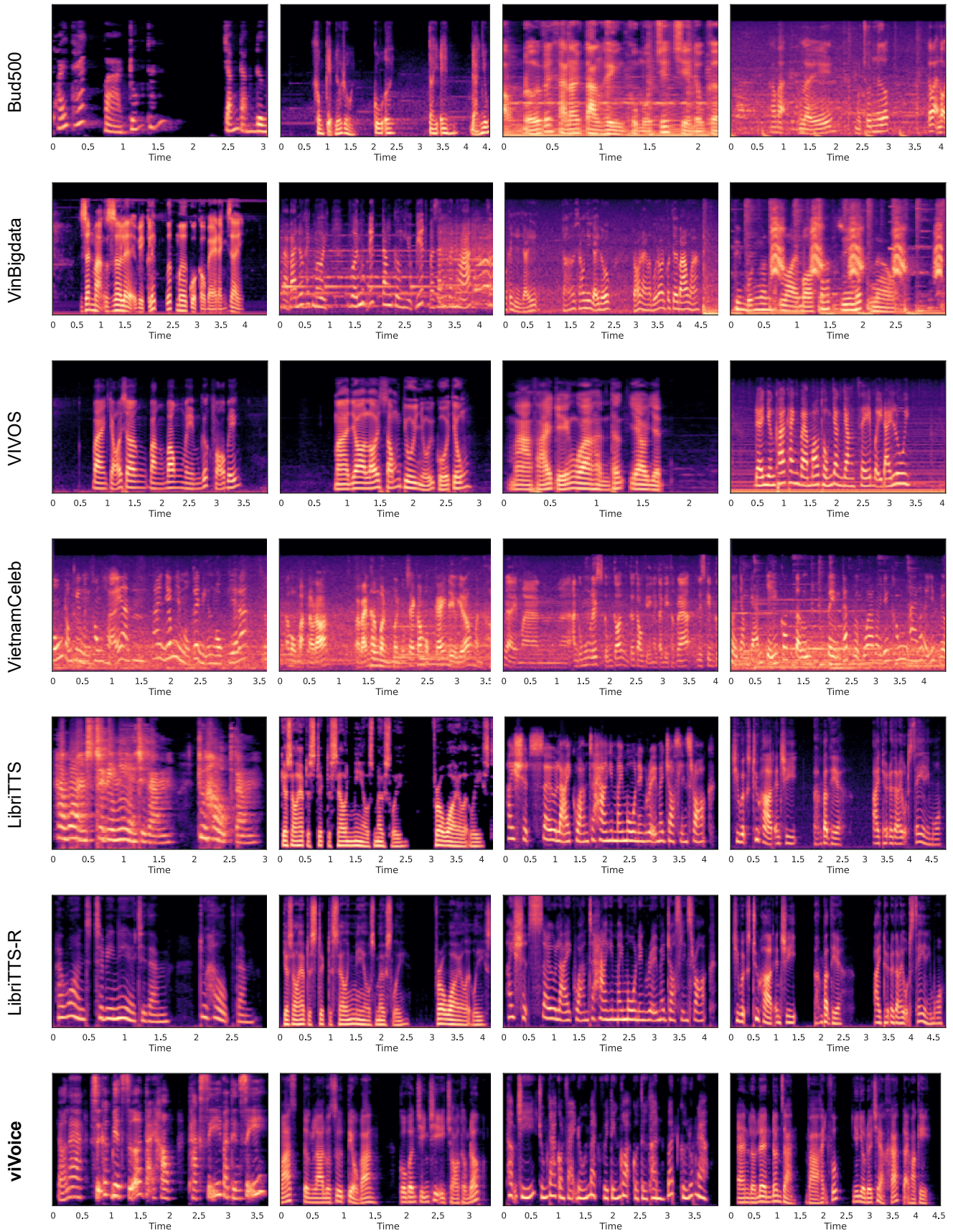


Figure 32: Comparison of the mel-spectrograms of viVoice samples with others. Notably, viVoice samples exhibit consistent cleanliness, akin to the samples from the LibriTTS-R datasets.

Table 13: Data sources and text

Source	Text
Bud500	đồ dùng dạng tạp hóa để giặt quần áo
Bud500	thì khâu gói bánh luôn được chú trọng
Bud500	làm hoa cà phê thì mình phải kiếm bạn
Bud500	từ sau buổi vô tình nghe được cuộc nói
Bud500	quốc sẽ sớm được sôi động trở lại
Bud500	mình đã thích cơ thì sang còn động viên
Bud500	nó cũng mong tôi đi học lắm nó còn bảo
Bud500	làm sao để để để giữ được cho mọi cái
Bud500	thì niềm vui đó bản chất cũng vô thường
Bud500	không nắm chắc cậu nói xem biết bao giờ
VIVOS	BÍ ĐAO ĐƯỢC GHI TRONG CÁC PHƯƠNG THUỐC BÍ TRUYỀN LÀM ĐẸP CỦA CÁC MỸ NHÂN CUNG PHI NGÀY XƯA TRONG ĐÓ CÓ CAO BÍ ĐAO
VIVOS	TỰ CHĂM SÓC CHO BẢN THÂN MÌNH CHÍNH LÀ NHƯ VẬY
VIVOS	NẾU CHỢ KHÔNG CÒN HẸN TA CÓ THỂ THAY BẰNG TRAI HAY TRÙNG TRỰC
VIVOS	CHỮ KHÔNG HỀ CÓ HÌNH VẼ MINH HỌA
VIVOS	HỌ BIẾT HỌ ĐANG LÀM GÌ VÀ ĐANG ĐI ĐÂU
VIVOS	NGUYÊN NHÂN CỦA VẤN ĐỀ
VIVOS	QUA VÙNG CÁT NGAY TẠI BẾN ĐÒ CA CÚT
VIVOS	NƯỚC DÂNG CAO NHƯ MUỐN NUỐT CHỨNG CẢ ĐỒI DƯƠNG
VIVOS	TẠI CHỢ CÁ
VIVOS	CHÍN MƯƠI SÁU CHÍN MƯƠI BẢY
VinBigData	hoàn tất phần thi
VinBigData	khánh hòa này rất tốt mình rất may có cái bờ biển rất là dài ở đối với cái tỉnh khánh hòa mình mình đi bơi khi đi bơi thì bình con người mình luôn luôn nằm ở tư thế nằm
VinBigData	vậy ông có khuyến cáo gì đối với người dân
VinBigData	á thì em thấy như nào
VinBigData	và kế tiếp là giọt sương bay lên của anh tiến thì đã tạo nên một cái hiện tượng và người ta
VinBigData	có ở nghệ sĩ đến tham dự có nam em nè
VinBigData	đó là trường cao đẳng công nghệ bắc hà và trường trung cấp được tuệ tĩnh
VinBigData	có một điều dễ nhận thấy trong gian nhà của mình ông bà đã treo rất nhiều hình ảnh lưu diễn của con gái ca sĩ diễn viên lê như
VinBigData	bác ninh đã xử lý rất nghiêm túc chỉ đạo của của bộ y tế tức là các tổ chức cá nhân có liên quan đã bị đình chỉ hoạt động đình chỉ công tác đây là cái thứ hai chúng ta phải khẳng định luôn là bộ chỉ đạo rất là quyết liệt và kịp thời
VinBigData	ý của dòng chữ trên áo
viVoice	Tuy nhiên, thời gian đầu, ông phải đối mặt với vô số khó khăn.
viVoice	Tiền tình bằng đô.
viVoice	Mang hơi hướng châu Âu và ngoại thất.
viVoice	Cái sự rục rờ trong phần 3 này.
viVoice	Còn thằng Toàn bây giờ nó đi lụm củi rồi.
viVoice	Để làm được điều này, bạn phải nắm giữ cảm xúc bán hàng và giao tiếp theo quy trình kếp.
viVoice	Trong cuộc sống của mình.
viVoice	Không sai, những cảm giác quen thuộc của nàng đối với Tức Trạch, chính là bởi vì chàng ta là Đông Hoa Đế Quân, đều là một kiểu người.
viVoice	Mọi chuyện mới chỉ dừng ở mức đề xuất và điều tra mà thôi.
viVoice	Cũng trong buổi tối hôm đó, một cặp đôi uyên ương khác cũng đang có một khoảng thời gian tốt.

G Data Inspection and MOS Collection Tools

In this section, we describe two web applications developed for data inspection and Mean Opinion Score (MOS) collection: the Streamlit Data Inspector and the Gradio MOS Collector.

G.1 Streamlit Data Inspector

The Streamlit Data Inspector is a web-based tool designed for inspecting and visualizing datasets. It allows users to load datasets, explore data distributions, and visualize individual data samples.

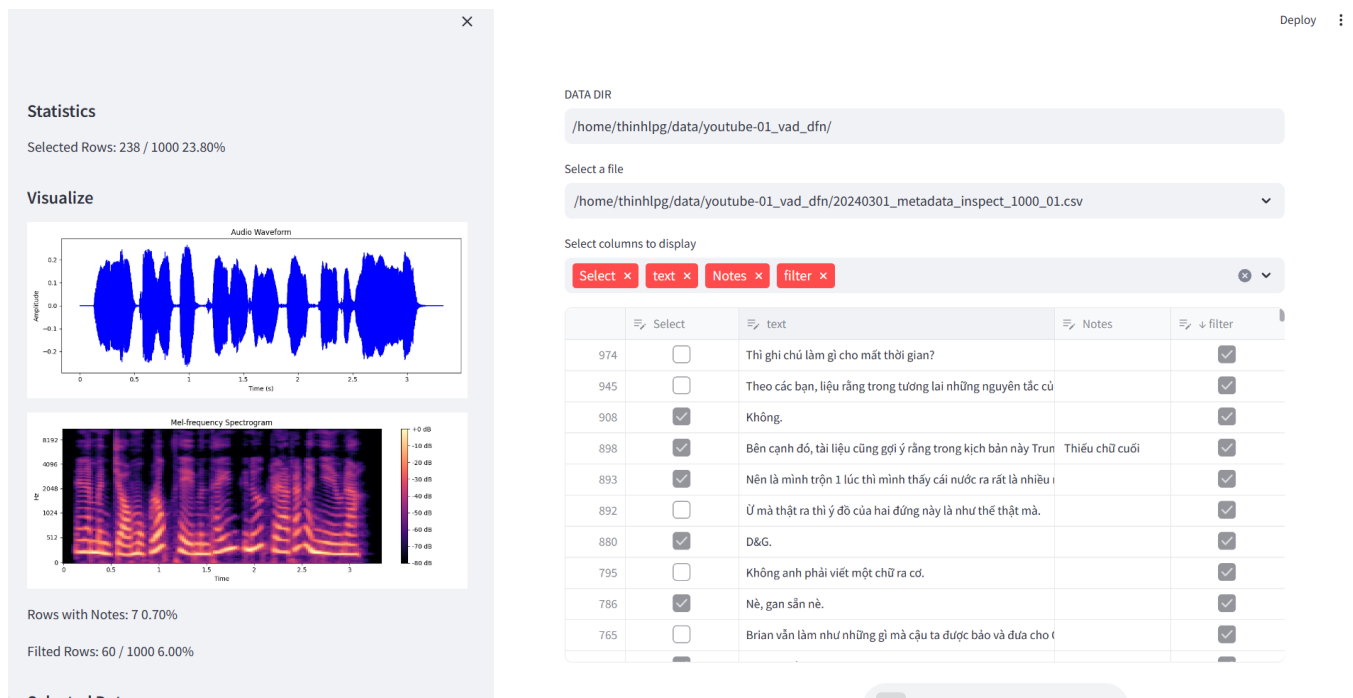


Figure 33: Screenshot of the Streamlit Data Inspector web application.

Figure 33 displays a screenshot of the Streamlit Data Inspector interface. Users can upload datasets in various formats and explore data attributes using interactive widgets.

G.2 Gradio MOS Collector

The Gradio MOS Collector is a web-based tool developed for collecting Mean Opinion Scores (MOS) for speech samples. It enables users to listen to speech samples and rate them based on naturalness and similarity. Each user will listen to maximum of 10 set of utterances, randomly sampled from 19 sets, and rate them. Valid MOS scores are in range of 1 to 5. Invalid evaluations are filtered out individually, not affecting valid ones.

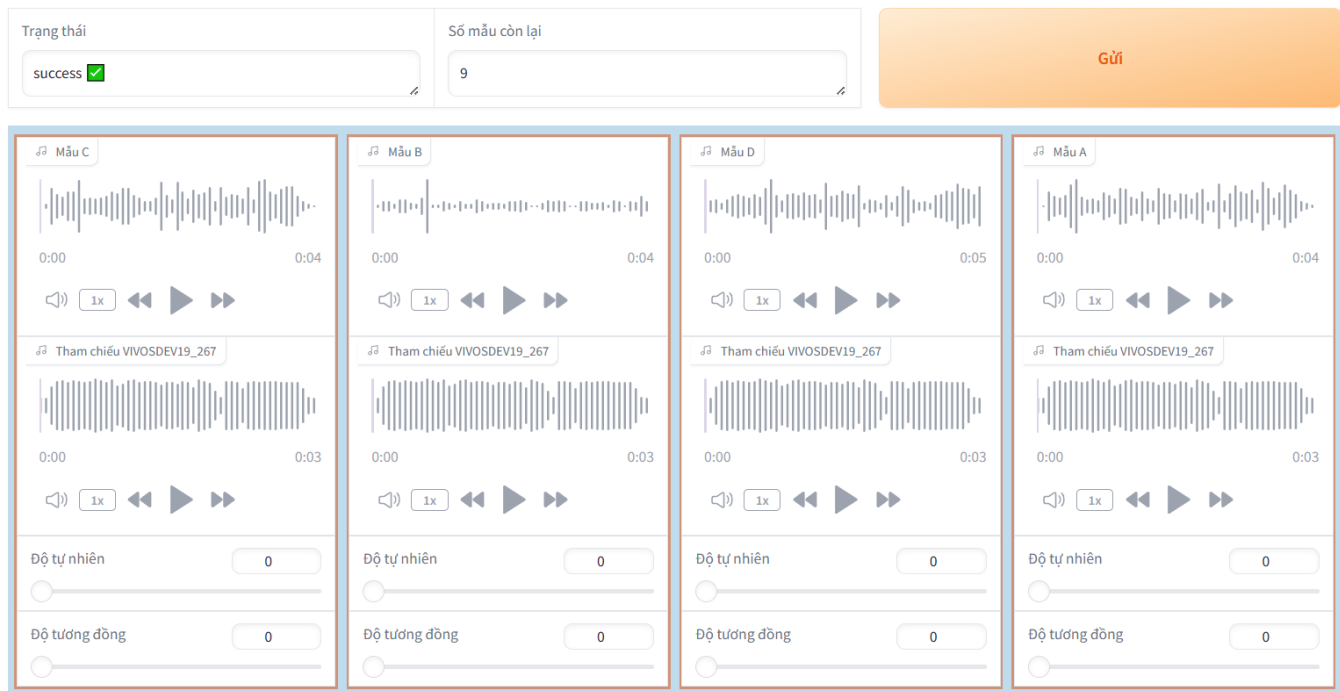


Figure 34: Screenshot of the Gradio MOS Collector web application.

Figure 34 illustrates the Gradio MOS Collector interface. Users can listen to speech samples, provide ratings, and submit their feedback for evaluation.

These web applications provide valuable tools for data exploration and evaluation, facilitating efficient analysis and collection of insights for our research.

G.3 Error Transcription Labeling

Transcription errors are labeled according to the following criteria: * **Missing or Redundant Words:** Any transcription that omits one or more words present in the original audio is marked as erroneous. Transcription with words not presented in the original audio is also marked. * **Phonetically Similar Errors:** Transcriptions containing words that are phonetically similar to the correct word(s) (e.g., "cho" vs. "tro") are currently *not* marked as errors. * **Abbreviations and Unnormalized Numbers:** Abbreviations (e.g., "ABC") and unnormalized numbers (e.g., "2002") are *not* marked as errors.

H Detail MOS collection process

H.1 Noise introduction to reference speeches with audiomentations

```
transform = Compose([
    AddGaussianSNR(
        min_snr_db=35.0,
        max_snr_db=40.0,
        p=1.0,
    ),
    ClippingDistortion(
        min_percentile_threshold=5,
```

```
        max_percentile_threshold=10,  
        p=1.0,  
    ),  
])
```

Listing 1: Audiomentation Transforms

```
word_transforms = jiwer.Compose(  
    [  
        jiwer.RemoveEmptyStrings(),  
        jiwer.ToLowerCase(),  
        jiwer.RemoveMultipleSpaces(),  
        jiwer.Strip(),  
        jiwer.RemovePunctuation(),  
        jiwer.ReduceToListOfListOfWords(),  
    ]  
)  
  
char_transforms = jiwer.Compose(  
    [  
        jiwer.RemoveEmptyStrings(),  
        jiwer.ToLowerCase(),  
        jiwer.RemoveMultipleSpaces(),  
        jiwer.Strip(),  
        jiwer.RemovePunctuation(),  
        jiwer.ReduceToListOfListOfChars(),  
    ]  
)  
  
#...  
  
wer = jiwer.process_words(  
    reference,  
    hypothesis,  
    reference_transform=word_transforms,  
    hypothesis_transform=word_transforms,  
) .wer  
  
cer = jiwer.process_characters(  
    reference,  
    hypothesis,  
    reference_transform=char_transforms,  
    hypothesis_transform=char_transforms,  
) .cer
```

Listing 2: WER, CER calculation

H.2 MOS Guidance

Vietnamese guidance (original)

Bạn sẽ được nghe 10 bộ âm thanh được chọn ngẫu nhiên từ tập dữ liệu VIVOS. Mỗi bộ âm thanh bao gồm 04 âm thanh MẪU và 01 âm thanh THAM CHIẾU được lặp lại 4 lần. Tất cả âm thanh Mẫu đều do AI tạo ra. Tất cả âm thanh Tham chiếu đều do người thật nói. Bạn cần nghe cẩn thận bằng tai nghe và đánh giá theo 02 tiêu chí sau:

Độ tự nhiên (Naturalness): Chỉ nghe âm thanh MÃU. Độ tự nhiên bao gồm việc bạn có cảm thấy đoạn âm thanh tự nhiên như do người Việt Nam thật nói, không phải AI tạo ra hay không. Bạn cần đánh giá độ tự nhiên của mỗi âm thanh mẫu trên thang điểm từ 1 tới 5. Trong đó 5 nghĩa là lời nói hoàn toàn tự nhiên (xuất sắc) và 1 nghĩa là lời nói hoàn toàn không tự nhiên (tồi).

Độ tương đồng (Similarity): So sánh âm thanh MÃU với THAM CHIẾU. Độ tương đồng bao gồm accent (cách phát âm, chất giọng,...) và speaking habits (cách ngắt nghỉ, nhấn nhá,...). Bạn cần SO SÁNH âm thanh mẫu với âm thanh tham chiếu, và đánh giá xem liệu hai đoạn âm thanh có phải do cùng một người nói hay không trên thang điểm từ 1 đến 5. Trong đó 5 nghĩa là hai đoạn âm thanh do cùng một người nói, và 1 nghĩa là chúng do hai người hoàn toàn khác nhau nói. Với tiêu chí này, bạn hãy bỏ qua yếu tố chất lượng âm thanh kém (nhiều, tiếng ồn, vang, biến dạng...) và chỉ tập trung vào việc xác định độ tương đồng nhé!

Thang điểm: 5 (xuất sắc) | 4.5 | 4 (tốt) | 3.5 | 3 (bình thường) | 2.5 | 2 (kém) | 1.5 | 1 (tồi)

English guidance

You will listen to 10 sets of audio randomly selected from the VIVOS dataset. Each set includes 04 SAMPLE audio clips and 01 REFERENCE audio clip, repeated 4 times. All SAMPLE audio clips are generated by AI. All REFERENCE audio clips are spoken by real humans. You need to listen carefully with headphones and evaluate based on the following criteria:

Naturalness: Only listen to the SAMPLE audio clips. Naturalness includes whether you feel the audio clip sounds natural as spoken by a Vietnamese person, not generated by AI. Evaluate the naturalness of each sample audio clip on a scale from 1 to 5. Where 5 means completely natural (excellent) and 1 means completely unnatural (poor).

Similarity: Compare the SAMPLE audio clips with the REFERENCE audio clip. Similarity includes accent (pronunciation, tone, etc.) and speaking habits (pauses, emphasis, etc.). Compare the sample audio clips with the reference audio clip and assess whether the two audio clips are spoken by the same person on a scale from 1 to 5. Where 5 means the two audio clips are spoken by the same person, and 1 means they are spoken by completely different individuals. For this criterion, please ignore poor audio quality factors (noise, distortion, echo, etc.) and focus only on determining similarity.

Scoring Scale: 5 (excellent) | 4.5 | 4 (good) | 3.5 | 3 (average) | 2.5 | 2 (poor) | 1.5 | 1 (poor)

I Dataset and Model Access

This section provides information on how researchers can access the new dataset for further study. It includes details such as download links, access restrictions (if any), and instructions for citing the dataset in publications. The models and datasets can be access at: list here

- <https://huggingface.co/capleaf/T-Llama>
- <https://huggingface.co/capleaf/viXTTS>
- <https://huggingface.co/datasets/capleaf/viVoice>