Title	言語認識型プルーニングと低リソース検索による高精度な知識を備えた言語特化型コンパクト大規模言語モデル
Author(s)	NGUYEN, DIEU HIEN
Citation	
Issue Date	2026-03
Type	Thesis or Dissertation
Text version	ETD
URL	https://hdl.handle.net/10119/20582
Rights	
Description	Supervisor: NGUYEN, Minh Le, 先端科学技術研究科, 博士

Doctoral Dissertation

Language-Specific Compact Large Language Models with Precise Knowledge through Language-aware Pruning and Low-resource Retrieval

NGUYEN, Hien Dieu

Supervisor NGUYEN, Minh Le

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March, 2026

Abstract

Large Language Models (LLMs) have achieved remarkable advances in reasoning, understanding, and generation across a wide range of natural language processing (NLP) tasks. However, their large scale imposes substantial computational and memory costs, posing challenges for efficient deployment, particularly in low-resource settings. Simultaneously, while retrieval-augmented methods have enhanced the factual grounding and interpretability of question answering (QA) systems, their effectiveness in low-resource languages remains limited by data scarcity and the absence of high-quality retrieval frameworks.

This thesis addresses these challenges through two complementary research directions. First, it introduces **LangCompress**, a language-aware model compression framework that integrates self-supervised instruction generation and vocabulary optimization to adapt LLMs for specific languages. LangCompress improves efficiency while preserving linguistic competence and can be seamlessly combined with existing pruning and quantization methods. Second, the thesis proposes a graph-based retrieval framework for multi-hop question answering that leverages Wikipedia's hyperlink structure to identify semantically connected evidence across documents. This framework enables efficient retrieval and reasoning in low-resource languages. To support this line of research, a generalizable dataset construction framework is developed, resulting in *VIMQA*, a Vietnamese multi-hop QA dataset designed to evaluate explainable and complex reasoning grounded in Wikipedia evidence.

Together, these contributions advance the development of retrieval- and compression-aware LLM systems that are efficient, interpretable, and inclusive across diverse linguistic environments.

[**Keywords**] low-resource language, quantization, pruning, large language model, retrieval, question-answering

Acknowledgement

I am sincerely grateful to my primary supervisor, Professor NGUYEN Le Minh, for his exceptional mentorship, profound knowledge, and unwavering support throughout my doctoral journey. I particularly appreciate his constructive critiques, stimulating discussions, and continuous encouragement, all of which have greatly enhanced both the quality of my work and my personal development. I would also like to express my deep appreciation to my second supervisor, Professor Kiyoaki Shirai, whose insightful advice and critical perspectives have been invaluable in this dissertation. Additionally, I am grateful to Professor Shinobu Hasegawa, my minor research supervisor. I sincerely appreciate his kind guidance and patient support, as well as his thoughtful feedback that helped me improve my work while accommodating the challenges I faced. I would also like to thank the professors who provided insightful comments during my pre-defense, whose guidance contributed meaningfully to refining this work.

I wish to acknowledge the Japan Advanced Institute of Science and Technology (JAIST) for providing an exceptional academic environment and the resources necessary to carry out my research. I am grateful to the school staff, whose support have greatly facilitated my academic journey. My gratitude also goes to the Japanese government for granting me the MEXT Scholarship.

I am deeply thankful to my family, friends, and colleagues for their encouragement, patience, and understanding throughout this journey. Their support has been a continuous source of motivation and strength, helping me navigate the challenges of doctoral research.

Finally, I wish to acknowledge everyone who has, in any way, contributed to the completion of this dissertation. The intellectual guidance, moral support, and practical assistance I have received from so many individuals have been indispensable, and I remain sincerely appreciative of their contributions to this rewarding journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Model Compression. Modern large language models (LLMs) have achieved strong performance across a wide range of tasks, driven by architectures with billions of parameters trained on extremely large-scale text corpora. Because training such models from the ground up requires prohibitive computational resources, current practice relies on pre-trained foundation models that are subsequently adapted to downstream applications. Although this paradigm enables broad generalization and reasoning abilities, it also poses significant challenges for practical deployment. In particular, the high memory footprint and computational demands of LLMs make inference expensive, limiting their usability in resource-constrained environments. To mitigate these limitations, model compression has become an essential research direction for improving the efficiency of LLM deployment. The goal of compression is to reduce model size and inference cost while maintaining acceptable performance. Widely adopted techniques include pruning and quantization, which have demonstrated effectiveness in lowering memory usage and latency. However, most existing compression methods [12, 76, 63, 13, 37] are developed and evaluated primarily for English-language models. As a result, their performance often fails to transfer reliably to multilingual or low-resource language settings. In such cases, compressed models can experience substantial performance degradation, largely due to uneven language coverage in pre-training data and the lack of language-aware adaptation during compression (Figure 1.1, top).

Question-Answering. Question Answering (QA) is a fundamental problem in natural language processing (NLP) and information retrieval (IR),

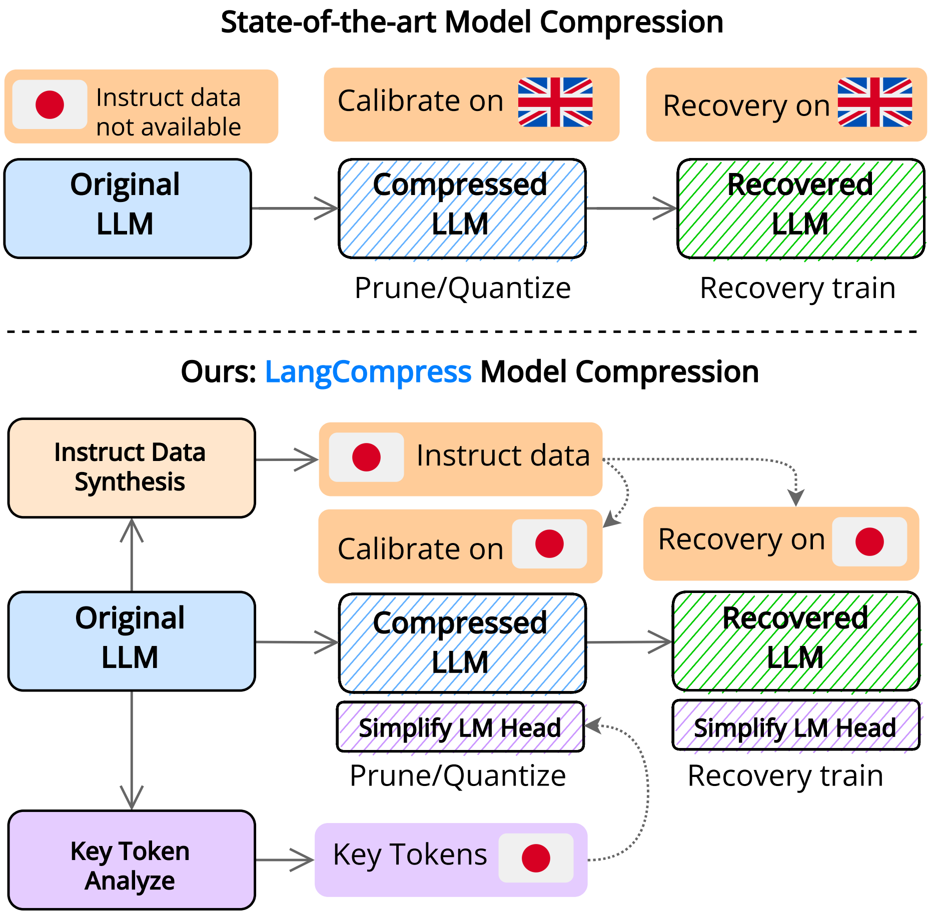


Figure 1.1: Comparison between standard compression methods and the proposed language-aware compression framework.

aiming to automatically generate correct answers to natural language questions using textual or structured knowledge sources. Effective QA systems must comprehend the semantics of both the question and its context, identify relevant information, and reason across multiple pieces of evidence. As such, QA serves as a key benchmark for assessing language understanding and reasoning capabilities.

Early QA systems, often termed traditional QA, focused on extracting factual information from single documents using lexical or statistical matching techniques [22, 60]. With the growth of large-scale text corpora such as Wikipedia, open-domain QA emerged, allowing retrieval and reasoning over diverse sources. Modern open-domain QA typically follows a dual-phase pipeline [8, 50]: a *retriever* first chooses candidate passages relevant to the query, and a *reader* then interprets these passages to extract or generate an answer. Earlier systems such as DrQA [9] relied on machine reading comprehension (MRC) models trained on annotated datasets and were effective for factoid questions but struggled with multi-hop reasoning and integrating information across documents.

Recent developments in extensive language models, exemplified by GPT, Llama, and Qwen, have profoundly influenced approaches to QA. Trained on massive multilingual corpora, LLMs demonstrate strong reasoning, instruction-following, and in-context learning abilities. Unlike span-based readers, LLMs can synthesize information from multiple retrieved passages and generate coherent, contextually grounded answers. This integration of retrieval, reasoning, and generation into a unified framework represents a natural evolution from traditional retriever–reader architectures. Figure 1.2 illustrates this progression in QA system development.

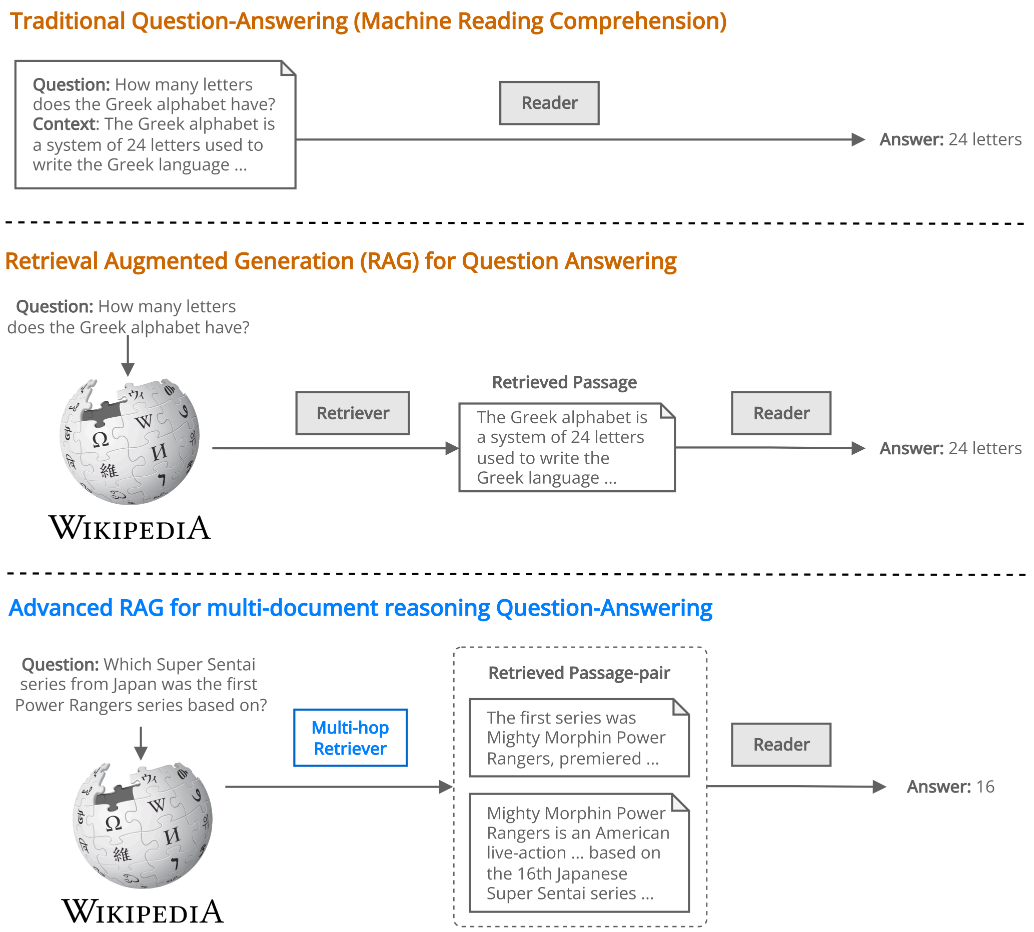


Figure 1.2: Comparison between types of Question Answering systems.

Knowledge Retrieval in the LLM Era. While open-domain QA systems have enhanced knowledge accessibility, multi-hop reasoning that necessitates the integration of evidence across documents or the inference of implicit connections between entities remains a core challenge. While multilingual encoders such as XLM-RoBERTa [11] and PhoBERT [43] have achieved strong performance on single-hop QA tasks, their performance often declines in scenarios requiring multi-step reasoning and contextual synthesis. These limitations are further exacerbated in low-resource languages, where annotated datasets for complex reasoning tasks are scarce.

Recent progress in Retrieval-Augmented Generation (RAG) has fundamentally shifted how information retrieval is integrated into QA systems. In essence, RAG builds upon the conventional retriever-reader architecture by substituting the reader component with a LLM that produces responses using the retrieved content. This paradigm allows models to access external knowledge beyond their internal parameters, improving factual grounding and interpretability. Nonetheless, the overall performance of such systems is highly dependent on the effectiveness of the retrieval stage. In particular, retrieval mechanisms that can exploit structured relationships are critical for enabling multi-step reasoning and identifying semantically related evidence across documents.

1.2 Motivation

Limitation of Low-resource model compression. Despite their impressive reasoning and generative capabilities, LLMs remain computationally expensive to deploy in practice. Operating full-scale models requires substantial GPU memory and energy consumption, which imposes significant constraints in resource-limited environments such as academic research laboratories, small enterprises, or developing regions. In such settings, users often aim to deploy LLMs for a specific languages rather than supporting all languages included in a multilingual model. To alleviate these computational constraints, researchers have increasingly explored techniques that shrink model parameters and lower resource demands. Nonetheless, the majority of compression techniques, including pruning and quantization, have been designed and evaluated primarily on English datasets. When applied to non-English languages, these techniques frequently result in disproportionate performance degradation due to differences in linguistic distribution, morphology, and vocabulary structure.

Addressing these disparities requires the development of language-aware compression strategies that selectively preserve linguistic knowledge relevant to the target language. Such methods would ensure that efficiency improvements do not compromise linguistic competence. By aligning model parameters with the statistical and semantic characteristics of a specific language, language-aware compression can facilitate more equitable and sustainable LLM deployment across diverse linguistic contexts. Furthermore, an additional source of inefficiency in LLMs lies in the final layer, the language modeling (LM) head, which maps hidden representations to an extensive multilingual vocabulary. Empirical studies show that, for any given language, only a small portion of this vocabulary is actively employed. For instance, roughly 5% of the total vocabulary may account for over 95% of the tokens in a specific language. This finding motivates a more targeted approach in which the LM head is simplified and adapted to the linguistic characteristics of the target language. Such adaptation can reduce model size and computational demands while potentially improving performance by removing redundant lexical representations.

Figure 1.1 better illustrates the conceptual distinction between conventional compression methods and the idea of our language-aware compression framework proposal. The upper part of the figure depicts the limitations of existing model compression techniques when applied to a specific non-English language, where linguistic characteristics are not adequately preserved. The lower part presents our proposed idea, which introduces language-aware mechanisms to enhance compression efficiency while maintaining language-specific performance.

Limitations of Low-resource Retrieval Approaches. RAG frameworks improve the factual accuracy of language models through the integration of outside information when producing responses. The overall performance of such systems, however, is highly dependent on the quality of the retrieval mechanism. Most existing retrievers rely on dense embedding similarity to identify relevant documents, often overlooking the structured relationships that exist between them. Consequently, they struggle to capture implicit semantic connections that are essential for multi-hop reasoning and complex question answering. Although large embedding models trained on massive datasets can better encode such semantic relationships, these resources are rarely available for low-resource languages. In contrast, Wikipedia provides a naturally interconnected knowledge source through its extensive hyperlink network, which can be leveraged to construct relational graphs representing document-level associations. Despite this potential, the use of such relational structures remains limited in current question answering pipelines, particularly for low-resource languages, where explicit link structures and annotated retrieval datasets are scarce or absent. This gap underscores the need for a retrieval framework capable of exploiting hyperlink-based knowledge graphs to retrieve contextually related evidence across multiple documents. Moreover, the scarcity of multi-hop QA datasets in low-resource languages poses a substantial barrier to both evaluation and improvement of such retrieval systems. This motivates the development of a new benchmark dataset, along with a flexible and language-agnostic framework for constructing multi-hop QA datasets across diverse linguistic settings.

Toward Efficient and Inclusive QA Systems. The limitations discussed above highlight two complementary challenges in modern QA research: (1) improving the retrieval process to enable structured, multi-hop reasoning over interconnected knowledge sources, and (2) optimizing LLM efficiency through compression techniques that remain robust in low-resource languages. Together, these directions aim to make QA systems not only more capable and interpretable, but also more inclusive and practically deployable across linguistic contexts. Addressing these challenges provides the foundation for the frameworks proposed in this study.

1.3 Research Goals and Key Contributions

To address the challenges outlined in the preceding sections, this thesis pursues two overarching research goals:

1. **Goal 1:** To design a language-aware compression framework that improves existing pruning and quantization strategies, enabling efficient deployment in multilingual and low-resource scenarios while preserving model accuracy.

2. **Goal 2:** To develop an effective retrieval framework for low-resource languages by leveraging graph-based corpora to retrieve semantically connected evidence across multiple documents, thereby enhancing multi-hop reasoning performance.

Together, these goals aim to advance the development of retrieval- and compression-aware QA systems that combine robust reasoning capabilities with practical deployability across diverse linguistic environments.

Contributions toward Language-Aware Model Compression

To achieve the second research goal, this thesis introduces LangCompress, a framework that enhances the efficiency and adaptability of LLMs through self-supervised data synthesis and vocabulary optimization tailored to specific languages.

- **Self-supervised instruction generation.** Proposed a fully automated pipeline for generating instruction-tuning data in any target language, enabling model adaptation and calibration without dependence on large-scale human-annotated datasets.

- **Vocabulary simplification and LM-head adaptation.** Develop a method for identifying core vocabulary tokens in the target language and restructuring the model's output layer to prioritize these tokens, improving both computational efficiency and linguistic accuracy.

- **Comprehensive evaluation and integration.** Extensive experiments demonstrate that LangCompress can be seamlessly combined with existing pruning and quantization methods, resulting in substantial gains in both efficiency and accuracy for language-specific downstream tasks.

Contributions toward Low-resource Multi-hop Retrieval

In pursuit of the first research goal, this thesis makes the following key contributions:

- **Framework for constructing low-resource multi-hop datasets.** This work introduces a generalizable framework for building multilingual, multi-hop question–answering datasets, initially implemented for the Vietnamese language. Using this framework, a new benchmark dataset **VIMQA** is developed to evaluate advanced reasoning capabilities and to generate explainable, multi-hop answers grounded in Wikipedia evidence.

- **Graph-based retrieval for interconnected documents.** A novel retrieval approach is proposed for multi-hop open-domain question answering, leveraging Wikipedia's hyperlink graph to discover semantically related documents. The method is computationally efficient, requiring only modest resources for training, and is inherently language-agnostic, making it adaptable to a wide range of domains that demand multi-step reasoning and evidence integration.

Chapter 2

Related Work

2.1 Model Compression for Efficient LLMs

Model compression has become an essential direction in scaling LLMs efficiently, aiming to reduce computational and memory requirements while retaining model performance. Among the most studied compression techniques are pruning and quantization, which address redundancy in model parameters from different perspectives. Pruning eliminates unnecessary parameters to enforce sparsity, while quantization reduces the precision of weights and activations. Both techniques aim to achieve a balance between efficiency and accuracy, yet their effectiveness depends heavily on the sparsity pattern, pruning granularity, and training constraints. This section reviews recent progress in unstructured, semi-structured, and structured pruning methods, followed by a discussion of quantization approaches and their limitations, especially in multilingual settings.

Pruning

Pruning-based compression methods aim to introduce sparsity into neural networks by removing redundant or low-saliency parameters, thereby reducing model size and inference cost while preserving predictive accuracy. The fundamental intuition is that large-scale neural networks are heavily over-parameterized, many of their connections contribute minimally to the output distribution, and can thus be eliminated without major degradation in performance. Depending on the granularity of pruning, approaches can be categorized as unstructured, semi-structured, or structured pruning.

Unstructured and Semi-Structured Pruning.

Unstructured pruning operates at the finest granularity by selectively zeroing out individual weights in the network. This approach offers high flexibility and can achieve substantial sparsity ratios, as each connection is independently evaluated. This idea can be traced back to early works such as Optimal Brain Damage [32] and Optimal Brain Surgeon [23], which leveraged second-order Taylor expansions of the loss function to quantify each parameter's contribution to the overall objective. Parameters associated with small second-order derivatives were considered unimportant and pruned to achieve parameter efficiency without retraining from scratch. These foundational studies established the theoretical link between pruning and the curvature of the loss landscape, but their reliance on explicit Hessian computation made them computationally prohibitive for modern large-scale networks. With the advent of transformer-based LLMs, pruning research has shifted toward post-training pruning (PTP), which involves eliminating parameters directly from pretrained models without retraining. This paradigm is especially attractive for billion-scale models, where fine-tuning or retraining is prohibitively expensive. Modern PTP techniques such as AdaPrune [34] and SparseGPT [12] approximate second-order curvature using local blockwise matrix updates, achieving efficient weight pruning with complexity reduced from $O(N^4)$ to $O(N^3)$. These approaches reconstruct remaining weights after pruning to minimize local reconstruction error, preserving model fidelity even under high sparsity levels. To further improve scalability, recent research has explored simpler pruning metrics that avoid heavy Hessian approximations. Simple Pruning (Wanda) [63] prunes weights based on the product of their magnitudes and corresponding input activations, capturing both static weight importance and dynamic activation strength. This design allows pruning without retraining, enabling direct deployment of the pruned model. Plug-and-Play Pruning (RIA) [76] extends this idea by introducing the Relative Importance and Activations metric, which accounts for both weight magnitudes and activation relationships, thereby preserving neuron interactions critical to LLM reasoning. RIA also integrates a channel permutation mechanism that reorganizes weight structures to better align with semi-structured sparsity constraints, resulting in improved trade-offs between compression and performance across multiple LLaMA variants. OptiPrune [31] represents another step forward in adaptive post-training pruning. Through a comprehensive analysis, it demonstrates that pruning effectiveness depends on the distribution of sparsity across layers. Uniform sparsity is often optimal at low compression ratios due to stability and balanced representation, whereas non-uniform sparsity yields better results under high compression by focusing pruning on redundant layers. OptiPrune adapts dynamically between these two strategies, employing layerwise deviation thresholds to balance efficiency and performance. This adaptivity allows it to maintain robustness across a broad range of architectures and sparsity levels, setting a new benchmark for pruning consistency.

Despite their conceptual elegance, unstructured methods face practical deployment challenges. The irregular sparsity patterns they produce lead to inefficient memory access and poor utilization of hardware accelerators such as GPUs or TPUs, which are optimized for dense matrix operations. Consequently, their theoretical computational savings do not translate directly into runtime speedups. To bridge this gap, semi-structured pruning introduces constrained sparsity patterns that retain partial regularity. The most widely adopted configuration is the N:M sparsity pattern [42], where N out of every M weights within a local block are pruned. This regular structure aligns with specialized GPU kernels that can skip fixed zero positions, enabling substantial practical speedups. Semi-structured pruning thus strikes an effective balance between compression flexibility and hardware compatibility. The RIA framework [76] demonstrates that incorporating activation-aware metrics and channel permutation under N:M constraints can even yield pruned LLMs that outperform their dense counterparts in zero-shot evaluations. These findings highlight that structured alignment between sparsity, activation dynamics, and hardware execution can produce models that are not only smaller and faster but occasionally more robust. Semi-structured pruning, therefore, serves as a pragmatic direction for deploying compressed LLMs at scale, especially in latency-sensitive applications.

Structured Pruning.

While unstructured and semi-structured pruning operate at the level of individual or grouped weights, they often produce models with irregular sparsity patterns. Such sparsity typically requires custom kernels or specialized hardware support to realize inference speedups, limiting its practical impact in real-world deployment scenarios. In contrast, structured pruning removes higher-level architectural components such as attention heads, MLP channels, and even entire transformer blocks, resulting in smaller, fully-dense networks with contiguous computation flows. This property ensures compatibility with highly optimized dense matrix multiplication libraries and makes structured pruning particularly attractive for accelerating inference on commodity hardware. Structured pruning has gained significant traction in the context of LLMs, where model sizes frequently exceed billions of parameters, posing severe challenges in memory capacity, computational efficiency, and energy consumption. A key design goal is therefore to compress the network while retaining its strong general-purpose generation and reasoning capabilities.

LLM-Pruner [39] adopts a task-agnostic structured pruning strategy that selectively removes non-critical coupled structures inside transformer blocks. The method evaluates the contribution of each structural unit using gradient-based saliency, targeting components that contribute minimally to model predictions across diverse downstream tasks. A notable strength of this approach is that it does not require access to the original pretraining corpus, reducing data handling burdens and enabling a practical post-training workflow. To restore model quality, a lightweight recovery stage is applied using LoRA-based fine-tuning [24], requiring only 50K examples and a short adaptation period (e.g., three hours). Experiments on LLaMA [66], Vicuna [10], and ChatGLM [75] show that the pruned models retain strong zero-shot reasoning and language generation performance while achieving substantial reductions in parameter count, making the framework suitable for multi-purpose LLM deployment under resource constraints.

SliceGPT [5] takes a different structured compression perspective by reducing the embedding dimension throughout the network. Instead of pruning individual units or branches, the method slices dense weight matrices into lower-dimensional subspaces, effectively slimming the width of the transformer while keeping the computation path intact. This design is enabled by the insight of computational invariance in transformer networks, which suggests that core representational properties can be preserved even when projection dimensionality is reduced. Crucially, SliceGPT performs sparsification in a post-hoc manner without requiring additional data structures or modification to execution kernels. This leads directly to runtime gains: slicing up to 25% of parameters from LLaMA2-70B [66] preserves up to 99% of zero-shot performance while reducing memory usage and inference compute by up to 36% on widely available A100 and consumer GPUs. These results demonstrate that structural reductions in network width can serve as a scalable and hardware-friendly alternative to conventional weight sparsification.

Overall, structured pruning represents a practical pathway toward efficient LLM deployment, enabling model downsizing while preserving dense computation for minimal disruption to existing acceleration infrastructure. Compared to unstructured pruning, these methods provide more predictable latency improvements, lower implementation barriers, and better alignment with production hardware environments. As the field continues to push toward trillion-parameter frontier models, structured pruning is likely to remain a central strategy in the design of resource-efficient large-scale LLMs.

2.2 Question Answering

The Retriever–Reader Paradigm for Open-Domain QA. The two-stage Retriever–Reader architecture has long been the dominant paradigm for open-domain QA, where the knowledge source is a large and heterogeneous corpus such as Wikipedia or web-scale content. The core motivation behind this design is to decouple knowledge access from knowledge reasoning: rather than relying solely on parametric memory within the model, a retriever first identifies a small set of potentially relevant passages, after which a reader processes these passages to extract or generate the answer. This separation enables high scalability, interpretable evidence, and the ability to continually update knowledge without retraining the system.

Early retrievers were predominantly sparse-vector methods such as TF-IDF and BM25, which measure lexical overlap and remain strong baselines due to their simplicity and efficiency (e.g., BERTserini [73]). However, sparse methods often struggle with semantic mismatch when relevant evidence uses paraphrased language. To address this challenge, dense retrievers embed questions and passages into a shared vector space using dual-encoder architectures, enabling semantic similarity search via approximate nearest neighbor techniques [26, 20, 16, 58]. Scaling dense retrieval to billions of documents and fine-grained matching has been advanced by late-interaction models such as ColBERT [28] and COLBERTv2 [61], which preserve token-level semantics with lightweight interaction at retrieval time. Baleen [27] further explores iterative retrieval and document expansion to handle long-tail queries and sparse supervision. More recent work extends retrieval to richer knowledge modalities. For instance, retrieval combined with graph reasoning enables propagation across entity relationships [35], while multi-modal retrieval broadens QA to settings requiring images or structured tables [36]. These developments reflect a growing recognition that real-world information needs demand more than unstructured text alone.

Once passages are retrieved, a reader is applied to interpret and answer the query. Reader architectures generally fall into two broad categories: extractive readers, which locate answer spans within retrieved texts [26, 20], and generative readers, which employ sequence-to-sequence generation to synthesize answers [33, 71]. Generative readers have become especially prominent due to their flexibility in handling paraphrasing, incomplete evidence, and multi-document synthesis. A particularly challenging variant of open-domain QA is multi-hop QA, where a question cannot be answered from a single passage alone. Instead, successful reasoning requires aggregating information across multiple documents or entities [74]. Accordingly, retrievers must locate multiple complementary passages, and readers must perform complex cross-passage reasoning [2, 71]. Iterative retrieval strategies, retrieval conditioned on intermediate reasoning steps, and supervised attention to supporting evidence have all been explored to address this challenge.

Finally, a crucial refinement to the pipeline is reranking, where initial retrieval results are re-ordered using more expressive but computationally expensive methods. Reranking using cross-encoders [49], generative scoring [55], or reader-aware signals [48, 40] has consistently shown to yield meaningful improvements in end-to-end QA accuracy. This step helps mitigate retrieval errors and ensures the reader processes the most relevant evidence.

Overall, the Retriever–Reader pipeline offers a modular and scalable framework that can leverage growing corpora and evolving retrieval technologies. Despite emerging alternatives such as end-to-end retrieval-augmented generation with large language models, the two-stage architecture continues to serve as a strong backbone for interpretable and efficient open-domain QA systems.

Graph-based Retrieval and Relational Knowledge. Beyond conventional lexical or dense retrieval, recent research has increasingly explored graph-based retrieval frameworks that explicitly model inter-document and inter-entity relationships to support complex information needs. Early studies demonstrated that leveraging hyperlink structures in large corpora such as Wikipedia can significantly improve multi-hop question answering by enabling explicit reasoning paths and enhancing interpretability [3]. By traversing hyperlink graphs, these approaches move beyond isolated document relevance and instead exploit the relational structure inherent in curated knowledge sources. Building on this intuition, more recent systems integrate graph construction and traversal directly into retrieval pipelines. DGR-CoQA [35] dynamically constructs document graphs during retrieval, allowing the system to iteratively expand relevant evidence across multiple hops. This dynamic graph formulation enables adaptive reasoning over evolving evidence sets, particularly in open-domain question answering scenarios. Similarly, GraphRAG [41] incorporates entity- and document-level graphs into retrieval-augmented generation, using graph traversal to select contextually related passages and improve factual coherence in generated responses. Collectively, these methods demonstrate that explicit relational structures (e.g., hyperlinks, shared entities, and citation links) provide critical contextual signals for multi-hop reasoning and evidence aggregation. Graph-based retrieval not only improves recall for compositional queries but also offers greater transparency by exposing reasoning chains across documents. However, most existing approaches are developed and evaluated primarily on high-resource languages and well-curated corpora, where relational signals such as hyperlinks and entity annotations are dense and reliable. In contrast, their applicability to low-resource or non-English languages remains underexplored, as relational graphs in these settings are often sparse, incomplete, or noisy. Addressing this limitation is crucial for extending graph-based retrieval and reasoning frameworks to multilingual and low-resource contexts.

Multilingual and Low-Resource QA. While much of the progress in QA has been driven by benchmarks and systems developed for high-resource languages (e.g., most notably English and Chinese), there is increasing recognition that these advances do not readily generalize to multilingual and low-resource settings. To mitigate data scarcity, early work has focused on cross-lingual and multilingual QA, leveraging shared representations to transfer knowledge across languages. For example, Asai et al. [4] propose a cross-lingual dense retriever that retrieves documents written in multiple languages for a question posed in a different language, enabling evidence aggregation across language boundaries. Ren et al. [57] similarly explore hybrid sparse–dense retrieval strategies for self-supervised multilingual retrieval. Beyond multilingual transfer, recent studies explicitly target extremely low-resource languages, where both annotated QA data and auxiliary linguistic resources are scarce. Pal et al. [52] demonstrate that structured data sources such as tables can partially compensate for the lack of unstructured corpora in low-resource Indic languages. Gaim et al. [15] construct the first native QA dataset for Tigrinya and evaluate monolingual, multilingual, and machine-translated ("silver") training regimes, highlighting the limitations of cross-lingual transfer when linguistic and cultural gaps are large. Similarly, Taffa et al. [64] introduce Amh-QuAD, the first publicly available QA benchmark for Amharic, enabling systematic evaluation for this language. Importantly, most existing multilingual and low-resource QA datasets and systems focus on single-hop question answering, where the answer can be derived from a single passage. For instance, Nguyen et al. [44] propose UIT-ViQuAD, a Vietnamese single-hop QA dataset inspired by SQuAD, while domain-specific resources such as ViMedAQA [68] and VLQA [45] address medical and legal QA, respectively. While these datasets are essential for establishing baseline QA capability, they do not explicitly evaluate multi-hop reasoning, which requires aggregating evidence across multiple documents, passages, or modalities. As a result, multi-hop QA in multilingual and low-resource settings remains largely underexplored. The challenges of multi-hop reasoning, such as effective evidence retrieval, relational reasoning, and error propagation across hops, are exacerbated by sparse corpora, limited hyperlink or entity graphs, and weaker cross-lingual alignment. Although the high-level QA architecture (retriever followed by reader or generator) remains similar, low-resource multi-hop QA places substantially greater demands on retrieval quality, relational knowledge modeling, and robustness to noise. This gap motivates the need for approaches that can support multi-hop reasoning under multilingual and low-resource constraints, particularly by leveraging structured relations, cross-lingual transfer, or alternative sources of relational evidence.

Chapter 3

Compact Model Size with Language-aware Compression

We proposed LangCompress, which is a language-aware framework designed to enhance the efficiency of existing model compression techniques such as structured pruning, semi-structured pruning, or quantization (referred to here as the backbone method). The framework comprises two complementary components: **instruction data synthesis** and **vocabulary simplification**. At the outset of compression, the pretrained LLM generates synthetic instruction data in the target language. This dataset is subsequently employed during the calibration and recovery fine-tuning of the backbone method, ensuring effective compression even in low-resource language scenarios. In parallel, a vocabulary analysis identifies a compact set of key tokens that cover the majority of the target language's functional lexicon. Following the compression process, the language modeling head is streamlined by retaining only the parameters corresponding to these key tokens. This modification not only reduces the overall model size but also improves its ability to specialize in the target language, yielding a compact yet linguistically focused model.

3.1 Prelimaries

Large Language Models and Vocabulary. Let $\mathcal{M}$ denote a pretrained LLM with a discrete vocabulary $\mathcal{V}$ of size $|\mathcal{V}|$. The model consists of multiple transformer decoder layers that process contextual token embeddings in a d-dimensional hidden space. Each layer captures long-range dependencies through self-attention and feed-forward mechanisms. The final hidden representation is passed to a language modeling (LM) head, parameterized by a weight matrix $\mathbf{W}_{\text{LM}} \in \mathbb{R}^{|\mathcal{V}| \times d}$, which projects the hidden states into a logit vector $\mathbf{l} \in \mathbb{R}^{|\mathcal{V}|}$ representing the predicted probability distribution over all vocabulary tokens.

The vocabulary $\mathcal{V}$ defines the model's linguistic granularity through a tokenizer, which establishes a bijective mapping between tokens and integer indices. During text generation, $\mathcal{M}$ iteratively samples from $\mathbf{l}$ to select the next token ID, which is then converted back to its textual representation. Vocabulary choice strongly influences multilingual performance; languages with rich morphology or complex scripts often suffer from suboptimal token coverage, reducing modeling quality.

Pruning. Formally, given a weight matrix $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ with elements $w \in \mathbb{R}$, pruning applies a sparsification operation by element-wise masking:

$$\hat{w} = m \cdot w,$$

where $m \in 0, 1$ is a binary pruning mask indicating whether the weight is retained ($m = 1$) or removed ($m = 0$). Equivalently, the pruned weight matrix can be expressed as

$$\hat{\mathbf{W}} = \mathbf{M} \odot \mathbf{W},$$

where $\mathbf{M} \in 0, 1^{d_{\text{out}} \times d_{\text{in}}}$ and $\odot$ denotes element-wise multiplication. The pruning mask $\mathbf{M}$ is typically constructed by ranking weights according to an importance criterion, such as magnitude or an approximation of second-order loss sensitivity, and retaining only the top-k weights under a predefined sparsity constraint.

Unstructured Pruning. Pruning aims to compress neural networks by eliminating redundant or less important parameters, thereby reducing computational and memory demands. Unstructured pruning operates at the individual weight level by setting selected parameters in the model's weight matrices to zero, resulting in fine-grained sparsity patterns. This approach provides high flexibility and precise control over sparsity distribution; however, unless the underlying hardware and software stack explicitly support sparse matrix multiplication, unstructured sparsity typically does not translate into inference speedups, as zero-valued weights are still processed by dense kernels.

Semi-Structured Pruning. Semi-structured pruning introduces regularity constraints on sparsity patterns to improve hardware efficiency while retaining some flexibility. A widely adopted formulation is the N:M sparsity pattern, where at least N out of every M consecutive weights within a tensor block are pruned. This predictable structure enables optimized GPU kernels to skip zero-valued weights, resulting in tangible inference speedups on modern hardware. Methods (e.g., SparseGPT [12]) estimate block-wise weight importance using local Hessian approximations, allowing effective sparsification with minimal accuracy loss. By balancing expressiveness and computational efficiency, semi-structured pruning provides a practical compromise between unstructured sparsity and fully structured model compression.

Structured Pruning. Structured pruning removes entire architectural components (e.g., attention heads, feed-forward network units, or even whole layers) rather than individual weights [5, 39]. By eliminating complete submodules, structured pruning preserves dense matrix operations and is therefore inherently compatible with existing hardware acceleration. Typically, structured pruning involves two stages: (1) identifying low-importance components using calibration data, and (2) recovering performance through task-specific fine-tuning.

In multilingual settings, pruning introduces additional challenges: pruning decisions made using calibration data dominated by high-resource languages can disproportionately remove components critical for low-resource languages. Consequently, multilingual-aware calibration or balanced data selection is often required to maintain cross-lingual robustness.

Quantization. Quantization is a widely adopted model compression technique that reduces the computational and memory cost of deep neural networks by representing weights and activations using reduced numerical precision. Instead of storing model parameters as high-precision floating-point values such as FP32 or FP16, quantization maps them into lower-bit integer formats (e.g., INT8, INT4) or even binary values in highly compressed systems. This reduction directly decreases model size and enables faster inference, especially on hardware optimized for integer-based operations such as tensor accelerators and edge NPUs. As a result, quantization is an essential component for deploying large-scale models in resource-constrained environments, including real-time applications or mobile devices.

Formally, for a weight matrix $\mathbf{W}$ with elements $w \in \mathbb{R}$, quantization applies a transformation such that each weight is mapped into a discrete set of integer values:

$$\hat{w} = \text{round}\left(\frac{w}{s}\right) + z,$$

where s denotes a scaling factor and z is an integer zero-point. The role of the zero-point is to shift the quantized value range to better approximate asymmetric real-valued distributions. During inference, an approximate value $\tilde{w}$ is reconstructed through a corresponding dequantization process:

$$\tilde{w} = s \cdot (\hat{w} - z).$$

The discrepancy between w and $\tilde{w}$ introduces quantization noise; maintaining this error within acceptable limits is crucial to preserving model accuracy.

Different variations of quantization exist based on how scaling parameters are shared. Per-tensor quantization uses a single scale and zero-point for the entire tensor, which is efficient but can be suboptimal for matrices with large inter-channel variance. Per-channel and per-group quantization assign separate scaling values to individual structural units (e.g., output channels of linear layers), reducing quantization error at the cost of additional metadata storage. The choice of granularity, therefore, reflects a trade-off between efficiency and numerical fidelity. Quantization can be applied either during or after training. Post-training quantization (PTQ) converts weights to reduced precision after model convergence without modifying the training procedure. PTQ is attractive due to its simplicity and low computational cost, particularly for LLMs with billions of parameters. In contrast, quantization-aware training (QAT) incorporates simulated quantization effects into the forward pass during fine-tuning, enabling the model to adapt to discretization noise. QAT generally achieves higher accuracy, though its compute and memory requirements make it less feasible at a massive scale.

In multilingual and low-resource language settings, quantization poses additional difficulties. Quantized models' performance can strongly depend on the representativeness of calibration data, which is used to estimate scaling factors. Calibration data typically reflects common, high-resource language patterns, yet token distributions in low-resource languages may differ significantly. This mismatch can lead to reduced token prediction accuracy and inconsistent quality across languages after quantization.

3.2 Data Synthesis for Target Language

Figure 3.1 illustrates the proposed instruction data synthesis pipeline for a target language, shown here using Japanese as an example. Prior research [72] has demonstrated that, due to the autoregressive nature of LLMs, instruction data can be automatically generated by providing only pre-query templates up to the position designated for user input, as long as a suitable system prompt is specified. In principle, this allows an LLM to autonomously infer and generate coherent instruction–response pairs without requiring explicit supervision for each example. However, this generative capability does

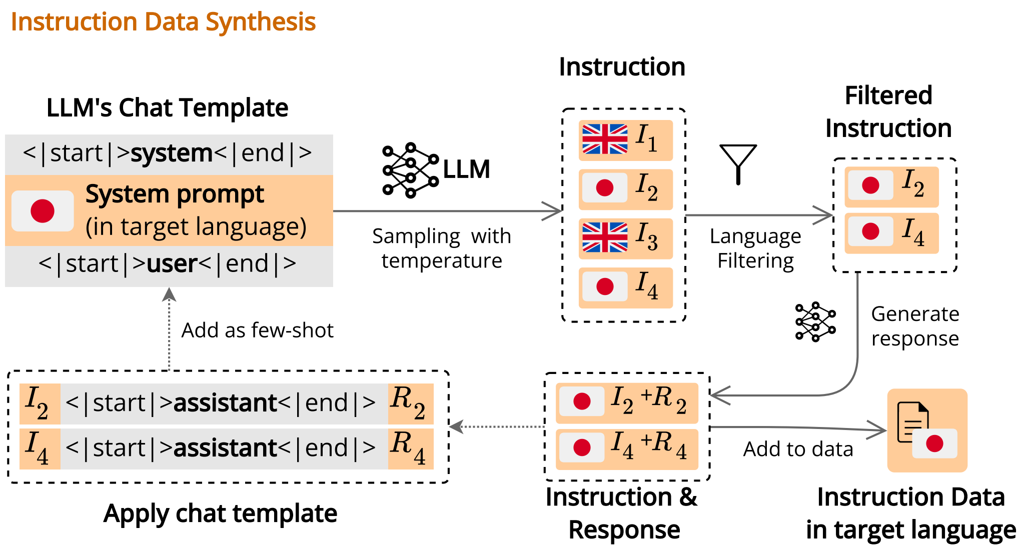


Figure 3.1: Instruction data synthesis pipeline in the target language, with an example of Japanese language.

not always guarantee linguistic consistency. In multilingual models, especially those trained on unbalanced corpora, the model may spontaneously revert to dominant languages such as English, even when the prompt is written in the target language. This issue becomes particularly pronounced for low-resource languages that receive limited exposure during pretraining, resulting in inconsistent, code-mixed, or semantically incoherent outputs.

To overcome these limitations, we introduce an iterative few-shot enhancement pipeline that progressively aligns the model's output distribution toward the target linguistic domain. The process begins by preparing a small set of high-quality instruction–response exemplars written in the target language. These examples are embedded into a chat-style prompt using a predefined template that emulates natural human–machine dialogue. The LLM is then prompted to generate additional instruction data conditioned on this augmented context. In each iteration, a new batch of candidate instructions is sampled, validated, and filtered for language consistency before being appended to the few-shot context. As the process repeats, the model gradually internalizes the linguistic characteristics of the target language, improving its ability to generate consistent and well-structured instructions. This iterative bootstrapping continues until the probability of producing target-language instructions converges to a stable and satisfactory level, ensuring both coverage and reliability in the synthesized dataset.

Algorithm 1 formalizes the synthesis procedure. The process begins with

Algorithm 1 Data Synthesis for Target Language

Require: Target language $\mathcal{L}$
Require: System prompt in target language $\mathcal{S}_{\mathcal{L}}$
Require: Original LLM $\mathcal{M}$
Require: Language filter function f_{filter}
Require: Chat template function f_{template}
Require: Maximum few-shot examples K
Require: Number of examples to generate N

Initialize dataset $\mathcal{D} \leftarrow \emptyset$
Initialize few-shot counter $k \leftarrow 0$
Initialize prompt $p \leftarrow f_{\text{template}}(\mathcal{S}_{\mathcal{L}})$
while $|\mathcal{D}| < N$ **do**
    Sample instructions $\mathbf{I} \leftarrow \mathcal{M}(p, \text{temp} = 1.0)$
    Filter instructions $\mathbf{I} \leftarrow f_{\text{filter}}(\mathbf{I}, \text{target} = \mathcal{L})$
    for $i = 1$ to $|\mathbf{I}|$ **do**
        $\mathcal{R}_i \leftarrow \mathcal{M}(\mathbf{I}_i)$ ▷ Generate response
        if $k < K$ **then** ▷ Append few-shot example
            $p \leftarrow p \oplus f_{\text{template}}(\mathbf{I}_i, \mathcal{R}_i)$
            $k \leftarrow k + 1$
        end if
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{I}_i, \mathcal{R}_i)\}$
    end for
end while
return $\mathcal{D}$

the initialization of a system prompt crafted in the target language. This prompt serves as the global instruction that defines the linguistic context for all subsequent generations. The system prompt is paired with a chat template that mimics multi-turn interactions, providing structural cues that encourage the model to produce realistic and contextually grounded dialogues. This combination ensures that the generated instructions resemble authentic user queries and assistant responses, which are essential for downstream fine-tuning. The LLM then samples a batch of candidate instructions using a temperature-based decoding strategy. This stochastic decoding mechanism introduces a controlled degree of randomness into the token sampling process, balancing creativity and diversity with coherence and grammaticality. By adjusting the temperature parameter, the generation process can explore a broader range of linguistic expressions while avoiding deterministic repetition or trivial outputs. However, due to the inherently multilingual nature of modern pretrained LLMs, the resulting samples often contain instructions in multiple languages. Such cross-lingual interference arises from the uneven representation of languages in pretraining data and the model's tendency to generalize across overlapping lexical and syntactic patterns.

To enforce linguistic purity and domain fidelity, we apply a probabilistic N-gram language filter [54] to all generated candidates. This filter computes the likelihood that each candidate belongs to the target linguistic domain by comparing its N-gram frequency distribution with that of a reference corpus in the target language. Samples whose probability falls below a threshold are discarded, while high-confidence instructions are retained for further processing. This filtering mechanism plays a crucial role in isolating clean, monolingual instruction data and preventing contamination from dominant or unrelated languages. The resulting corpus thus remains both linguistically consistent and semantically diverse.

After filtering, the LLM generates responses corresponding to each retained instruction, forming coherent instruction–response pairs that encapsulate both pragmatic and linguistic features of human dialogue. These pairs constitute the fundamental units of the synthetic dataset. In addition to being stored for downstream use, selected pairs are reintegrated into the chat template as new few-shot exemplars, creating a self-reinforcing feedback loop. Over successive iterations, this loop enables the model to refine its internal representation of the target language and reduce its reliance on dominant language priors. Empirically, we find that this self-improvement mechanism enhances both the fluency and naturalness of the generated text, particularly in low-resource language settings.

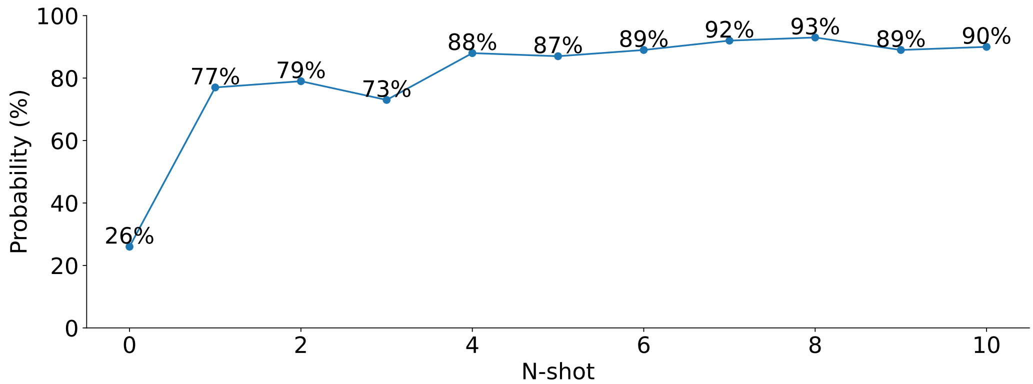


Figure 3.2: Relationship between the number of few-shot Japanese examples in the prompt and the probability of Japanese instruction generation for Llama3-8B-Instruct.

Figure 3.2 and 3.3 present the empirical relationship between the number of few-shot examples and the probability of generating instructions in the

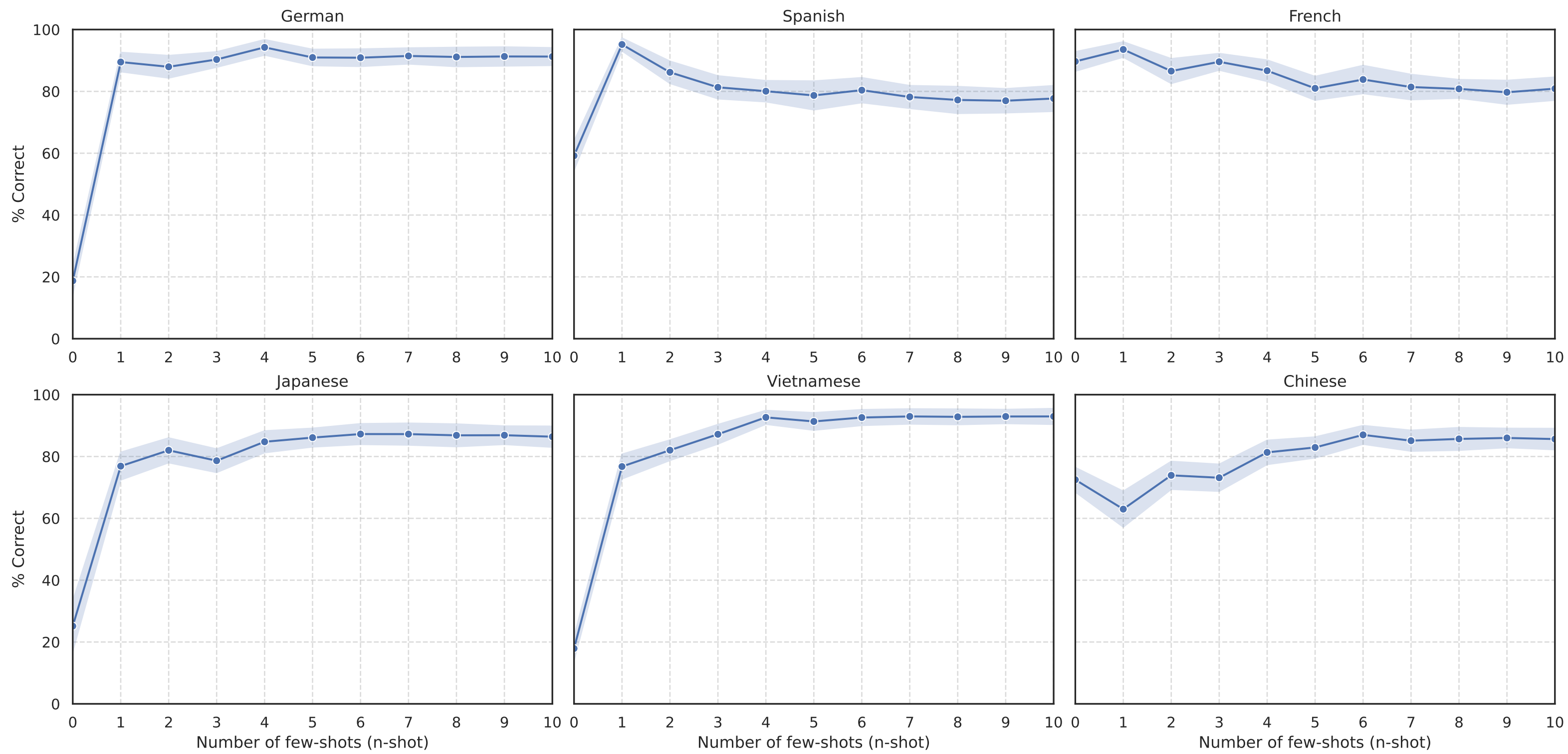


Figure 3.3: Relationship between the number of language-specific few-shot examples in the prompt and the probability of language-specific instruction for Llama3-8B-Instruct.

target language. The probability increases rapidly as more examples are added, before reaching a plateau at a stable generation rate. In practice, we observe that approximately ten high-quality few-shot examples are sufficient to achieve consistent generation across most target languages. Beyond this point, additional examples yield diminishing returns. This finding highlights the efficiency of the iterative enhancement approach, which achieves strong language fidelity with minimal manual supervision.

In addition, to ensure that only high-quality instruction data are selected, we apply quality control filters based on estimated instruction quality. Specifically, we retain only generated samples labeled as "good" or "excellent" in quality, discarding low-quality instructions. This filtering strategy, adopted from previous work [72], is not a novel contribution but serves to maintain a high standard of instructional clarity in the synthesized data.

Overall, the proposed synthesis pipeline integrates three key components, namely stochastic generation, language-aware filtering, and iterative few-shot reinforcement, to construct high-quality and language-consistent instruction datasets. This framework effectively harnesses the generative potential of large language models while addressing the inherent imbalance of multilingual pretraining. The resulting data not only supports robust fine-tuning for underrepresented languages but also provides a foundation for downstream model compression, recovery, and calibration within the LangCompress framework. Examples of the generated data and the corresponding generation prompts are provided in Appendix 7.

3.3 Vocabulary Simplification

Figure 3.4 and Algorithm 2 illustrate the vocabulary analysis and simplification procedure used to identify key tokens for reducing the size of the language model (LM) head. This component aims to improve model efficiency by focusing on the most frequently used linguistic units in the target language while maintaining high coverage of the input space.

The process begins with a large-scale raw corpus in the target language, such as Wikipedia, multilingual C4, or FineWeb. The corpus is first tokenized using the model's native tokenizer to ensure consistency with the pretrained vocabulary. We then compute the frequency distribution of all tokens and sort them in descending order of occurrence. The top-k most frequent tokens are selected as key tokens, representing the subset of vocabulary that captures the dominant linguistic patterns of the target language. This frequency-based selection strategy aligns with Zipf's law, which states that natural language exhibits a highly skewed token distribution, where a small proportion of

Algorithm 2 Key Token for Target Language

Require: Target language $\mathcal{L}$
Require: Original LLM $\mathcal{M}$
Require: Tokenizer $\mathcal{F}_{\text{token}}$ of $\mathcal{M}$
Require: Raw corpus $\mathcal{C}_{\mathcal{L}}$ in language $\mathcal{L}$
Require: Number of desired key tokens k
1: Tokenize corpus: $\mathbf{T} \leftarrow \mathcal{F}_{\text{token}}(\mathcal{C}_{\mathcal{L}})$
2: Initialize frequency map: $\mathcal{F} \leftarrow \emptyset$
3: **for** each token $t \in \mathbf{T}$ **do**
4: $\quad\mathcal{F}[t] \leftarrow \mathcal{F}[t] + 1$
5: **end for**
6: Sort tokens by frequency: $\mathbf{S} \leftarrow \text{SortDescending}(\mathcal{F})$
7: Select top-k tokens: $\mathcal{V}_{\text{simplify}} \leftarrow \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_k\}$
8: **return** $\mathcal{V}_{\text{simplify}}$

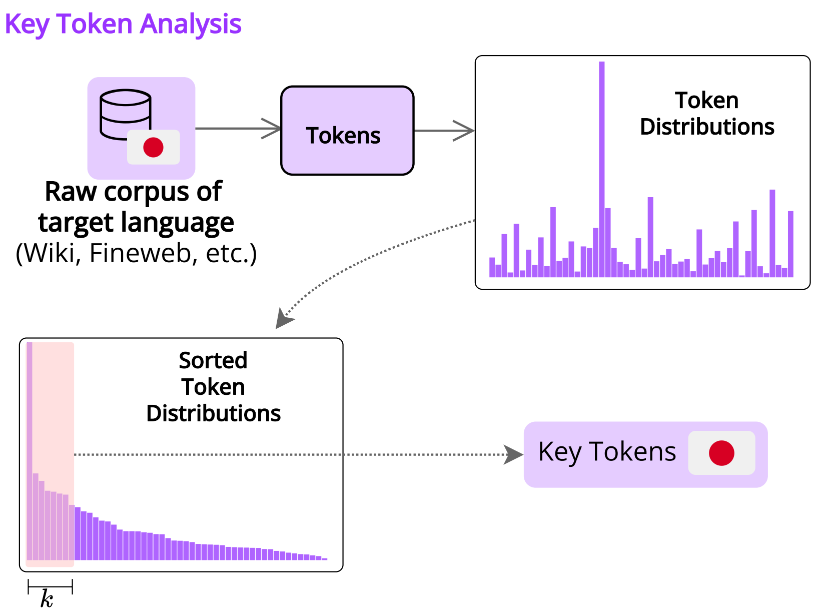


Figure 3.4: Instruction data synthesis pipeline in the target language, with an example of Japanese language.

words accounts for the vast majority of textual content.

Empirical evidence supports this principle. As shown in Figure 3.5, the top 5% of tokens in the FineWeb corpus collectively cover more than 95% of all token occurrences. This finding implies that a significant degree of lexical redundancy exists in the long tail of the vocabulary, providing an opportunity

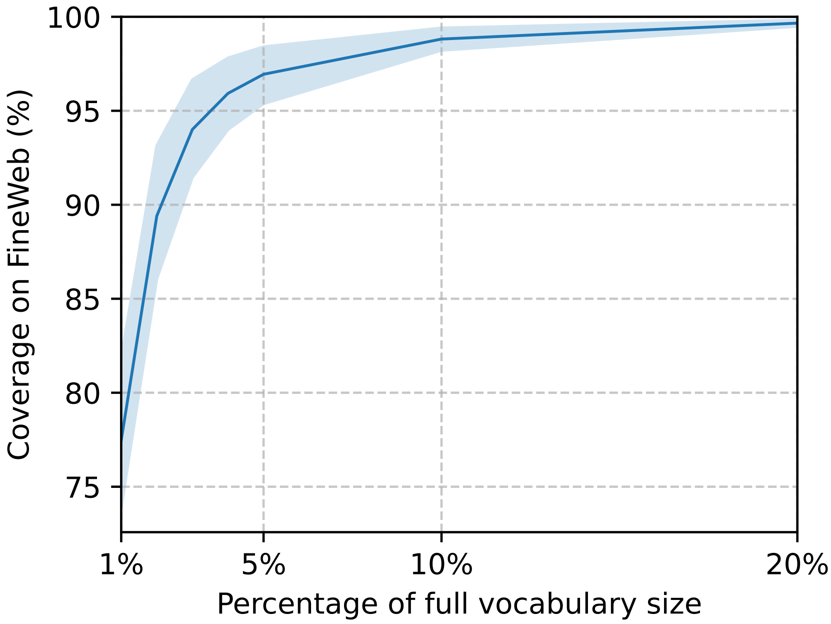


Figure 3.5: Coverage on FineWeb of the top 1% to 20% highest-frequency vocabulary tokens, averaged across six languages: German, Spanish, French, Japanese, Chinese, and Vietnamese.

for compression without substantial loss in representational capacity. Based on this observation, we simplify the LM head by retaining only the rows of the output projection matrix corresponding to these key tokens. In practice, this reduces the effective parameter count and memory footprint of the LM head while preserving most of the model's predictive capability for common linguistic patterns. Additionally, we measure token coverage across multiple languages using both the FineWeb2 dataset and the Llama3 tokenizer with a 128K-token reference vocabulary. Token coverage is defined as the proportion of corpus tokens that can be represented using the reduced vocabulary. Table 3.1 summarizes these results.

Vocab Size	**English**	**Vietnamese**	**Chinese**	**Japanese**
8K (6%)	82.79	98.94	99.37	99.39
16K (13%)	89.58	99.55	99.79	99.76
32K (25%)	95.34	99.88	99.94	99.92
64K (50%)	99.33	99.99	99.99	99.99

Table 3.1: Token coverage (%) by vocabulary size for each language, evaluated on Fineweb2 with Llama3 (128K-token vocabulary).

In summary, the proposed vocabulary simplification strategy leverages frequency-based key token extraction to retain the most informative subset of tokens in the LM head. This approach capitalizes on the natural redundancy of linguistic distributions, allowing the model to achieve memory-efficient inference and fine-tuning without significant loss in representational accuracy. The resulting architecture provides a flexible mechanism for adapting large multilingual models to specific target languages, especially under resource-constrained deployment scenarios.

3.4 LangCompress's Compression

With both the target-language instruction dataset $\mathcal{D}$ and the selected key token set $\mathcal{V}_{\text{simplify}}$, the compression phase of LangCompress is carried out using a chosen backbone method (i.e., pruning or quantization). This stage integrates vocabulary simplification with model compression to achieve substantial parameter and memory reductions while maintaining performance in the target language.

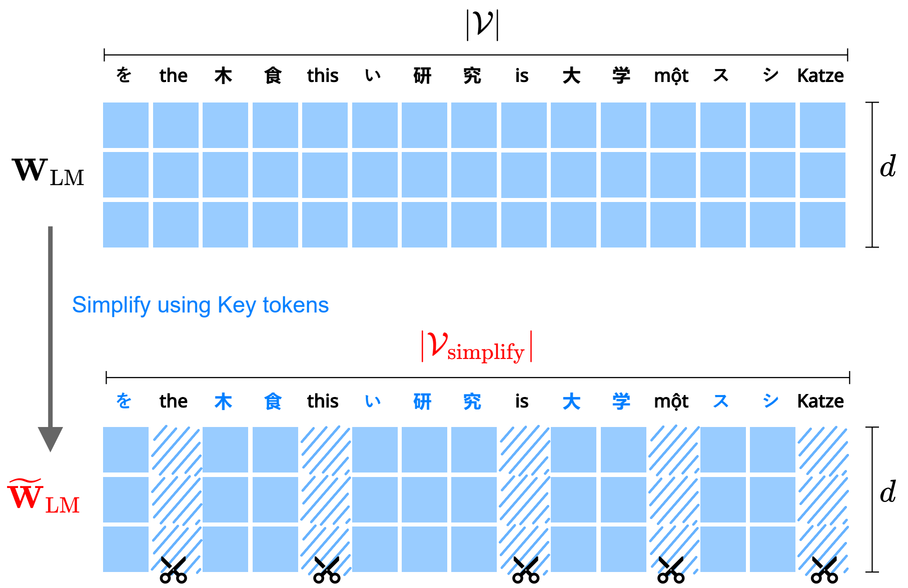


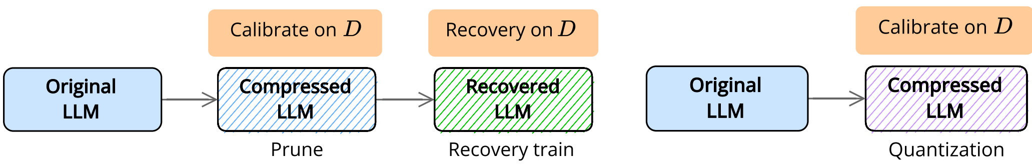
Figure 3.6: LM Head Simplification process.

LM Head Simplification. To simplify the LM head, we adopt a vocabulary-aware transformation inspired by prior work on frequency-based vocabulary reduction [77]. The original LM head is represented as a weight matrix

$\mathbf{W}_{\text{LM}} \in \mathbb{R}^{|\mathcal{V}| \times d}$, where $|\mathcal{V}|$ denotes the size of the full vocabulary and d is the hidden dimension. We construct a reduced LM head $\widetilde{\mathbf{W}}_{\text{LM}} \in \mathbb{R}^{|\mathcal{V}_{\text{simplify}}| \times d}$ by retaining only the rows corresponding to the most frequent and informative tokens identified during vocabulary analysis:

$$\widetilde{\mathbf{W}}_{\text{LM}}[i,:] = \mathbf{W}_{\text{LM}}\big[\mathcal{V}_{\text{simplify}}[i],:\big], \quad i = 1,\ldots,|\mathcal{V}_{\text{simplify}}|$$

Figure 3.6 provides a schematic overview of the vocabulary-aware simplification of the LM head. This operation effectively prunes the output projection layer to focus exclusively on the high-coverage subset of tokens $\mathcal{V}_{\text{simplify}}$, which accounts for the majority of token occurrences in the target language. By replacing $\mathbf{W}_{\text{LM}}$ with its simplified counterpart $\widetilde{\mathbf{W}}_{\text{LM}}$, we achieve a smaller and more efficient LM head that retains linguistic expressiveness for the target domain.



(a) Pruning-based (b) Quantization-based

Figure 3.7: Applying instruction dataset $\mathcal{D}$ to Compression Methods

Backbone Compression and Adaptation. After simplifying the LM head, the next step involves compressing the backbone network using the selected compression technique. This process is guided by an instruction dataset $\mathcal{D}$, which is synthesized specifically for the target language to ensure that compression remains sensitive to linguistic nuances. The dataset plays different roles depending on the compression strategy adopted. For pruning-based methods, $\mathcal{D}$ is utilized during a recovery or fine-tuning phase following the introduction of sparsity. This recovery training allows the pruned model to regain lost performance and to realign its internal representations with the linguistic and semantic characteristics encoded in the target-language instructions. By exposing the model to $\mathcal{D}$ during this stage, the pruned parameters are adapted to preserve both task performance and language-specific fidelity. In contrast, when applying quantization-based compression, $\mathcal{D}$ functions as a calibration dataset used to estimate the activation and weight distributions within the model. These statistics are critical for determining optimal quantization scales and zero-points, thereby minimizing precision loss and maintaining output quality across different numerical bit widths. Proper calibration ensures that quantization does not disproportionately affect features that are linguistically salient in the target language. In both cases, the instruction dataset provides essential contextual grounding that enhances the robustness of the compressed model.

Overall, LangCompress integrates instruction-driven synthesis, vocabulary reduction, and backbone compression into a unified framework. By jointly leveraging language-specific vocabulary simplification and data-guided compression, the resulting model attains a compact yet linguistically expressive representation optimized for efficient inference in the target language. This design facilitates the effective adaptation of large multilingual models to low-resource languages while preserving their generative capacity and representational strength. Consequently, the compressed model strikes a favorable balance among computational efficiency, linguistic fidelity, and downstream task performance.

Chapter 4

Low-resource Language Multi-hop Retrieval

4.1 Multilingual multihop dataset creation

4.1.1 Data creation framework

We propose a comprehensive framework for constructing multilingual, multi-hop question–answering (QA) datasets. The framework is designed to generate high-quality and explainable QA samples while remaining adaptable to different languages and knowledge sources. It operates through four primary stages.

In the first stage, *data collection*, interconnected articles or documents are selected from structured knowledge sources such as Wikipedia, forming the foundation for multi-hop reasoning. Next, during *question creation*, human annotators compose questions that require reasoning across multiple contexts, ensuring that each question is both challenging and answerable. The third stage, *answer and supporting facts annotation*, involves identifying the correct answer and marking the specific sentences that justify it, thereby enhancing interpretability and traceability. Finally, in the *validation and normalization* stage, all collected samples are reviewed for accuracy, consistency, and linguistic quality, followed by normalization procedures to ensure compatibility across languages.

Through this structured process, the proposed framework produces datasets that facilitate the evaluation and development of advanced QA systems capable of performing multi-hop reasoning and generating transparent, explainable predictions.

Data Collection Pipeline

The data collection process begins by defining a set of suitable source documents, which are filtered to ensure they are amenable to multi-hop question creation. The selected documents are treated as nodes in a directed graph, with edges representing relationships or hyperlinks. From this graph, pairs of related paragraphs are sampled to serve as the context for multi-hop questions. For comparison-type questions, similar entities (e.g., musicians, scientists, or organizations) are grouped, and paragraph pairs are drawn from the same category. This allows questions to require reasoning that compares or connects multiple entities. Figure 4.1 illustrates the data collection pipeline.

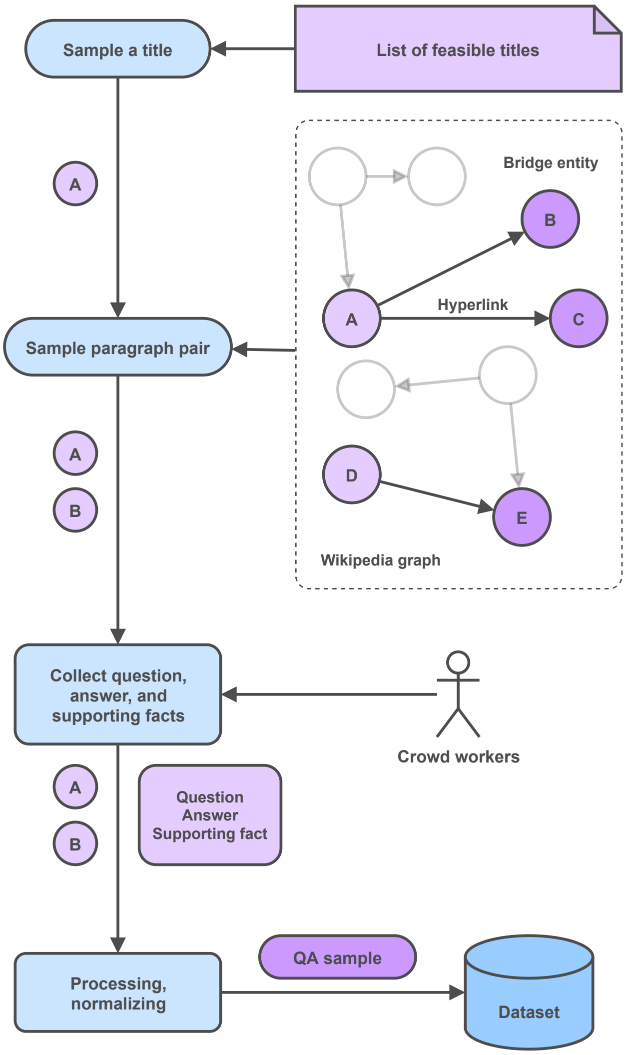


Figure 4.1: Overall data collecting pipeline

Annotation Process

Annotators are provided with the paragraph pairs and instructed to create multi-hop questions. For each question, they supply the correct answer along with supporting sentences. Figure 4.2 shows the user-friendly annotation interface, which is designed to guide workers and enforce rules, thereby minimizing human error.

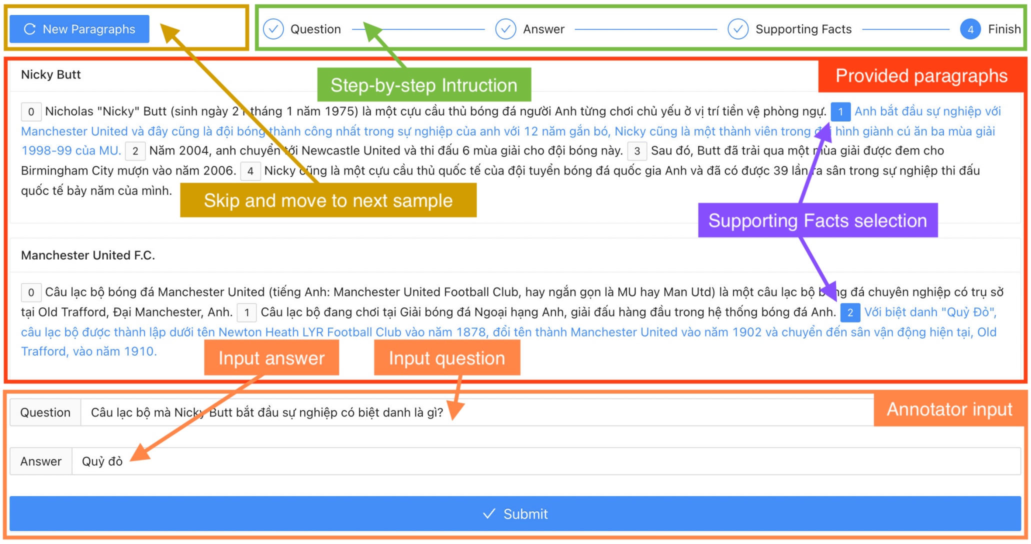


Figure 4.2: Annotator interface for entering a sample

Processing and Normalization

After annotation, the data undergoes post-processing to clean, standardize, and normalize language-specific elements. While the framework includes tools for certain languages, it is fully configurable to accommodate the unique characteristics of any language with minimal modifications. This methodology establishes a flexible, systematic approach to building high-quality, multilingual datasets for multi-hop reasoning, supporting both research and system development in QA.

4.1.2 VIMQA: Vietnamese Multi-hop QA Dataset

VIMQA is a large-scale Vietnamese dataset designed to facilitate research on QA systems that require multi-step reasoning and explainable predictions.

It was constructed using the multilingual multi-hop dataset framework proposed in this thesis, which enables the systematic collection of QA pairs from any structured knowledge source. VIMQA addresses a notable gap in Vietnamese QA resources by providing over 10,000 high-quality question-answer pairs based on Wikipedia articles, each requiring reasoning across multiple contexts.

Dataset Features

VIMQA exhibits several distinctive characteristics that make it a valuable benchmark for evaluating multi-hop question-answering systems. Each question in the dataset is designed to require the integration of information from two or more paragraphs, thereby assessing a model's capacity for complex reasoning across linked knowledge sources. In addition, every QA pair is accompanied by sentence-level supporting facts, enabling the development and evaluation of models capable of transparent and interpretable reasoning.

The dataset encompasses a diverse range of reasoning types, including temporal, numerical, and causal reasoning, as well as entity comparison and relational inference. As a Vietnamese dataset, VIMQA also presents language-specific challenges such as word segmentation and diacritic handling, which are rarely encountered in high-resource languages. Furthermore, with more than 10,000 QA pairs collected from a broad selection of Wikipedia articles, the dataset offers extensive topical coverage across domains such as history, notable figures, and technical concepts. Collectively, these characteristics establish VIMQA as a comprehensive and challenging benchmark for advancing multilingual and explainable multi-hop QA research.

Construction Using the Proposed Framework

The creation of VIMQA demonstrates the effectiveness of the proposed multilingual multi-hop framework. Wikipedia articles were treated as nodes in a directed graph, with hyperlinks representing edges. Pairs of connected paragraphs were sampled as the basis for multi-hop questions. Human annotators generated questions, identified answers, and marked supporting sentences, following the pipeline described in the methodology section. Post-processing and normalization ensured consistency and language-specific correctness. This approach not only allowed the systematic creation of VIMQA but also establishes a template for constructing similar datasets in other languages.

Benchmarking and Impact

Initial experiments on VIMQA with state-of-the-art multilingual QA models show that existing methods often struggle to perform accurate multi-hop reasoning in Vietnamese. This confirms the dataset's complexity and highlights the need for improved QA architectures for low-resource languages. By providing both challenging questions and explicit supporting facts, VIMQA supports research on interpretable and robust QA systems. Detailed experimental settings and quantitative results have been thoroughly reported in our prior work on VIMQA [30], and are therefore not repeated here.

Availability

The VIMQA dataset [30], initially presented at the LREC conference, has been made publicly accessible to support scholarly inquiry. The complete dataset, along with its accompanying multilingual framework, is hosted and available for download from its official repository: `https://github.com/vimqa/vimqa`. Descriptive statistics detailing the dataset's composition are provided in Table 4.1. Sample VIMQA instances are presented in Appendix 7.

Split	**Number of Questions**
Train	8,041
Development	1,003
Test	1,003
Total	10,047

Table 4.1: Statistics of the VIMQA dataset.

Overall, VIMQA exemplifies how a structured, language-agnostic framework can produce a high-quality, explainable QA dataset, advancing research in multi-hop reasoning for Vietnamese and potentially other underrepresented languages.

4.2 Hyperlink-based retrieval system

We propose a retrieval framework, which introduces a hyperlink-based retrieval mechanism for efficient multi-hop question answering. Although the system was initially developed for Vietnamese, its architecture and retrieval logic are language-agnostic, allowing it to be adapted to other languages that provide Wikipedia or similar hyperlinked corpora. This design enables cross-lingual scalability and facilitates the development of multilingual question answering systems.

4.2.1 Prelimaries

Problem Definition. We address the open-domain QA task as follows. Given a natural language question q, the goal is to find the correct answer by querying a knowledge collection $\mathcal{C}$. This collection contains c candidate passages $P_1, \ldots, P_c$, where each passage P_i consists of tokens $p_i^{(1)}, \ldots, p_i^{(l_i)}$. For multi-hop questions, the system must identify and reason over multiple related passages (e.g., P_i and P_j) to produce the final answer.

Notation Summary Table 4.2 lists the symbols used in this work along with their concise definitions.

Symbol	Definition
[CLS]	Token representing the full input; its hidden state is used for classification/regression.
[SEP]	Segment separator token (e.g., between question and passage).
Tok_i	i-th token in a tokenized question or passage.
E_i	Input embedding for Tok_i.
C	Final hidden state of the *[CLS]* token.
T_i	Hidden state of the i-th question token after encoding.
T'_i	Hidden state of the i-th passage token after encoding.
P_i	i-th candidate passage retrieved for the question.
a_i	i-th predicted answer by the reader model.
R_i	Passage included in the final retrieval results.
Q_i	Passage retrieved at the second hop in multi-hop reasoning.
A_i	Article in the knowledge collection.
$A_{i,j}$	j-th article linked to A_i via hyperlinks.
$P_{i,j,k}$	k-th passage within article $A_{i,j}$.

Table 4.2: Description of the notations used in the study.

Transformer Architecture

The Transformer architecture [69] forms the foundation of most modern language models. It replaces recurrent operations with a self-attention mechanism, allowing each token in a sequence to directly attend to all other tokens. This mechanism enables efficient modeling of long-range dependencies and supports parallel computation.

The core component of the Transformer is the Scaled Dot-Product Attention, which computes the weighted representation of a sequence based on query (Q), key (K), and value (V) matrices as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V,$$

To capture diverse contextual relationships, multiple attention heads are applied in parallel, forming the Multi-Head Attention mechanism:

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O, \\ \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \\ \text{where } W_i^Q, W_i^K, W_i^V, W^O & \text{ are are learnable parameter matrices} \end{aligned}$$

This attention mechanism is used in both encoder and decoder layers, enabling self-attention within a sequence and cross-attention between encoder and decoder representations. In addition, the Transformer incorporates position-wise feed-forward networks, residual connections, and layer normalization to enhance representational capacity and stabilize training.

4.2.2 Construction of the Wikipedia Knowledge Base

The knowledge base was constructed using the Vietnamese Wikipedia dump (January 20, 2022). Following standard preprocessing procedures, inspired by Chen et al. [9], the WikiExtractor tool was used to remove semi-structured content such as tables and lists, preserving only the textual segments of each article. The text of each article was divided into overlapping passages using a sliding window of 100 tokens with a stride of 50 tokens, yielding approximately 3.9 million passages in total.

In addition, a Wikipedia hyperlink graph was built to represent the structural relationships between articles. In the graph, nodes correspond to articles, and a directed edge (u, v) exists when u links to v. Hyperlinks were extracted from HTML tags, normalized, and mapped to the corresponding articles in the corpus. This hyperlink graph serves as the relational foundation for multi-hop retrieval, enabling the system to follow semantic connections between articles. The same hyperlink-based reasoning process remains applicable across linguistic boundaries.

4.2.3 Hyperlink-based Multi-hop Retrieval

To effectively retrieve evidence for complex questions that require reasoning over multiple sources, the system integrates both textual relevance and hyperlink structure. The retrieval process proceeds in two main stages.

First-hop Retrieval: The system initiates a single-hop retrieval process (illustrated in Figure 4.3). Initially, BM25 [38] retrieves the top-m candidate passages based on lexical similarity between the question and the corpus, where m is selected to balance coverage and computational cost for downstream reranking. These candidates are then reranked using a Cross-Encoder built on XLM-RoBERTa [11], which measures the semantic relevance of each question–passage pair. Concurrently, a Reader model identifies candidate answer segments within each passage, and the top n segments with the strongest prediction scores are kept. Both the relevance scores and the extracted answer spans are subsequently used to rerank the passages. The passage with the highest combined score, denoted P_1, is chosen as the first-hop evidence, and its corresponding article, A_1, is identified for hyperlink-based expansion.

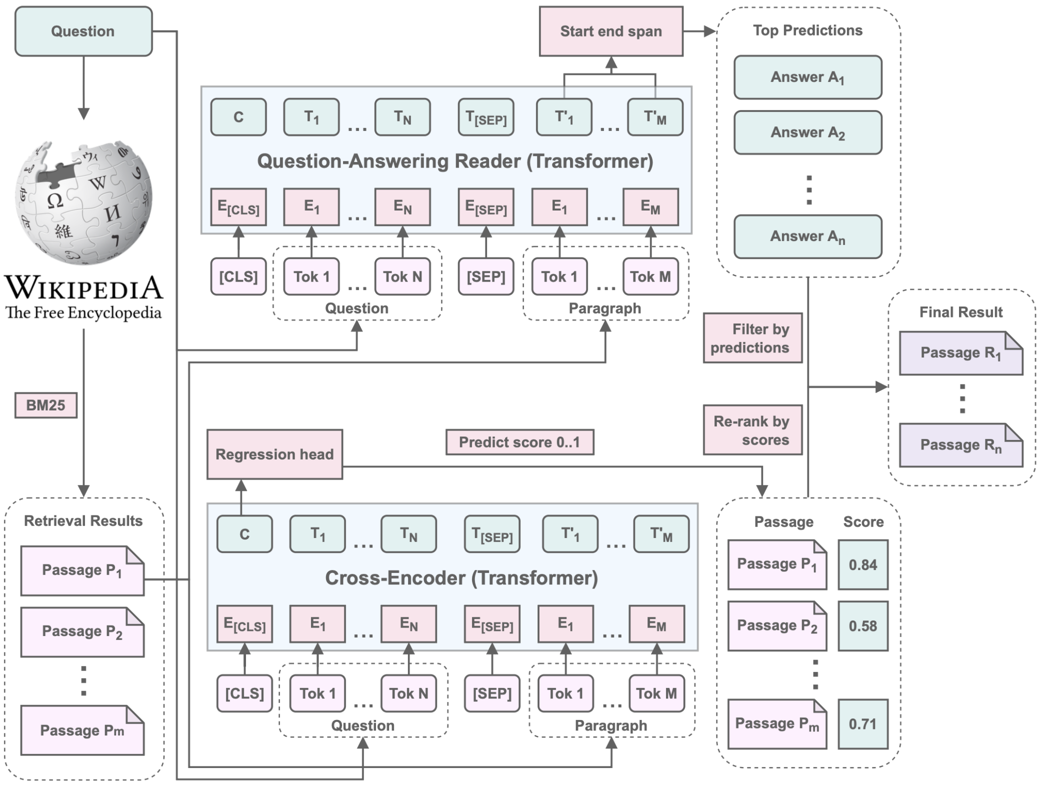


Figure 4.3: Architecture of the single-hop retriever used in first-hop retrieval

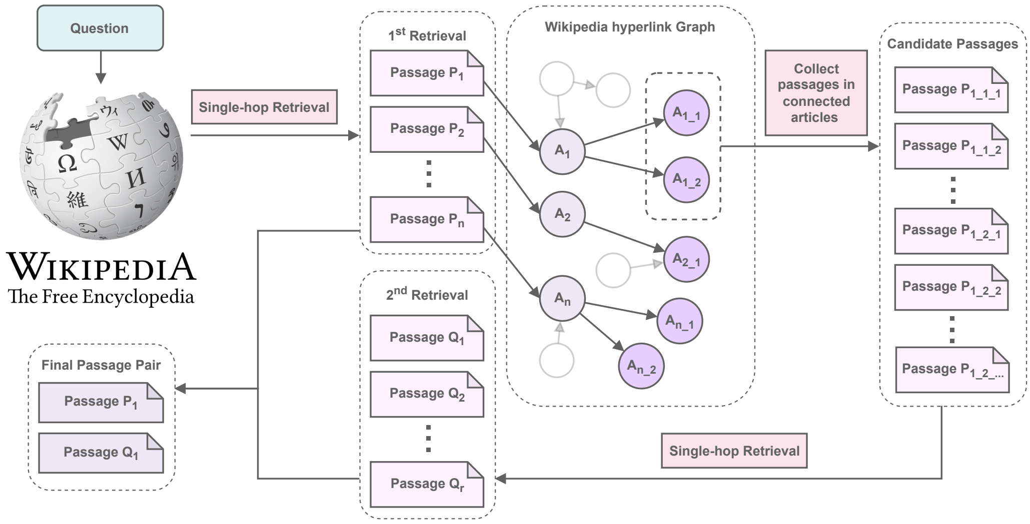


Figure 4.4: Architecture of the multi-hop retriever used for 2nd-hop retrieval

Second-hop Retrieval via Hyperlinks: Figure 4.4 illustrates how the second-hop retrieval is performed by iteratively applying the first-hop retrieval procedure. Once the first-hop article A_1 is identified, the system explores its outgoing hyperlinks to discover related articles $A_{1_1}, A_{1_2}, ..., A_{1_{c1}}$. These linked articles provide potential candidates for the second-hop evidence. Passages from the linked articles are reranked using the same Cross-Encoder model, and the top passage Q_1 is selected as the second-hop result. The resulting pair (P_1, Q_1) constitutes a multi-hop evidence pair that connects semantically related information distributed across multiple documents. This procedure can be extended to lower-ranked first-hop passages $\{P_2, \ldots, P_n\}$ and their neighboring articles in the hyperlink network, as represented below:

$$P_2 \to \{A_{2_1}, A_{2_2}, \ldots, A_{2_{c_2}}\}$$
$$P_3 \to \{A_{3_1}, A_{3_2}, \ldots, A_{3_{c_3}}\}$$
$$\cdots$$
$$P_n \to \{A_{n_1}, A_{n_2}, \ldots, A_{n_{c_n}}\}$$

This hyperlink-based expansion allows the system to go beyond surface-level textual matching, leveraging Wikipedia's structured relationships to discover contextually connected content.

4.2.4 Cross-Encoder for Relevance Scoring

The Cross-Encoder component plays a crucial role in scoring and reranking retrieved passages. Figure 4.5 illustrates the overall architecture. The multilingual capability of XLM-RoBERTa [11] makes it suitable for our work, so we employ it as the backbone encoder for our non-English language development. Because XLM-RoBERTa is a multilingual pretrained model, this retrieval mechanism can be readily adapted to other languages.

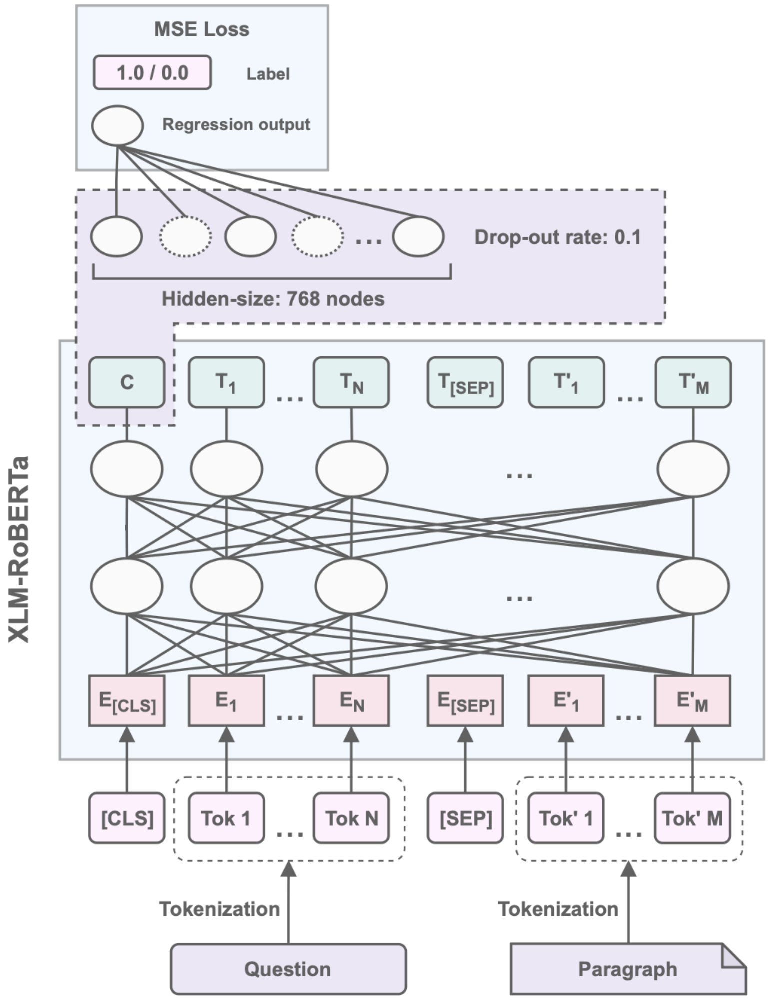


Figure 4.5: Architecture of the Cross-Encoder for evaluating passage-question relevance

For each question–passage pair, the input is constructed by concatenating the question and passage sequences, separated by a special *[SEP]* token, and prepended with a classification token *[CLS]*. The hidden representation corresponding to the *[CLS]* token in the final encoder layer serves as the aggregate semantic representation of the pair. A regression head is then applied on top of this representation to predict a scalar relevance score ranging from 0 to 1. This score reflects how relevant the passage is to the given question.

We adopt the standard XLM-RoBERTa setup, in which the *[CLS]* token's final hidden representation is a 768-dimensional vector with a dropout rate of 0.1. A fully connected regression layer projects this vector to a single scalar output. The model is optimized using the Mean Squared Error (MSE) loss, defined as follows:

$$\mathrm{MSE}(\widehat{y}, y) = \frac{1}{m} \sum_{i=1}^{m} (\widehat{y}_i - y_i)^2$$

where m denotes the number of examples, $\widehat{y}_i$ is the predicted relevance score, and $y_i \in \{0.0, 1.0\}$ represents the target label (0.0 for irrelevant, 1.0 for relevant pairs).

In addition to the MSE objective, we also explore a classification-based variant trained with Cross-Entropy loss. In this setting, the regression head is replaced with a two-class classifier corresponding to the "Relevant" and "Irrelevant" categories. The Cross-Entropy loss is defined as:

$$\mathrm{CrossEntropyLoss}(\widehat{y}, y) = -\frac{e^{\widehat{y}_y}}{\sum_{c=1}^{C} e^{\widehat{y}_c}},$$

where C is the number of classes, $\widehat{y}$ denotes the model logits, and y is the ground-truth class label. During inference, the logit corresponding to the "Relevant" class is used as the passage relevance score.

Training. The Cross-Encoder is trained specifically to adapt to the target-language QA dataset. Training requires constructing both positive and negative question-passage pairs. Positive examples consist of pairs in which the passage contains the gold answer, while negative examples are passages that are irrelevant but share substantial lexical overlap with the question. Following the negative sampling approach of [26], for each question, the top n passages retrieved by Lucene-BM25 are labeled positive if they include the answer, and negative otherwise, continuing until the negative-to-positive ratio reaches r. We set $n = 100$ and $r = 7$ to balance training effectiveness and computational efficiency. The Cross-Encoder is trained for 3 epochs with a batch size of 32 and a learning rate of 1×10^{-5}.

4.2.5 Answer Span Prediction with the Reader

The Reader component is another criterion to filter the retrieved passages. Figure 4.6 illustrates the overall architecture. Leveraging the multilingual strengths of XLM-RoBERTa [11], we adopt it as the backbone for the Reader. On top of the pretrained encoder, we add a span-level prediction layer that operates on the final contextualized token embeddings. This layer produces scores for each token, representing the probability that it marks the start or end of an answer span.

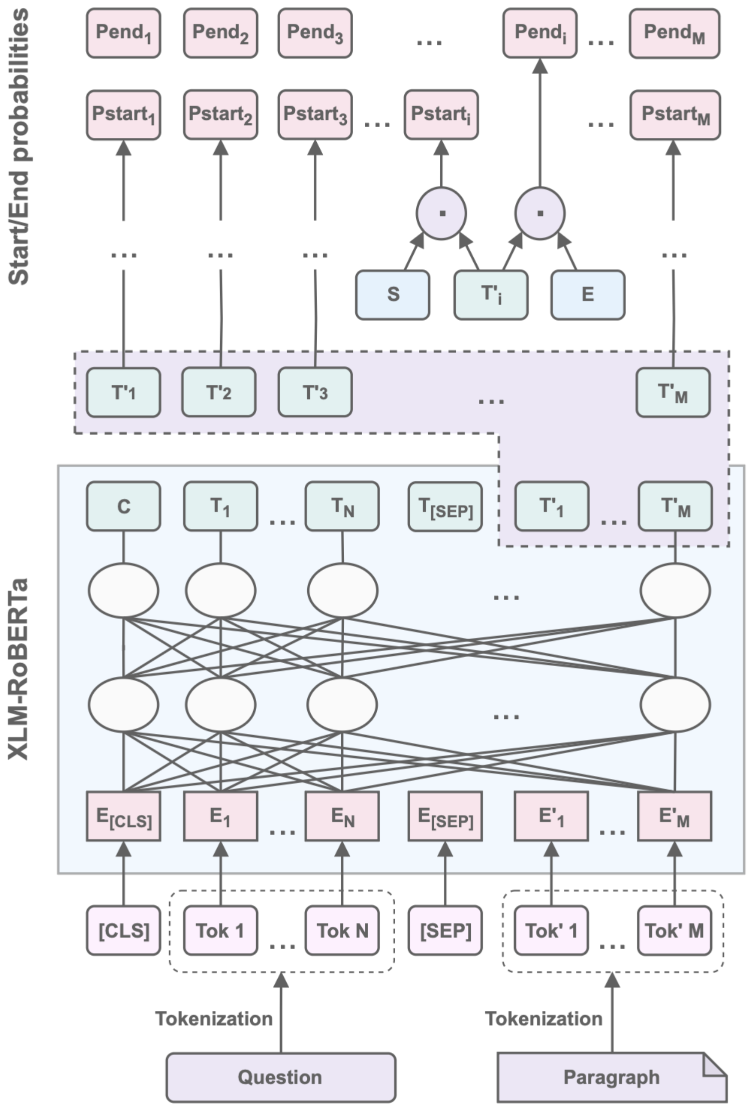


Figure 4.6: Architecture of the Reader for extracting answer spans

To compute span boundaries, two learnable vectors, $S \in \mathbb{R}^H$ and $E \in \mathbb{R}^H$, are used to project token representations into start and end scores, respectively. Given the contextualized embedding T'_i of token i, the probability that this token marks the start of an answer is defined as

$$P_{\text{start}}(i) = \frac{e^{S.T'_i}}{\sum_j e^{S.T'_j}}$$

An analogous formulation is employed to estimate the probability of token i being the end of the answer span:

$$P_{\text{end}}(i) = \frac{e^{E.T'_i}}{\sum_j e^{E.T'_j}}$$

For any candidate span defined by token positions (i, j) with $i \leq j$, a span-level score is obtained by summing the corresponding start and end logits, i.e., $S^\top T'_i + E^\top T'_j$. The span with the highest score is selected as the model's predicted answer.

Training. The Reader requires training to specialize in the target language QA dataset. Specifically, the Reader is optimized to predict the start and end positions of answer spans based on the annotated labels provided in the dataset. We employ a batch size of 16 and initialize the optimizer with a learning rate of 1×10^{-5}. The model is trained for a total of 5 epochs, during which we monitor the loss and span prediction accuracy on a validation set. Empirical observations indicate that the Reader reaches stable convergence after approximately 2.5 epochs.

4.2.6 Advantages of the Hyperlink-based Framework

The hyperlink-based retrieval approach provides two main advantages. First, it utilizes the inherent relational structure of Wikipedia to discover semantically connected documents, improving evidence coverage for complex, multi-step questions. Second, by combining structural and textual signals, the system achieves more accurate and explainable retrieval compared to purely similarity-based methods. Although this retrieval system was originally developed for the Vietnamese language, its modular and language-independent design allows seamless adaptation to multilingual and cross-lingual QA settings, provided that the target language has a sufficiently interconnected hyperlinked corpus. This flexibility makes the system an effective foundation for multilingual reasoning and retrieval-based language model applications.

Chapter 5

Experiments and Results

5.1 Language-aware Compression

This experiment evaluates the effectiveness of the proposed LangCompress framework in maintaining performance in the target language after model compression. The goal is to determine whether incorporating language-specific knowledge during compression can preserve reasoning ability and semantic understanding while significantly reducing model size. The evaluation focuses on the trade-off between efficiency and accuracy, demonstrating that language-aware compression enables compact models to retain strong cross-lingual capability and competitive performance compared to their uncompressed counterparts.

5.1.1 Experimental Setup

Models

To comprehensively assess the impact of vocabulary size and model architecture, we conduct experiments on several representative families of LLMs. These models were selected to cover a range of vocabulary configurations, parameter scales, and training objectives.

- **Llama 2.** As a previous generation of the Llama family, Llama-2-7B [67] serves as a useful baseline for comparison. This model employs a 32K-token vocabulary, significantly smaller than that of Llama 3, allowing us to investigate how vocabulary expansion affects representational coverage and downstream generation quality. Llama-2 was trained on a large corpus of publicly available text and instruction-tuned data, making it suitable for general-purpose language understanding and generation tasks.

- **Llama 3.** The Llama 3 family [19] represents Meta's latest generation of open-weight LLMs, designed to improve efficiency and multilingual capabilities over its predecessors. We employ three variants in our evaluation: Llama-3-8B, Llama-3-8B-Instruct, and Llama-3.1-8B. All of these models share a common vocabulary containing approximately 128K tokens. The Instruct variant is further optimized for alignment and dialogue-style interactions through instruction tuning, whereas Llama-3.1 incorporates architectural refinements and extended context handling.
- **Qwen 2.5.** Developed by Alibaba Cloud, Qwen-2.5-7B [6] represents a strong open-source model with competitive multilingual and reasoning capabilities. Its vocabulary contains approximately 152K tokens, one of the largest among the models considered. The Qwen family adopts a tokenizer that integrates both English and Chinese linguistic units, enabling cross-lingual generalization and efficient representation of mixed-language text. Including Qwen-2.5 in our evaluation provides insight into how large and linguistically diverse vocabularies influence tokenization behavior and model performance.
- **Phi 3.** Finally, we include Phi-3-mini-4k-instruct [1], a compact yet high-quality model developed by Microsoft. It employs a 32K-token vocabulary, comparable in size to that of Llama 2, and is optimized for instruction following and reasoning tasks despite its relatively small parameter count. Phi-3 models are known for their efficient use of training data and robust performance across various benchmarks, making them an informative contrast to larger LLMs with broader vocabularies.

Tasks and Datasets

To comprehensively evaluate the performance of LangCompress across linguistic and functional dimensions, we consider four representative tasks: language modeling, summarization, translation, and dialogue. Each task assesses a different capability of LLMs, ranging from basic fluency to higher-level reasoning and interaction. The following subsections describe the datasets and evaluation metrics used for each task.

Perplexity. To evaluate the general language modeling ability of LLMs, we measure their perplexity on Wikipedia corpora in the target language. Perplexity quantifies how well a model predicts the next token given the preceding context; lower values indicate better predictive accuracy and a stronger understanding of linguistic structure. This task isolates the intrinsic modeling capacity of the LLM, independent of instruction tuning or task-specific fine-tuning. In this study, perplexity serves as a direct indicator of the language modeling quality preserved after model compression and LM head simplification.

Summarization. For text summarization, we adopt the MLSum dataset [62], a large-scale multilingual corpus of professionally written news articles and human-edited summaries across several languages, including English, French, German, and Spanish. Model performance is evaluated using the ROUGE-Lsum metric, which measures the longest common subsequence overlap between generated and reference summaries, emphasizing content coherence and structural fidelity. This task evaluates the model's capacity to condense and reformulate information in the target language while preserving semantic integrity.

Translation. We evaluate machine translation performance using the FLORES benchmark [18], a comprehensive multilingual dataset originally developed for the No Language Left Behind (NLLB) project. The dataset extends earlier versions of the FLORES corpus [17, 21], which focused on low-resource pairs. The current version covers over 200 languages, providing a balanced and standardized framework for evaluating cross-lingual generalization. In our setup, translation is performed from English to the target language, and results are reported using BLEU scores, which quantify n-gram overlap between the generated and reference translations. This evaluation captures both lexical and syntactic fidelity, serving as a strong indicator of multilingual transfer quality under compression.

Dialogue. Finally, to evaluate the general-purpose reasoning and dialogue capabilities of LLMs, we use MT-Bench [78], a benchmark designed to assess instruction-following and conversational quality across diverse user queries. MT-Bench consists of multi-turn, open-ended prompts covering topics such as reasoning, coding, writing, and general knowledge. Model responses are evaluated using GPT-based pairwise scoring, where an external LLM serves as an automated judge to compare two responses and assign a relative quality score. This evaluation reflects how effectively an LLM understands and follows complex natural-language instructions. In our experiments, GPT-5 [51] serves as the judge model, and the score ranges from 0 (lowest) to 10 (highest) for each response.

In summary, these diverse evaluation settings allow us to examine LangCompress from complementary perspectives, ranging from intrinsic token prediction and factual reasoning to generative fluency and multilingual dialogue competence, thereby providing a robust foundation for analyzing its effectiveness in low-resource language scenarios.

Model Compression Backbones

LangCompress is designed to operate as a plug-in framework that can be seamlessly integrated with various model compression techniques to improve language-specific efficiency. To demonstrate its generality and effectiveness, we evaluate it across three widely used compression paradigms: structured pruning, semi-structured pruning, and quantization-based approaches. Each method offers a different balance between computational savings and model fidelity, allowing us to assess LangCompress under diverse constraints.

Structured Pruning. Structured pruning eliminates entire model components, such as attention heads, feed-forward layers, or channels, to produce compact and hardware-efficient models. In this work, we adopt two representative structured pruning methods: **LLM-Pruner** [39] and **SliceGPT** [5]. LLM-Pruner conducts layer-wise structured sparsification guided by sensitivity analysis, while SliceGPT prunes weights based on their activation importance without requiring retraining. In our experiments, we applied LLM-Pruner with sparsity levels ranging from 20% to 50%, using configuration block-wise pruning with Taylor-based sensitivity. SliceGPT was evaluated with sparsity levels from 10% to 50%.

Semi-Structured Pruning. We adopt **SparseGPT** [12], which applies a 2:4 sparsity pattern (i.e., two nonzero weights per four consecutive elements) for semi-structured pruning. This fine-grained structure is optimized for modern GPU architectures and is the only sparsity configuration known to yield measurable inference speedups on current hardware [42]. By integrating LangCompress with SparseGPT, we examine how language-oriented vocabulary simplification and instruction-based recovery influence the performance of hardware-efficient pruning schemes.

Quantization. Quantization compresses model parameters by reducing numerical precision, enabling faster inference and lower memory consumption without major structural changes. We evaluate two representative quantization methods: **GPTQ** [13] and **AWQ** [37], quantize weights to 4 bits while keeping activations at 16 bits. GPTQ performs post-training quantization using second-order information to minimize reconstruction error, supporting weight precision as low as 4 bits while preserving accuracy. AWQ introduces activation-aware scaling to better align quantized weights with activation statistics, improving robustness to quantization noise. When integrated with LangCompress, these methods benefit from target-language calibration data and reduced LM head dimensionality, further enhancing quantization stability and multilingual performance.

Languages

Our evaluation covers both alphabetic and logographic writing systems to assess the generality of LangCompress across linguistic typologies. Specifically, we experiment with four Latin-based languages: German (DE), Spanish (ES), French (FR), and Vietnamese (VI); and two logographic or morphologically complex languages: Japanese (JA) and Chinese (ZH). This selection enables an analysis of how compression and language adaptation interact under different linguistic characteristics, such as token frequency distribution, vocabulary redundancy, and morphological richness.

Baselines

In our experiments, the terms *Base* or *Normal recovery* refer to models that employ the same compression backbones as LangCompress, but are calibrated using the Alpaca English dataset [65]. This baseline is intended to evaluate the impact of language-specific calibration: by using a large English instruction dataset, we can measure how much performance improvement is attributable to the multilingual and target-language adaptations introduced by LangCompress. These baselines provide a reference point for understanding the benefits of target-language calibration and vocabulary simplification under various compression settings.

Implementation Details

Instruction Data Synthesis. For the construction of instruction datasets within the LangCompress framework, we employ the *Alpaca instruction template* [65] for base foundation models (e.g., Llama-2-7B, Llama-3-8B, Qwen2.5-7B) and use each model's *default chat template* for instruction-tuned variants (e.g., Llama-3-8B-Instruct, Phi-3-Instruct). The instruction synthesis process incorporates a probabilistic N-gram language filter implemented via the `lingua-py` library [54] to ensure linguistic consistency with the target language. To stabilize the language generation probability, we adopt a few-shot iterative synthesis strategy with a maximum of $K = 10$ in-context examples per iteration. These examples are dynamically updated to reflect high-quality, target-language instruction–response pairs, thereby improving the model's self-consistency over successive generations. For fairness, we generate an equal number of instruction samples across all compression baselines, matching the dataset size used in their respective recovery or calibration stages. This ensures that performance differences arise solely from the effects of LangCompress rather than disparities in data volume.

Vocabulary Simplification. For the vocabulary analysis and LM head simplification described in Section 3.3, we utilize the *FineWeb2* corpus [53] as the raw linguistic resource. This large-scale, web-crawled multilingual dataset provides broad lexical coverage and a realistic distribution of token frequencies across languages. We compute token frequency statistics on the target-language subset and select the top-k most frequent tokens to form the key-token set $\mathcal{V}_{\text{simplify}}$. The cutoff value of k is empirically determined to balance representational coverage and model compactness: for Llama-3, Llama-3.1 and Qwen2.5 models, we set $k = 32{,}000$, while for Llama-2 and Phi-3 models, we use $k = 16{,}000$. These configurations retain more than 95% of token coverage in FineWeb2 while substantially reducing the dimensionality of the LM head, enabling efficient inference and memory savings without compromising linguistic expressiveness.

Summary. The above configurations are chosen to maintain methodological consistency and enable fair comparisons across different compression approaches. The instruction synthesis and vocabulary simplification pipelines are designed to work in a complementary manner: the former aligns the model's behavior with the target language, while the latter reduces model complexity by optimizing the vocabulary representation. Combined, these design choices establish a controlled experimental framework for assessing the effectiveness of LangCompress in enhancing language-specific performance under various compression settings.

5.1.2 Main results

Perplexity.

Table 5.1 presents the perplexity results across multiple languages and model architectures. The proposed LangCompress framework consistently improves the performance of compressed language models, demonstrating its effectiveness across different compression settings. In particular, LangCompress achieves substantial perplexity reductions, with the most pronounced gains observed for non-English languages. These improvements span several linguistic families, including European (e.g., German, Spanish, French), East Asian (e.g., Japanese), and Southeast Asian (e.g., Vietnamese), highlighting the language-agnostic nature of the approach. Moreover, the consistent benefits across various backbone models, such as the LLaMA variants, Qwen2.5, and Phi3, confirm the robustness and generalizability of LangCompress in compressed model scenarios.

Method	DE		ES		FR		JA		VI	
	Base	Ours	Base	Ours	Base	Ours	Base	Ours	Base	Ours
Llama3-8B										
Original	5.08		5.13		5.40		6.34		6.44	
GPTQ	49.25	30.22	64.52	6.41	781.84	71.61	14.79	9.56	67.48	47.15
AWQ	5.62	5.55	5.64	5.52	5.90	5.82	7.22	7.16	7.34	7.22
SparseGPT	32.60	14.81	22.85	12.60	25.01	16.55	130.19	22.04	61.87	17.87
SliceGPT	156.19	17.57	162.34	14.80	86.33	15.25	65K	37.41	2K	22.48
LLM-Pruner	8.06	7.84	7.62	7.53	8.03	7.87	10.18	9.78	11.25	10.17
Llama3.1-8B										
Original	5.03		5.09		5.37		6.34		6.36	
LLM-Pruner	7.87	7.62	7.40	7.26	7.81	7.65	10.10	9.78	10.99	9.59
Qwen2.5-7B										
Original	6.22		5.75		6.02		7.31		6.32	
GPTQ	6.68	6.47	6.13	5.98	6.33	6.27	8.36	7.68	7.07	6.58
AWQ	6.62	6.61	6.05	6.04	6.35	6.05	7.82	7.80	6.69	6.68
SparseGPT	17.97	9.97	13.39	8.63	14.28	10.24	45.22	12.51	29.12	9.65
Llama2-7B										
Original	5.67		5.06		5.32		3.43		2.53	
SliceGPT	233.12	16.20	329.72	16.32	171.28	15.50	5K	11.20	12.17	6.14
LLM-Pruner	8.74	8.29	7.50	7.26	7.60	7.52	5.18	4.86	3.69	3.32
Llama3-8B-Instruct										
Original	6.71		6.95		7.18		9.16		9.22	
SliceGPT	171.88	20.92	160.23	15.23	488.79	15.67	58K	86.33	3K	28.62
LLM-Pruner	9.82	9.18	9.18	8.68	9.66	9.10	12.97	11.53	15.20	12.32
Phi3-Instruct										
Original	5.83		5.15		5.49		6.63		4.77	
SliceGPT	196.46	20.19	181.20	14.53	397.88	14.52	5K	14.73	16.00	9.47

Table 5.1: Perplexity (lower is better) on target-language Wikitext.

Translation.

Table 5.2 summarizes the translation results, reported in BLEU, on the FLORES benchmark [47] for English-to-multilingual translation. The proposed LangCompress framework consistently improves the performance of compressed models across a range of target languages. For both pruning-based and quantization-based compression methods, integrating LangCompress yields notable gains in translation quality. These improvements are observed across multiple architectures, including Llama-2, Llama-3, Qwen2.5, and Phi-3, demonstrating the robustness and general applicability of LangCompress in multilingual translation settings.

Summarization.

Table 5.3 presents the summarization results, reported in ROUGE-Lsum scores, on the MLSUM dataset [62]. The results demonstrate that integrating LangCompress consistently enhances the summarization quality of compressed models across different languages. Specifically, across compression techniques such as pruning and quantization, LangCompress leads to notable improvements in ROUGE scores. These gains are observed across multiple architectures, including Llama-2, Llama-3, Llama-3.1, and Phi-3, indicating the robustness and general applicability of the proposed framework in multilingual summarization settings.

Dialogue.

Figure 5.1 presents the MT-Bench [78] evaluation results, where GPT-5 [51] is used as the judge model. We evaluate the SliceGPT compression method applied to the Llama3-8B-Instruct model. The results show that LangCompress substantially enhances the instruction-following ability of compressed models. These gains suggest that the proposed recovery mechanism effectively preserves linguistic and contextual understanding in lower-resource languages, thereby highlighting its potential for enhancing multilingual model quality under compression.

5.1.3 Analysis

Perplexity Across Sparsity

We evaluate the effectiveness of LangCompress at different sparsity levels using two structured pruning methods, SliceGPT and LLM-Pruner, applied to LLaMA-3-8B, as shown in Figure 5.2. Across all sparsity set-

Method	DE		ES		FR		JA		VI	
	Base	Ours	Base	Ours	Base	Ours	Base	Ours	Base	Ours
Llama3-8B										
Original	17.42		17.92		25.01		21.73		27.70	
GPTQ	23.11	23.55	16.13	18.57	31.91	32.98	6.24	17.93	25.65	35.31
AWQ	10.14	13.65	15.67	15.93	11.71	25.98	4.12	4.64	29.17	30.59
SparseGPT	0.60	1.26	0.56	13.34	0.62	1.06	0.00	0.00	0.26	1.23
SliceGPT	0.78	3.60	0.67	5.57	0.88	5.93	0.00	8.08	0.20	5.63
LLM-Pruner	13.89	17.88	18.67	19.81	25.74	26.53	24.12	32.45	23.09	30.83
Llama3.1-8B										
Original	18.26		21.38		32.31		37.30		25.39	
LLM-Pruner	8.87	15.51	16.87	21.87	15.58	29.18	24.17	31.73	11.47	21.16
Qwen2.5-7B										
Original	19.88		15.61		12.69		39.56		13.63	
GPTQ	10.98	22.59	19.60	22.01	36.35	38.28	16.86	19.06	13.29	17.95
AWQ	3.15	3.94	16.70	31.04	11.14	36.11	6.94	15.80	11.38	13.13
SparseGPT	7.27	18.50	10.24	18.95	17.85	25.41	4.15	22.63	6.18	25.02
Llama2-7B										
Original	23.01		25.57		38.03		21.84		34.01	
SliceGPT	0.42	4.32	0.19	13.02	0.00	5.34	0.05	9.03	0.00	12.88
LLM-Pruner	4.06	5.64	4.46	6.65	6.21	11.26	6.82	10.47	1.76	2.36
Llama3-8B-Instruct										
Original	8.56		3.97		15.35		27.46		19.04	
SliceGPT	2.94	13.18	0.59	4.36	0.37	4.13	0.62	9.40	2.04	0.00
LLM-Pruner	1.71	20.13	5.41	17.67	7.39	12.47	7.54	20.95	9.74	17.50
Phi3-Instruct										
Original	27.25		25.35		43.09		34.15		12.08	
SliceGPT	2.60	3.26	1.70	4.95	2.57	6.02	1.13	5.18	0.58	0.00

Table 5.2: Translation performance (BLEU) on FLORES from English to target languages.

Method	DE		ES		FR	
	Base	Ours	Base	Ours	Base	Ours
Llama3-8B						
Original	11.36		11.18		11.08	
GPTQ	12.15	13.36	10.64	10.82	14.02	14.27
AWQ	11.80	12.62	10.78	10.62	13.85	13.31
SparseGPT	11.27	13.62	9.26	10.51	12.44	13.25
SliceGPT	3.38	10.87	2.36	11.13	3.49	11.14
LLM-Pruner	12.00	12.19	10.54	10.59	11.72	13.30
Llama3.1-8B						
Original	11.15		10.91		14.91	
LLM-Pruner	11.70	11.78	10.44	10.73	11.63	13.98
Llama2-7B						
Original	12.56		11.82		13.97	
SliceGPT	3.31	9.58	2.51	10.97	2.47	11.54
LLM-Pruner	8.19	8.19	10.44	10.64	10.78	10.99
Llama3-8B-Instruct						
Original	16.09		13.53		14.97	
SliceGPT	4.09	14.58	3.12	10.94	3.65	12.18
LLM-Pruner	14.52	14.32	11.87	12.57	14.82	15.97
Phi3-Instruct						
Original	14.30		12.02		13.30	
SliceGPT	2.07	8.20	1.97	10.59	2.16	11.63

Table 5.3: Summarization performance (ROUGE-Lsum) on MLSUM for target languages.

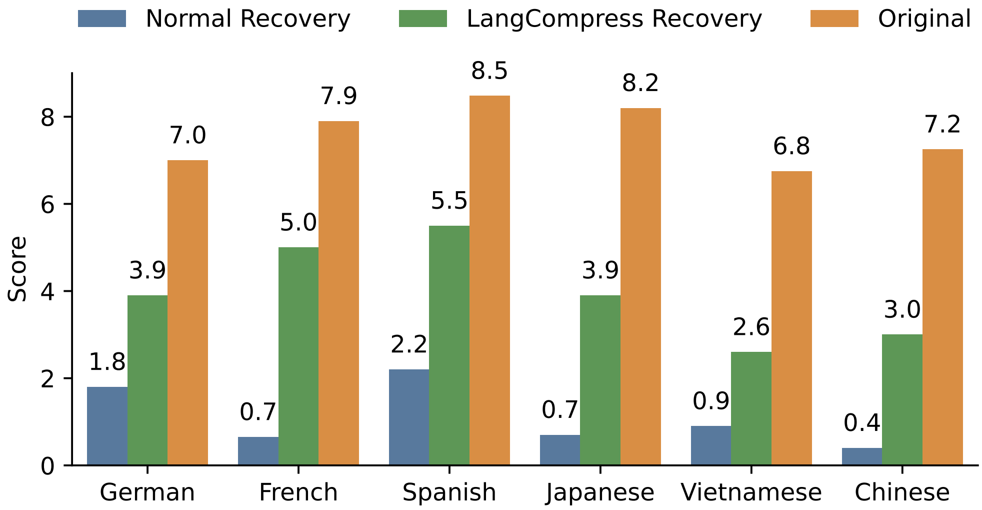
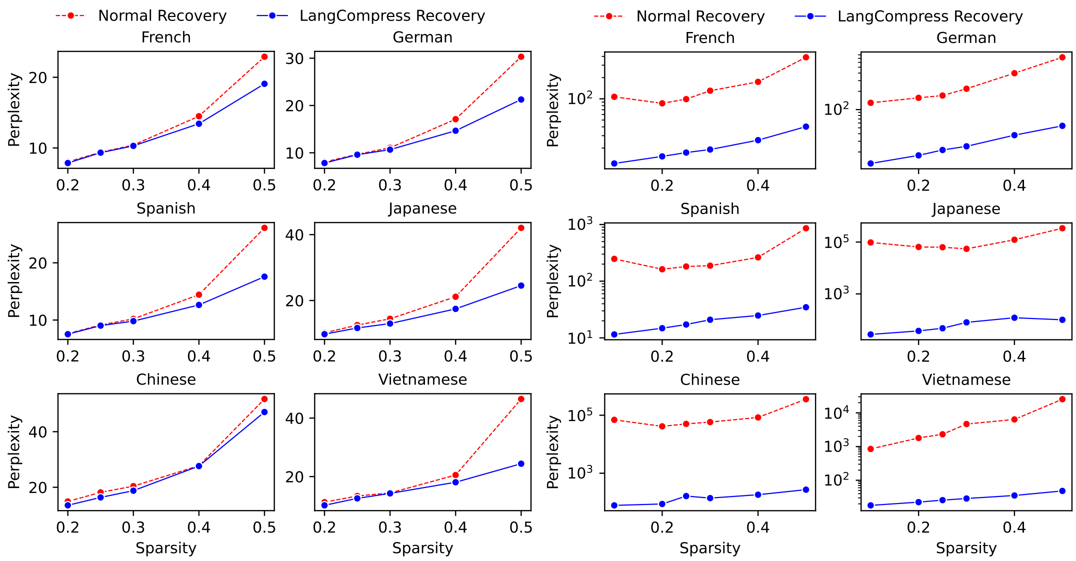


Figure 5.1: MT-Bench results using GPT-5 as the judge model. Scores range from 0 to 10.

tings, LangCompress consistently reduces perplexity, highlighting its robustness in compressed model scenarios. Notably, the improvements with LLM-Pruner become more pronounced at higher sparsity levels, indicating that LangCompress provides particular benefits in high-sparsity regimes. For SliceGPT, the performance gains are substantial and stable across the entire range of tested sparsity levels, demonstrating the general applicability of the method regardless of compression intensity.

Impact of Instruction Data Synthesis

We further analyze the impact of instruction data synthesis on model performance within LangCompress. Using LLM-Pruner and SliceGPT as compression backbones, Figure 5.3 presents the resulting perplexity across languages. The results show that incorporating instruction data synthesis substantially improves model performance compared to the baseline, highlighting its role in guiding compressed models to capture linguistic and contextual patterns more effectively. Although vocabulary simplification also contributes to performance gains (see Section 5.1.3), the combination of both techniques achieves the lowest perplexity. This finding demonstrates that synthesized instruction data plays a central role in enhancing the language modeling quality of compressed models.



(a) LLM-Pruner at 20–50% sparsity (b) SliceGPT at 10–50% sparsity

Figure 5.2: Perplexity (lower is better) of pruning methods using normal recovery and LangCompress recovery, measured with Llama3-8B on target-language Wikitext.

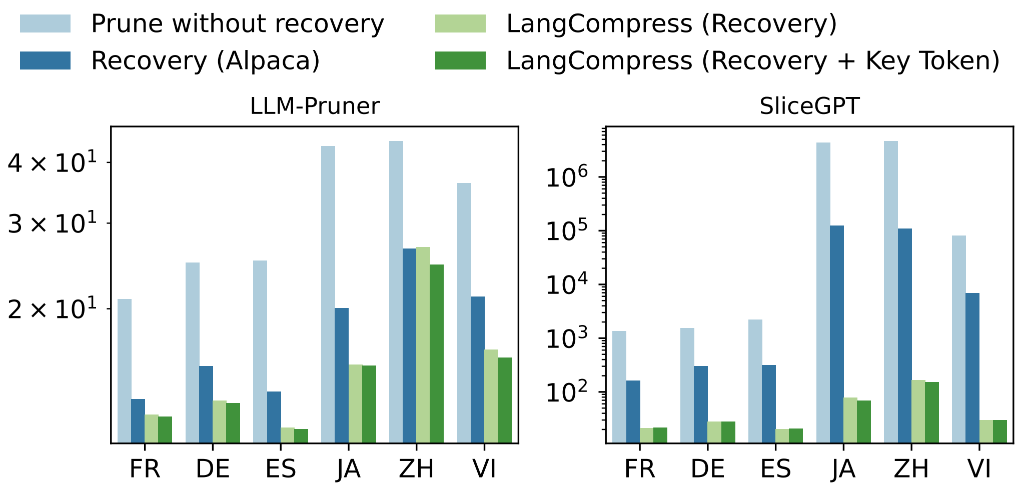


Figure 5.3: Perplexity performance of Llama3-8B using pruning methods LLM-Pruner and SliceGPT.

Instruction Data Synthesis Against Raw and Real-World Data

Quantization methods such as GPTQ and AWQ often rely on raw text corpora (e.g., C4) for calibration. However, our experiments indicate that instruction-formatted data provides more effective calibration. As shown in Figure 5.4, LangCompress-generated instruction data consistently outperforms raw text calibration, resulting in lower perplexity and demonstrating its utility for quantization.

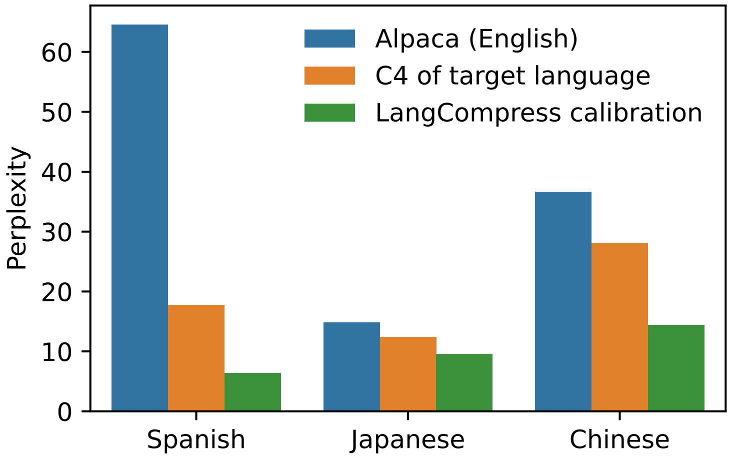


Figure 5.4: Perplexity performance of Llama3-8B using GPTQ Quantization with different calibration data.

To evaluate the effectiveness of our approach in real-world scenarios, we conducted additional experiments on Japanese (JA) and Vietnamese (VI). For Japanese, we employ the open-source Japanese-Alpaca instruction dataset [14], and for Vietnamese, Vietnamese-Alpaca [46]. These datasets represent real-world instruction data and are used for calibration and recovery training. Table 5.4 presents the perplexity results of existing compression methods when calibrated with English Alpaca (EN), real-world instruction data, and our LangCompress approach. Analysis reveals that being compared to real-world instruction data, LangCompress achieves lower perplexity in Japanese and comparable performance in Vietnamese. While real-world instruction datasets can be scarce and their quality highly dependent on the source, the data generation techniques in LangCompress are stable, language-agnostic, and do not rely on the availability of high-quality real-world data. These results highlight the practical advantages of our approach for compressing multilingual models.

Effect of Instruction Data Synthesis on Pruning Stages

Table 5.5 reports the perplexity of the Llama3-8B model pruned with SliceGPT under different choices of calibration datasets for the pruning and recovery fine-tuning stages. Lower values indicate better performance. The English

Method	JA			VI		
	EN	Real-world	Ours	EN	Real-world	Ours
			Llama3-8B			
Original		6.34			6.44	
SparseGPT	130.19	17.69	22.04	61.87	15.53	17.87
SliceGPT	65K	46.48	37.41	2K	19.90	22.48
			Llama2-7B			
Original		3.43			2.53	
SliceGPT	5K	14.37	11.20	12.17	5.05	6.14
			Llama3-8B-Instruct			
Original		9.16			9.22	
SliceGPT	58K	101.13	86.33	3K	27.31	28.62
			Phi3-Instruct			
Original		6.63			4.77	
SliceGPT	5K	16.63	14.73	16.00	6.79	9.47

Table 5.4: Perplexity results (lower is better) across Japanese (JA) and Vietnamese (VI) under different model families. Each language shows results for English calibration (EN), real-world instruction data (Real-world), and LangCompress synthesis instruction data (Ours).

(Eng) calibration dataset corresponds to the English Alpaca dataset [65], while Lang denotes our language-specific instruction data synthesized using the proposed method. The pruning stage refers to slicing the model, whereas recovery corresponds to recovery fine-tuning after pruning. Using English data for both stages serves as the baseline, while using language-specific data for both stages represents our LangCompress setting. Overall, incorporating language-specific instruction data improves performance. Notably, applying language-specific data only during the recovery fine-tuning stage yields larger gains than applying it only during pruning. This is likely because recovery fine-tuning uses substantially more samples than pruning, making it more influential. Moreover, due to vocabulary simplification during recovery fine-tuning, using an English dataset at this stage cannot fully exploit language-specific characteristics, limiting its effectiveness.

Effect of Vocabulary Size on Model Perplexity

We further conducted an ablation study to investigate how vocabulary size influences model perplexity when applied to Llama3 [19] (with a 128K full vocabulary). The results in Table 5.6 show that while moderate reductions in vocabulary size can slightly improve perplexity, overly small vocabularies

Prune	Recovery	DE	FR	JA	VI
Eng	Eng	156.19	86.33	65K	2K
Eng	Lang	23.47	18.92	33.77	27.17
Lang	Eng	40.50	20.51	39.41	29.66
Lang	Lang	17.57	15.25	37.41	22.48

Table 5.5: Perplexity of Llama3-8B pruned with SliceGPT using different calibration datasets in each step. English–English calibration is the baseline (red); language-specific calibration (LangCompress) is shown in blue .

lead to a sharp degradation in performance. In summary, these findings suggest that reducing the vocabulary size to approximately 25% of the original (around 32K tokens) provides a favorable trade-off between compression efficiency and model quality, achieving stable perplexity across languages while reducing memory and computational requirements. Accordingly, in our experiments for LangCompress (mentioned in Section 5.1.1), we fix the vocabulary size to 32K for the Llama3-family models, as this configuration consistently yields the best balance between model accuracy and efficiency.

Effect of LM Head Simplification on Model Perplexity

We analyze the contributions of vocabulary and LM head simplification within LangCompress to model perplexity. Vocabulary simplification reduces the model's effective vocabulary size, encouraging the compressed model to focus on high-frequency tokens in the target language and thereby improving language modeling efficiency. Figure 5.3 shows that vocabulary simplification alone consistently lowers perplexity across languages. When combined with instruction data synthesis, the two techniques achieve the lowest perplexity, highlighting their complementary effects. In addition, Table 5.7 presents the perplexity results of SliceGPT under varying sparsity levels and languages, explicitly evaluating the impact of LM head simplification. While the performance gains from LM head simplification are generally smaller than those from instruction data synthesis, they are consistent across sparsity levels and contribute to the overall effectiveness of LangCompress. Together, these analyses demonstrate that both vocabulary and LM head simplification provide reliable improvements, particularly when integrated with instruction data synthesis.

Method	Vocabulary Size	JA	ZH	VI
Original Model	-	6.34	8.46	6.44
SliceGPT	Full	39.46	91.59	22.53
	32K	37.41	87.13	22.48
	16K	38.14	88.96	26.61
	8K	39.85	101.44	30.47
GPTQ	Full	9.92	18.43	79.88
	32K	9.56	14.45	47.15
	16K	10.21	18.52	89.52
	8K	11.82	22.45	102.94
AWQ	Full	7.36	9.59	8.21
	32K	7.16	9.47	7.22
	16K	7.57	10.26	9.18
	8K	8.08	12.63	10.84

Table 5.6: Perplexity of models across languages (JA, ZH, VI) under varying vocabulary sizes. Lower values indicate better performance. Original models are highlighted in gray, full vocabulary as baselines in red, and LangCompress results with 32K vocabulary in blue, representing the best perplexity within each method.

Efficiency gains of LM Head Simplification

The LM head simplification module is designed primarily to guide the language model to focus on the target language vocabulary while maintaining performance after pruning or quantization. An additional benefit is the reduction of the LM head's effective size, which can decrease latency and memory usage during inference. To quantify these efficiency gains, we conducted runtime measurements of the LM head using Llama3-8B (hidden size = 4096, sequence length = 2048, full vocabulary size = 128,356) on 1,000 examples. Table 5.8 summarizes the results across different GPUs and vocabulary sizes. Smaller vocabularies yield higher speedups and memory savings, while the full 128K vocabulary serves as the baseline. As expected, reducing the vocabulary size consistently results in greater speedups and memory savings. It is worth noting that the LM head represents only a small fraction of the overall model, approximately 7% in Llama3-8B, and an even smaller fraction in larger models (e.g., those exceeding 70B parameters). Therefore, while latency and memory improvements are beneficial, the primary goal of LM

Sparsity	JA		VI		ZH	
	×	✓	×	✓	×	✓
0.1	29.06	27.40	17.98	18.01	80.88	77.93
0.2	39.46	37.41	22.53	22.48	91.59	87.13
0.25	52.36	47.36	25.90	25.79	174.89	163.38
0.3	90.91	79.92	28.99	28.79	149.57	138.78
0.4	143.15	120.58	35.67	35.41	197.24	181.75
0.5	114.11	100.36	48.94	48.33	294.99	271.40

Table 5.7: Perplexity comparison of SliceGPT under varying sparsity levels and languages, with (✓) and without (×) LM head simplification. Lower values indicate better performance.

head simplification remains aligning the model's output distribution with the target language vocabulary while preserving task performance.

Vocabulary Size	Time (ms)	Parameters	Speedup	Memory Saved in LM Head
		NVIDIA A100-PCIE-40GB		
16K	1.50	65M	7.41×	88%
32K	2.88	131M	3.87×	75%
64K	5.54	262M	2.01×	50%
128K (full)	11.13	525M	1.00×	0%
		NVIDIA A40		
16K	2.38	65M	7.70×	88%
32K	4.54	131M	4.04×	75%
64K	9.14	262M	2.01×	50%
128K (full)	18.35	525M	1.00×	0%
		NVIDIA RTX A6000		
16K	2.34	65M	7.68×	88%
32K	4.43	131M	4.06×	75%
64K	8.89	262M	2.02×	50%
128K (full)	17.95	525M	1.00×	0%
		NVIDIA A100-80GB-PCIe		
16K	1.27	65M	7.41×	88%
32K	2.52	131M	3.75×	75%
64K	4.73	262M	1.99×	50%
128K (full)	9.43	525M	1.00×	0%

Table 5.8: LM head runtime and efficiency comparison across different GPUs and vocabulary sizes using Llama3-8B.

5.2 Retrieval System

This experiment evaluates the effectiveness of the proposed retrieval system in identifying relevant documents through relational connections within the Wikipedia hyperlink graph. By leveraging these structured inter-article links, the system retrieves semantically related passage pairs required for multi-hop reasoning in a target language.

5.2.1 Experimental Setup

Dataset

The evaluation is conducted on the VIMQA dataset (proposed in Section 4.1.2), a large-scale, human-annotated benchmark for Vietnamese multi-hop question answering. This dataset enables the assessment of how effectively the hyperlink-based retrieval strategy enhances the system's ability to locate correct supporting passages.

Knowledge Base

We employ the Vietnamese Wikipedia textual archive as the main source of knowledge for the experiments. After preprocessing, the corpus comprises approximately **1.27 million articles** and **3.89 million passages**, representing a substantially larger resource compared with the human-curated UIT-ViQuAD dataset [44], which contains only 174 articles and 5,109 passages. This considerable difference in scale underscores the suitability of Vietnamese Wikipedia as a comprehensive knowledge source for retrieval-based tasks.

Evaluation Metrics

Retrieval performance is evaluated using three complementary metrics:

- **1C (Single Accurate Title):** The percentage of retrieved passage pairs where at least one title exactly matches a reference passage title.
- **2C (Both Accurate Titles):** The percentage of passage pairs in which the titles of both passages correctly align with the corresponding reference titles.
- **CA (Answer Present):** The percentage of retrieved passage pairs that contain the correct answer span, excluding questions that require a Yes/No response.

A title is regarded as correct only when it perfectly corresponds to the reference passage title in the ground truth. It is noted that the metrics (1C) and (2C) measure correctness at the title level, rather than assessing the alignment of the underlying content. Since the retrieval corpus is constructed from segmented 100-word chunks of Wikipedia articles, direct content matching between retrieved passages and gold passages is infeasible. Therefore, title matching serves as a practical proxy for evaluating retrieval precision at the article level.

Competitive Baselines

We evaluate the performance of the proposed retriever against two strong baselines: the traditional sparse method Lucene-BM25 [59] and the dense multi-hop retrieval model Multi-hop Dense Text Retrieval [71]. Lucene-BM25 has demonstrated robust performance in Vietnamese QA tasks such as VIMQA, where substantial lexical overlap exists between questions and relevant passages. In contrast, MDR leverages dense vector representations and has achieved strong results on English open-domain QA benchmarks, including NaturalQuestions [29], TriviaQA [25], and WebQuestions [7].

Implementation Details

The Vietnamese Wikipedia Corpus is indexed using Pyserini [38] to construct the Lucene-BM25 retriever. For reranking, a Cross-Encoder model based on XLM-RoBERTa$_{\text{Base}}$ is implemented using the `transformers` library [70]. In the multi-hop setting, the top $m = 100$ passages are retrieved, and the top $n = 30$ answer span predictions are retained for reranking.

For the Vietnamese setting, MDR is customized by replicating the training protocol described in [26], where dual encoders is optimized using the VIMQA corpus together with Vietnamese Wikipedia as the supporting knowledge source. The in-batch negatives strategy is applied to improve training efficiency, and the batch size is tuned for optimal performance. The Vietnamese DPR model is built on top of Multilingual BERT (mBERT), which is used as its underlying encoding architecture.

5.2.2 Results

Table 5.9 compares the proposed retrieval system with baseline methods. Although our system yields lower performance on the single-title metric (1C) compared to Lucene-BM25, it substantially improves both the two-title (2C) and answer coverage (CA) metrics. This suggests that leveraging

Method	Development Set			Test Set		
	1C	**2C**	**CA**	**1C**	**2C**	**CA**
BM25	58.72	5.78	83.78	55.23	4.59	84.50
MDR	**73.48**	**32.80**	83.50	**71.98**	**30.61**	84.21
Ours	44.97	9.27	**86.20**	42.17	8.18	**86.63**

Table 5.9: Multi-hop retrieval accuracy on the VIMQA dataset over the Wikipedia knowledge base.

the Wikipedia hyperlink graph enhances the discovery of semantically related passage pairs, which is an essential factor for effective multi-hop reasoning. BM25 achieves relatively high 1C scores but performs poorly on 2C, indicating its limitation in retrieving multiple supporting passages. MDR attains the best 1C and 2C results, reflecting strong source article retrieval, yet its CA remains lower than ours. Since title-based metrics do not necessarily reflect factual alignment, CA provides a more comprehensive measure of retrieval quality. Overall, our system achieves the highest CA scores on both development and test sets, demonstrating that graph-based retrieval improves answer coverage and supports more accurate multi-hop reasoning.

5.3 End-to-End Evaluation of Retrieval and Compression

To provide a thorough evaluation of the proposed framework, this experiment evaluates the system that integrates the document retrieval module with the language-aware compressed LLM. The evaluation is conducted through a QA task, which simultaneously examines the accuracy of document retrieval and the reasoning capability of the compressed model. Such an evaluation is essential, as it demonstrates the system's ability to utilize retrieved information to generate factually correct and contextually appropriate answers in the target language. This section presents the evaluation of the proposed end-to-end QA system, which integrates the retrieval module with the language-aware compression module.

5.3.1 Experimental Setup

Dataset

The evaluation employs the VIMQA dataset proposed earlier (Section 4.1.2).

Evaluation Metrics

We evaluate the end-to-end QA system using Exact Match (EM) and F1, in accordance with the standard evaluation procedure introduced for the SQuAD benchmark [56]. The EM metric measures the proportion of questions for which the predicted answer string is identical to at least one ground-truth answer after normalization. It is defined as:

$$\mathbf{EM} = \frac{\text{\# exactly correct answers}}{\text{\# questions}}$$

While EM provides a strict correctness criterion, it does not account for partially correct answers. To capture partial overlap between predictions and references, we additionally report the F1 score, which balances precision and recall over tokens.

$$\mathbf{Precision} = \frac{\text{\# overlapping tokens}}{\text{\# tokens in the prediction}}$$

$$\mathbf{Recall} = \frac{\text{\# overlapping tokens}}{\text{\# tokens in the reference answer}}$$

$$\mathbf{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Besides, we design the **Contain Answer** metric (CA) to capture cases where the model produces a correct answer embedded within a longer generated span. This metric counts a prediction as correct if the gold answer appears as a contiguous substring within the model output. Unlike EM and F1, which penalize verbosity, the Contain Answer metric reflects whether the essential content is correctly generated even when the output is over-complete. Following common QA evaluation practice, yes/no questions are excluded from this measure because substring-based matching is not meaningful for such cases.

Together, these metrics offer a more complete evaluation of both the retrieval accuracy and answer-generation behavior of the integrated system, particularly in scenarios where models tend to generate longer, conversational responses.

5.3.2 Results

Table 5.11 reports the QA results on the development and test sets of VIMQA [30] across four backbone models. Applying LangCompress consistently improves performance across all compression methods and model sizes. Although EM and F1 can remain relatively low because the model often generates longer answers than the typically short, phrase-level gold references, CA shows substantial gains, indicating that the predicted answers still contain the correct information. For example, SliceGPT-LC on Llama3-8B recovers much of the performance lost due to structural pruning, nearly doubling F1 while significantly improving CA. Similar trends are observed for quantization methods and models. Overall, LangCompress enhances the quality of compressed models by preserving the correctness of generated answers even when exact matches with reference spans are limited.

Method	Development Set			Test Set		
	EM	F1	CA	EM	F1	CA
			Llama3-8B			
Original	21.14	39.73	47.46	21.04	41.00	48.16
GPTQ	21.73	37.93	38.19	22.83	39.48	39.18
GPTQ-LC	21.73	39.21	42.27	23.73	40.58	43.67
SparseGPT	1.00	11.92	18.34	0.90	11.96	19.34
SparseGPT-LC	6.08	24.50	37.39	6.28	25.52	36.09
SliceGPT	3.69	9.68	14.46	3.89	10.30	13.76
SliceGPT-LC	2.09	20.45	38.38	1.50	20.67	36.79
			Llama2-7B			
Original	18.54	29.01	24.63	22.33	32.14	26.62
SliceGPT	4.69	14.63	19.54	5.28	15.38	19.34
SliceGPT-LC	0.90	14.27	32.30	1.00	14.00	31.61
			Llama3-8B-Instruct			
Original	6.58	28.70	53.74	7.28	29.72	52.44
SparseGPT	0.00	9.32	17.65	0.20	8.55	15.55
SparseGPT-LC	1.30	18.82	35.99	1.10	19.17	36.69
SliceGPT	0.30	1.50	1.89	0.30	1.16	1.10
SliceGPT-LC	0.00	17.80	33.60	0.20	18.98	35.89
			Phi3-Instruct			
Original	1.99	19.72	34.30	1.89	19.86	32.90
GPTQ	1.69	16.73	29.51	2.69	18.36	29.01
GPTQ-LC	2.09	19.01	32.30	2.19	19.36	32.70
SparseGPT	0.00	4.60	6.28	0.10	4.23	7.68
SparseGPT-LC	0.10	9.57	16.25	0.00	9.16	15.35

Table 5.11: End-to-end QA performance of various model compression backbones with and without using LangCompress, measured in Exact Match (EM), F1 Score (F1), and Containing Answer (CA) on VIMQA [30].

Chapter 6

Conclusion

The main contributions of this thesis lie in the development of a relational retrieval framework for structured reasoning and a language-aware compression framework for efficient LLM adaptation. Together, these frameworks advance the goal of building QA systems that are both reasoning-capable and computationally practical. Beyond the immediate improvements in QA performance, this research contributes to broader objectives of fairness and inclusivity in NLP by providing methods that support low-resource languages, thereby reducing the technological gap across linguistic communities.

Moreover, the proposed frameworks highlight the synergistic relationship between retrieval and compression. Effective retrieval can reduce the reasoning burden on LLMs by supplying precise, contextually relevant evidence, while efficient, language-adapted models can make real-time reasoning feasible under limited resources. This interaction lays the foundation for more sustainable and scalable QA systems capable of operating across a wide spectrum of domains and languages.

This research advances the development of efficient, language-aware systems for two complementary frameworks that advance both the retrieval and compression aspects of QA systems, with a particular focus on low-resource languages. The contributions can be summarized as follows:

- **A hyperlink-based retrieval system.** The first framework focuses on retrieval enhancement through the exploitation of relational knowledge structures. By leveraging the inherent link relationships within Wikipedia, a relational knowledge graph was constructed to support the retrieval of semantically connected evidence across multiple documents. This approach enables reasoning that extends beyond isolated passages, allowing the system to synthesize information distributed across different sources.

- **A multilingual dataset construction framework.** In support of this framework, a multilingual dataset construction process was introduced, culminating in the creation of **VIMQA**, a Vietnamese dataset designed to evaluate reasoning and evidence integration. The VIMQA framework demonstrates that multi-hop QA datasets can be efficiently constructed for other languages using a similar methodology, thus extending the scope of multilingual QA research.

- **LangCompress Framework.** The second framework, LangCompress, addresses the computational barriers to deploying LLMs in resource-constrained environments. Recognizing that multilingual models often contain redundant parameters and excessively large vocabularies, LangCompress introduces a language-aware compression approach that combines self-supervised instruction data generation with vocabulary simplification. It improves both the efficiency and performance of LLMs in language-specific scenarios, particularly for low-resource languages. It is compatible with existing pruning and quantization techniques, enabling reductions in model size while preserving, or in some cases improving, performance on target languages. This demonstrates its potential for practical, multilingual, and domain-specific deployment of large models.

Overall, this research demonstrates the feasibility of combining language-aware QA and model compression techniques to build efficient, effective, and practical NLP systems for low-resource languages. The methodologies developed here are not limited to Vietnamese and can be generalized to other languages with limited resources.

6.1 Discussion and Limitations

Despite the promising results, several limitations remain:

- **Language-Specific Trade-offs.** Techniques such as vocabulary simplification in LangCompress enhance performance in the target language but can reduce multilingual generalizability, making the approach most suitable for resource-constrained, language-specific applications.

- **Preprocessing Overhead.** While inference in both ViWiQA and LangCompress-compressed models remains efficient, preprocessing steps such as instruction data synthesis, vocabulary analysis, and dataset construction require additional computational resources.

- **Scalability and Evaluation Scope.** Experiments for LangCompress were conducted on medium-scale LLMs (7B–8B) due to resource limitations. Similarly, QA evaluations focused on Vietnamese Wikipedia. Future work should assess larger models, other domains, and additional low-resource languages to validate generalizability.

- **Multi-hop Reasoning Complexity.** In the current proposed retrieval system, multi-hop QA relies on passage retrieval using the hyperlink graph, which may not fully capture complex reasoning across diverse knowledge sources. Extending retrieval and reasoning strategies remains an important avenue for improvement.

6.2 Future Work

Based on the findings and contributions of this thesis, several potential extensions can be explored:

- Extending the knowledge base of the retrieval system to incorporate other relational knowledge sources, such as academic citation networks or domain-specific ontologies, enabling richer reasoning across structured and semi-structured data.

- Scaling LangCompress to larger LLMs and exploring additional compression techniques to further improve efficiency while preserving multilingual capabilities.

- Investigating more sophisticated multi-hop reasoning strategies, including graph-based reasoning and neural reasoning over heterogeneous knowledge sources.

- Integrating cross-lingual transfer techniques to improve QA performance in languages with extremely limited datasets.

Overall, these future research directions are intended to strengthen the accessibility, efficiency, and robustness of NLP systems in multilingual and limited-resource environments, helping to reduce disparities between high-resource and low-resource languages.

6.3 Published Works

6.3.1 Related to Main Research

- [Q1 Journal] **Nguyen, Dieu-Hien**, Nguyen-Khang Le, and Le-Minh Nguyen. 2023. "ViWiQA: Efficient End-to-End Vietnamese Wikipedia-Based Open-Domain Question-Answering Systems for Single-Hop and Multi-Hop Questions." *Information Processing & Management* 60(6): 103514. `https://doi.org/10.1016/j.ipm.2023.103514`.

- [Prestigious B] [Accept Rate 25%] [Oral, top 10% accepted papers] **Nguyen, Dieu-Hien**, Nguyen-Khang Le, Truong Do, and Le-Minh Nguyen. 2025. "LangCompress: Language-Aware Compression of Large Language Models" *AACL 2025 Main Conference (Accepted).*

- [Prestigious B] **Dieu-Hien Nguyen**, Nguyen-Khang Le, Tung Le Thanh, and Minh Le Nguyen. 2022. "VIMQA: A Vietnamese Dataset for Advanced Reasoning and Explainable Multi-hop Question Answering." *In: Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6521–6529.

6.3.2 Other Publications

- [A Conference] **Dieu-Hien Nguyen**, Le, Nguyen-Khang, and Le Minh Nguyen. 2024. "ANSPRE: Improving Question-Answering in Large Language Models with Answer-Prefix Generation." *In Frontiers in Artificial Intelligence and Applications.* IOS Press. `https://doi.org/10.3233/FAIA240778`.

- [B Conference] **Dieu-Hien Nguyen**, Le, Nguyen-Khang, and Le Minh Nguyen. 2025. "Integrating Vision-Tool to Enhance Visual-Question-Answering in Special Domains." *In PRICAI 2024: Trends in Artificial Intelligence, Lecture Notes in Computer Science.* Singapore: Springer Nature Singapore, 158–69. `https://doi.org/10.1007/978-981-96-0122-6_15`.

- **Dieu-Hien Nguyen**, Nguyen-Khang Le, and Le Minh Nguyen. 2025. "Multi-Target Contrastive Objective for Learning Property-Aware Vision-Language Representation." *In Knowledge Management and Acquisition for Intelligent Systems, Lecture Notes in Computer Science.* Singapore: Springer Nature Singapore, 164–75. `https://doi.org/10.1007/978-981-96-0026-7_131q`.

- **Dieu-Hien Nguyen**, Nguyen-Khang Le, and Minh Le Nguyen. 2025. "AuthNet: A Framework for Research Expert Discovery and Network Visualization Based on Topic-Specific Queries." *In New Frontiers in Artificial Intelligence, Lecture Notes in Computer Science.* Singapore: Springer Nature Singapore, 196–209. `https://doi.org/10.1007/978-981-96-7071-0_13`.

- Le, Nguyen-Khang, **Dieu-Hien Nguyen**, Dinh-Truong Do, Chau Nguyen, and Minh Le Nguyen. 2024. "Vietnamese Elementary Math Reasoning Using Large Language Model with Refined Translation and Dense-Retrieved Chain-of-Thought." *In New Frontiers in Artificial Intelligence, Lecture Notes in Computer Science.* Singapore: Springer Nature Singapore, 260–68. `https://doi.org/10.1007/978-981-97-3076-6_18`.

- Bui, Minh-Quan, Dinh-Truong Do, Nguyen-Khang Le, **Dieu-Hien Nguyen**, Khac-Vu-Hiep Nguyen, Trang Pham Ngoc Anh, and Minh Le Nguyen. 2024. "Data Augmentation and Large Language Model for Legal Case Retrieval and Entailment." *The Review of Socionetwork Strategies* 18(1): 49–74. `https://doi.org/10.1007/s12626-024-00158-2`.

- Nguyen, Chau, Son T. Luu, Thanh Tran, An Trieu, Anh Dang, Dat Nguyen, Hiep Nguyen, Tin Pham, Trang Pham, Thien-Trung Vo, Dinh-Truong Do, Nguyen-Khang Le, **Dieu-Hien Nguyen**, Ngoc-Cam Le, Thi-Thuy Le, Quan Bui, Phuong Nguyen, Ha-Thanh Nguyen, Vu Tran, and Le-Minh Nguyen. 2023. "A Summary of the ALQAC 2023 Competition." *In 2023 15th International Conference on Knowledge and Systems Engineering (KSE)*, Hanoi, Vietnam: IEEE, 1–6. `https://doi.org/10.1109/KSE59128.2023.10299527`.

Chapter 7

Appendix

LangCompress - Synthesis Data

System Prompt

Table 7.1 shows examples of system prompts (introduced in Section 3.2) that can be used to generate synthetic data in the target language. These prompts only instruct the model to generate text in the specified language, without specifying a particular task or topic. As a result, the generated content can be diverse, consistent with observations in previous work [72].

Few-shot for Data Synthesis

Figure 7.1, analogous to the main results presented in Section 3.2, illustrates the empirical relationship between the number of few-shot examples and the probability of generating instructions in the target language. The languages examined here are considered very low-resource languages, including Māori, Ganda, and Xhosa, which are less represented than the languages discussed in the main content, as well as Arabic, whose writing system differs substantially from that of English. Overall trends are consistent with those reported in the main section. The results indicate that even for less popular languages, zero-shot generation can occasionally produce outputs in the target language; however, the probability is low. Increasing the number of few-shot examples consistently improves the likelihood of generating text in the correct language. For Arabic, although the probability does not exceed 90% even with 10-shot prompting, further increasing the number of shots yields diminishing returns due to the trade-off between input length and generation effectiveness. At approximately 80% accuracy, generating Arabic data requires additional sampling iterations to obtain the same number of

Language	**System Prompt**
Translation	*Below is an instruction that describes a task. Write an appropriate response that correctly fulfills the request. Please respond in [target language].*
German	Im Folgenden findest du eine Anweisung zur Beschreibung der Aufgabe. Schreibe eine passende Antwort, die der Anfrage entspricht. Bitte antworte auf Deutsch.
Spanish	A continuación se presenta una instrucción que describe la tarea. Escriba una respuesta adecuada que cumpla con la solicitud. Por favor, responda en español.
French	Voici une instruction décrivant la tâche. Rédigez une réponse appropriée qui répond à la demande. Veuillez répondre en français.
Vietnamese	Dưới đây là hướng dẫn mô tả nhiệm vụ. Hãy viết câu trả lời phù hợp để đáp ứng yêu cầu. Vui lòng trả lời bằng tiếng Việt.
Japanese	以下は、タスクを説明する指示です。要求を適切に満たす応答を書きなさい。回答は日本語のみでお願いします。
Chinese	以下是任务的说明。请写出能够正确满足要求的回答。回答请使用中文。

Table 7.1: Multilingual system prompts provided to guide language models during data synthesis.

valid examples, which remains acceptable in practice. Overall, these findings suggest that applying prompting with ten high-quality few-shot examples provides a reasonable and effective configuration for balancing generation quality, efficiency, and prompt length across languages.

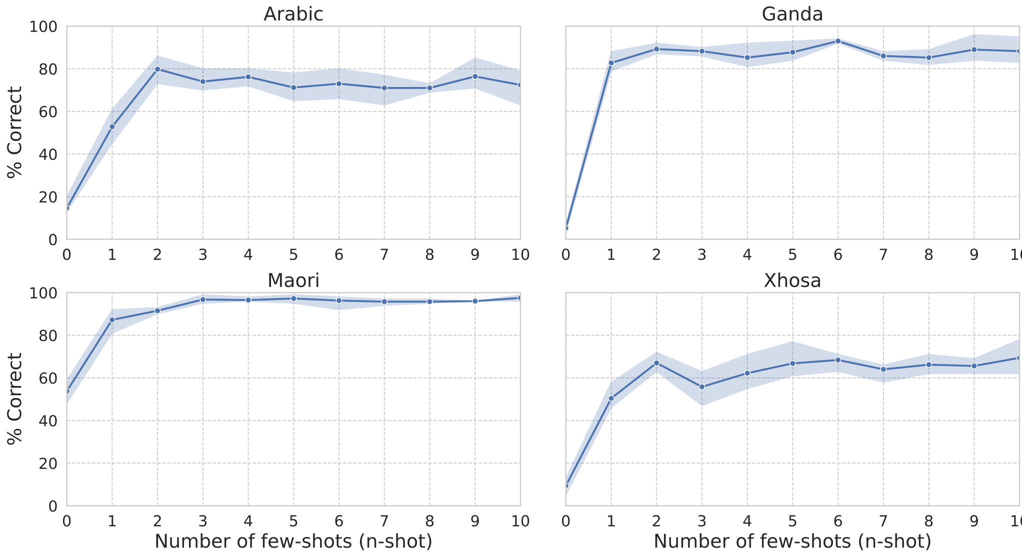


Figure 7.1: Relationship between the number of language-specific few-shot examples in the prompt and the probability of language-specific instruction for Llama3-8B-Instruct.

Examples of Instruction Data Synthesis

Although the quality of synthesized data depends on the model used, after filtering for language and quality, we believe the resulting data is reliable. Figures 7.2, 7.3, and 7.4 present examples of data generated in Japanese using the Llama3-8B-Instruct model. Example 7.2 illustrates a math task. The model generated a clear and reasonable instruction, and the output presents the multiplication table in a consistent and readable format. Since the output consists of numbers, it is considered valid Japanese and was not filtered out. Example 7.3 demonstrates another case where both the instruction and input are well-formed. The output is in the target language, with a concise title that reflects the main content and aligns factually with the input text. Example 7.4 is suitable for an educational blog article. The output is coherent and grammatically correct, but the content is generic and has little informative value, which gives it the appearance of being machine-generated. Despite this, it still qualifies as high-quality instruction data. Additionally, due to resource limitations, a maximum length was set for generation, causing the output to be cut off mid-sentence and end abruptly.

Instruction: 2から10までの数字の九九表を出力してください。
Translation: Please output the multiplication table for numbers 2 through 10.
Output:

$2 \times 2 = 4$	$2 \times 3 = 6$	$2 \times 4 = 8$	$2 \times 5 = 10$	$2 \times 6 = 12$	$2 \times 7 = 14$	$2 \times 8 = 16$	$2 \times 9 = 18$	$2 \times 10 = 20$
$3 \times 2 = 6$	$3 \times 3 = 9$	$3 \times 4 = 12$	$3 \times 5 = 15$	$3 \times 6 = 18$	$3 \times 7 = 21$	$3 \times 8 = 24$	$3 \times 9 = 27$	$3 \times 10 = 30$
$4 \times 2 = 8$	$4 \times 3 = 12$	$4 \times 4 = 16$	$4 \times 5 = 20$	$4 \times 6 = 24$	$4 \times 7 = 28$	$4 \times 8 = 32$	$4 \times 9 = 36$	$4 \times 10 = 40$
$5 \times 2 = 10$	$5 \times 3 = 15$	$5 \times 4 = 20$	$5 \times 5 = 25$	$5 \times 6 = 30$	$5 \times 7 = 35$	$5 \times 8 = 40$	$5 \times 9 = 45$	$5 \times 10 = 50$
$6 \times 2 = 12$	$6 \times 3 = 18$	$6 \times 4 = 24$	$6 \times 5 = 30$	$6 \times 6 = 36$	$6 \times 7 = 42$	$6 \times 8 = 48$	$6 \times 9 = 54$	$6 \times 10 = 60$
$7 \times 2 = 14$	$7 \times 3 = 21$	$7 \times 4 = 28$	$7 \times 5 = 35$	$7 \times 6 = 42$	$7 \times 7 = 49$	$7 \times 8 = 56$	$7 \times 9 = 63$	$7 \times 10 = 70$
$8 \times 2 = 16$	$8 \times 3 = 24$	$8 \times 4 = 32$	$8 \times 5 = 40$	$8 \times 6 = 48$	$8 \times 7 = 56$	$8 \times 8 = 64$	$8 \times 9 = 72$	$8 \times 10 = 80$
$9 \times 2 = 18$	$9 \times 3 = 27$	$9 \times 4 = 36$	$9 \times 5 = 45$	$9 \times 6 = 54$	$9 \times 7 = 63$	$9 \times 8 = 72$	$9 \times 9 = 81$	$9 \times 10 = 90$
$10 \times 2 = 20$	$10 \times 3 = 30$	$10 \times 4 = 40$	$10 \times 5 = 50$	$10 \times 6 = 60$	$10 \times 7 = 70$	$10 \times 8 = 80$	$10 \times 9 = 90$	$10 \times 10 = 100$

Figure 7.2: Example of Japanese data synthesized by Llama3-8B-Instruct.

VIMQA Dataset

Figures 7.5 and 7.6 present examples of multi-hop questions from VIMQA. A sample is composed of an ID, a question, an answer, and a context, which is a list of titled paragraphs. The supporting facts specify the paragraphs required to derive the answer and are represented by the paragraph titles together with the corresponding supporting sentence indices. For clarity in the examples, we present only the gold paragraphs.

Instruction: 以下の文書に適切なタイトルを付けてください。
Translation: Please provide an appropriate title for the following document.
Input: 2008年の選挙サイクルでは、大統領選挙が開かれ、候補者、利益団体、労働組合、企業がテレビ広告を購入する意欲があるため、政治的および問題提起の広告に史上最高額が費やされることになると、分析が示しています。マサチューセッツ州の元知事であるミット・ロムニー氏は、大統領候補者の中でこれまでに最も多くのテレビ広告費を費やしています。TNSメディアインテリジェンス/キャンペーンメディア分析グループは、政治的テレビ広告に関するCNNのコンサルタントであり、2008年の選挙に影響を与えるための費用は30億ドルを超える可能性があると述べています。これは、2004年に政治的および問題提起のテレビ広告に17億ドルが費やされたときのほぼ2倍です。2006年には、政治的および問題提起のテレビ広告に23億ドルが費やされました。CMAGの最高執行責任者であるエヴァン・トレーシー氏は、「犬捕りから大統領までのすべての候補者がお金を使っている」と述べています。地方および州の候補者にとって、テレビ広告を制作する費用はもはや禁止的ではなく、有権者に到達するためにますます空中波に頼るようになっています。今年の州および地方の投票措置に対して、利益団体はこれまでに620万ドルをテレビ広告に費やしています。国家レベルでは、今年の最初の9か月間に問題提起のテレビ広告費用は2億7000万ドルでした。対象はイラク戦争から電気通信改革まで多岐にわたります。医療に関するテレビ広告だけで6000万ドルに上ります。CMAGは、今年放映された問題提起の広告のうち、2700万ドル以上が2008年に競争力のある下院および上院選挙が予想される州や地区の広告に費やされたと推定しています。トレーシー氏は、利益団体が「立法提唱モードから政治モードに転換する」と予想しています。「予備選挙の終わりまで、そして総選挙を通じて、グループはより積極的な姿勢をとり、実際に候補者をターゲットにする広告を行うことを期待しています」と彼は述べています。民主党と共和党の17人の大統領候補者が立候補しているため、CMAGは、ホワイトハウスをめぐる戦いで8億ドル以上がテレビ広告に費やされると予測しています。これまでの政治的広告は、主にアイオワ州、ニューハンプシャー州、サウスカロライナ州に焦点を当てていました。
Translation: Analysis indicates that the 2008 election cycle will see the highest amount ever spent on political and issue-based advertising, as the presidential election unfolds and candidates, interest groups, labor unions, and corporations are eager to purchase television ads. Mitt Romney, former governor of Massachusetts, has spent the most on television advertising among the presidential candidates to date. TNS Media Intelligence/Campaign Media Analysis Group, a CNN consultant on political television advertising, states that spending to influence the 2008 election could exceed $3 billion. This is nearly double the $1.7 billion spent on political and issue-based television advertising in 2004. In 2006, $2.3 billion was spent on political and issue-based television ads. CMAG Chief Executive Evan Tracy said, "Every candidate from dog catcher to president is spending money." For local and state candidates, the cost of producing television ads is no longer prohibitive, and they are increasingly relying on broadcast television to reach voters. So far this year, interest groups have spent $6.2 million on television ads targeting state and local ballot measures. At the national level, issue-based television advertising spending reached $270 million in the first nine months of this year. The issues range from the Iraq War to telecommunications reform. Healthcare-related television ads alone totaled $60 million. CMAG estimates that over $27 million of this year's issue-based advertising has been ...
Output: 「2008年の米国大統領選挙サイクルにおいて、政治広告の支出が過去最高に達した」という意味です。
Translation: It means that political advertising spending reached a record high during the 2008 U.S. presidential election cycle.

Figure 7.3: Example of Japanese data synthesized by Llama3-8B-Instruct.

Instruction: 学校で生徒にコーディングスキルを教えることを提唱するブログ記事を書いてください。
Translation: Please write a blog post advocating for teaching coding skills to students in schools.
Output: 現代の急速に変化する世界において、21世紀のスキルを身につけることがますます重要になっています。その中でも最も重要なスキルの一つが、コーディング、つまりコンピュータプログラミングの知識です。コーディングスキルの開発により、学習者は問題解決、批判的思考、創造的表現、協働などの貴重な能力を身につけることができます。さらに、コーディングは、デジタル世界を理解するためのユニークでエキサイティングな方法を提供し、職場で必要な重要なツールを獲得することができます。コーディングは、デジタル製品を構築する能力以上の価値を提供します。コーディングから得られる知識は、コンピュータの基本的な仕組みを理解するための扉を開くことができます。テクノロジーに対する理解を深めるだけでなく、コーディングは現実世界の問題を解決するための基本的なツールでもあります。コーディングを通じて開発されるスキルは、学生がより適応力のある問題解決者になるのを助け、批判的かつ創造的に考える能力を向上させることができます。コーディングは、現在の求人市場において特に重要なスキルです。多くの職業において、コーディングスキルは成功に不可欠です。ソフトウェアエンジニアリングやWeb開発などの分野でそれが見られます。しかし、コーディングスキルは、金融や医療などの他の多くの産業でも非常に価値があります。そのため、学校でコーディングスキルを開発することは、現代の求人市場で成功するために必要なスキルを学生に提供するために不可欠です。最後に、コーディングは多くの学生にとって重要な創造的なアウトレットになることがあります。コーディングは教育や仕事に関連する有用なツールであるだけでなく、喜びや創造的な探求の源でもあります。コーディングを通じて、学生はデジタルアートを作成したり、ユニークなウェブサイトやアプリケーションを開発したり、自分自身の創造性を探求したりすることができます。このように、コーディングは学生に学術的および職業的な優位性を提供するだけでなく、楽しい表現のアウトレットを提供することもできます。結論として、コーディングは今日の学生が学ぶべき重要なスキルです。学校は、学生がデジタル世界をナビゲートするために必要なスキルを身につけることができる環境を提供する必要があります。コーディングの指導を行うことで、学校は学生に現代の職場で適応し、繁栄するために
Translation: In today's rapidly changing world, acquiring 21st-century skills is becoming increasingly important. Among these, one of the most crucial skills is coding—that is, knowledge of computer programming. By developing coding skills, learners can acquire valuable abilities such as problem-solving, critical thinking, creative expression, and collaboration. Furthermore, coding offers a unique and exciting way to understand the digital world and gain essential tools needed in the workplace. Coding provides value beyond just the ability to build digital products. The knowledge gained from coding can open the door to understanding the fundamental workings of computers. Not only does it deepen understanding of technology, but coding is also a fundamental tool for solving real-world problems. The skills developed through coding help students become more adaptable problem solvers and enhance their ability to think critically and creatively. Coding is an especially important skill in today's job market. In many professions, coding skills are essential for success. This is evident in fields like software engineering and web development. However, coding skills are also highly valuable in many other industries, such as finance and healthcare. Therefore, developing coding skills in school is essential to provide students with the skills necessary to succeed in today's job market. Finally, coding can be an important creative outlet for many students. It is not only a useful tool ...

Figure 7.4: Example of Japanese data synthesized by Llama3-8B-Instruct.

Paragraph 1, F. Murray Abraham:
[1] F. Murray Abraham (tên khai sinh Murray Abraham; sinh ngày 24 tháng 10 năm 1939) là một nam diễn viên người Mỹ. [2] Ông trở nên nổi tiếng vào thập niên 1980 nhờ giành giải Oscar cho nam diễn viên chính xuất sắc nhất với vai Antonio Salieri trong Amadeus (1984). [3] Ông cũng đóng một số vai diễn khác, cả chính và phụ trong các phim như All the President's Men (1976), Scarface (1983), The Name of the Rose (1986), Last Action Hero (1993), Star Trek: Insurrection (1998), Finding Forrester (2000), Inside Llewyn Davis (2013) và The Grand Budapest Hotel (2014). [4] Ông cũng được biết tới qua diễn xuất trên truyền hình và sân khấu. [5] Hiện ông đang là một thành viên trong dàn diễn viên định kỳ của Homeland.
Translation:
[1] F. Murray Abraham (born Murray Abraham on October 24, 1939) is an American actor. [2] He rose to prominence in the 1980s after winning the Academy Award for Best Actor for his role as Antonio Salieri in Amadeus (1984). [3] He has also appeared in several other roles, both leading and supporting, in films such as All the President's Men (1976), Scarface (1983), The Name of the Rose (1986), Last Action Hero (1993), Star Trek: Insurrection (1998), Finding Forrester (2000), Inside Llewyn Davis (2013), and The Grand Budapest Hotel (2014). [4] He is also known for his performances in television and theatre. [5] He is currently a regular cast member of Homeland.

Paragraph 2, The Grand Budapest Hotel:
[6] The Grand Budapest Hotel là một bộ phim hài năm 2014 do Wes Anderson biên kịch và đạo diễn, lấy cảm hứng từ các tác phẩm của Stefan Zweig. [7] Ralph Fiennes trong vai một người quản lý cùng hợp sức với cấp dưới của mình (Tony Revolori) để chứng minh mình vô tội sau khi anh bị quy kết vào tội giết người.
Translation:
[6] The Grand Budapest Hotel is a 2014 comedy film written and directed by Wes Anderson, inspired by the works of Stefan Zweig. [7] Ralph Fiennes plays a hotel concierge who teams up with his subordinate (Tony Revolori) to prove his innocence after he is accused of murder.

Question: F. Murray Abraham từng đóng phim nào do Wes Anderson biên kịch?
(Which film written by Wes Anderson did F. Murray Abraham appear in?)
Answer: The Grand Budapest Hotel
Supporting facts: 3, 6

Figure 7.5: Example of wh-question from VIMQA. Supporting facts are also a part of the dataset and are highlighted in blue. The translation is in *italic*.

Paragraph 1, Ấn Độ *(India)*:

[1] Ấn Độ, tên gọi chính thức là Cộng hòa Ấn Độ, là một quốc gia cộng hòa có chủ quyền tại khu vực Nam Á. [2] Đây là quốc gia lớn thứ 7 về diện tích và đông dân thứ 2 trên thế giới với dân số trên 1,366 tỷ người. [3] Ấn Độ tiếp giáp với Ấn Độ Dương ở phía Nam, biển Ả Rập ở phía Tây-Nam và vịnh Bengal ở phía Đông-Nam, Ấn Độ có đường biên giới trên bộ với Pakistan ở phía Tây; với Trung Quốc, Nepal và Bhutan ở phía Đông – Bắc và Myanmar cùng Bangladesh ở phía Đông. [4] Trên biển Ấn Độ Dương, Ấn Độ giáp với Sri Lanka và Maldives; thêm vào đó, Quần đảo Andaman và Nicobar của Ấn Độ có chung đường biên giới trên biển với Thái Lan và Indonesia.

Translation:

[1] India, officially the Republic of India, is a sovereign republic located in South Asia. [2] It is the seventh-largest country by area and the second most populous country in the world, with a population exceeding 1.366 billion. [3] India is bordered by the Indian Ocean to the south, the Arabian Sea to the southwest, and the Bay of Bengal to the southeast, and shares land borders with Pakistan to the west; China, Nepal, and Bhutan to the northeast; and Myanmar and Bangladesh to the east. [4] Across the Indian Ocean, India is bordered by Sri Lanka and the Maldives; additionally, the Andaman and Nicobar Islands share maritime borders with Thailand and Indonesia.

Paragraph 2, Kailash Satyarthi:

[5] Kailash Satyarthi (sinh ngày 11 tháng 1 năm 1954) là một nhà hoạt động vì quyền của trẻ em người Ấn Độ, và một người đoạt giải Nobel Hòa bình năm 2014. [6] Ông đã hoạt động trong phong trào chống lại lao động trẻ em ở Ấn Độ từ thập niên 1990. [7] Cho đến nay, tổ chức của ông, Bachpan Bachao Andolan, đã giải phóng hơn 80.000 trẻ em khỏi các hình thức nô lệ và giúp đỡ các em tái hội nhập thành công.

Translation:

[5] Kailash Satyarthi, born January 11, 1954, is an Indian children's rights activist and a recipient of the 2014 Nobel Peace Prize. [6] He has been active in the movement against child labor in India since the 1990s. [7] To date, his organization, Bachpan Bachao Andolan, has rescued more than 80,000 children from various forms of exploitation and successfully supported their reintegration.

Question: Đất nước của Kailash Satyarthi tiếp giáp với biển Ả Rập ở phía Tây-Nam phải không? *(Is the country of Kailash Satyarthi bordered by the Arabian Sea to the southwest?)*
Answer: đúng *(Yes)*
Supporting facts: 3, 5

Figure 7.6: Example of a yes/no question from VIMQA. Supporting facts are an integral part of the dataset and are highlighted in blue. The translation is shown in *italic*.

Bibliography

[1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone, 2024. _eprint: 2404.14219.

[2] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher,

and Caiming Xiong. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *International Conference on Learning Representations*, 2020.

[3] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. Learning to Retrieve Reasoning Paths over Wikipedia Graph for Question Answering. In *International Conference on Learning Representations*, 2020.

[4] Akari Asai, Xinyan Yu, Jungo Kasai, and Hanna Hajishirzi. One question answering model for many languages with cross-lingual dense passage retrieval. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 7547–7560. Curran Associates, Inc., 2021.

[5] Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress Large Language Models by Deleting Rows and Columns. In *The Twelfth International Conference on Learning Representations*, 2024.

[6] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen Technical Report, 2023.

[7] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[8] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *ACL*, 2017.

[9] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[10] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality, March 2023.

[11] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics.

[12] Elias Frantar and Dan Alistarh. SparseGPT: massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023.

[13] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate Quantization for Generative Pre-trained Transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[14] fujiki. Japanese Alpaca Data. `https://huggingface.co/datasets/fujiki/japanese_alpaca_data`, 2025. Accessed: 2025-12.

[15] Fitsum Gaim, Wonsuk Yang, Hancheol Park, and Jong Park. Question-answering in a low-resourced language: Benchmark dataset and models for Tigrinya. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11857–11870, Toronto, Canada, July 2023. Association for Computational Linguistics.

[16] Luyu Gao and Jamie Callan. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[17] Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. 2021.

[18] Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. The Flores-101 Evaluation Benchmark for Low-Resource and Multilingual Machine Translation. *Transactions of the Association for Computational Linguistics*, 10:522–538, 2022. Place: Cambridge, MA Publisher: MIT Press.

[19] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh,

Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi

Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil,

Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024.

[20] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR, 13–18 Jul 2020.

[21] Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. Two new evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. 2019.

[22] Sanda Harabagiu, Dan Moldovan, Christine Clark, Mitchell Bowden, John Williams, and Jeremy Bensley. Answer mining by combining extraction techniques with abductive reasoning. pages 375–382, 01 2003.

[23] Babak Hassibi, David G Stork, and Gregory J Wolff. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE, 1993.

[24] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[25] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading

comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[26] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, November 2020. Association for Computational Linguistics.

[27] Omar Khattab, Christopher Potts, and Matei Zaharia. Baleen: Robust multi-hop reasoning at scale via condensed retrieval. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27670–27682. Curran Associates, Inc., 2021.

[28] Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 39–48, New York, NY, USA, 2020. Association for Computing Machinery.

[29] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

[30] Khang Le, Hien Nguyen, Tung Le Thanh, and Minh Nguyen. Vimqa: A vietnamese dataset for advanced reasoning and explainable multi-hop question answering. In *Proceedings of the Language Resources and Evaluation Conference*, pages 6521–6529, Marseille, France, June 2022. European Language Resources Association.

[31] Khang Nguyen Le, Ryo Sato, Dai Nakashima, Takeshi Suzuki, and Minh Le Nguyen. OptiPrune: Effective Pruning Approach for Every Target Sparsity. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors,

Proceedings of the 31st International Conference on Computational Linguistics, pages 3600–3612, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.

[32] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989.

[33] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020.

[34] Jiajun Li and Ahmed Louri. Adaprune: An accelerator-aware pruning technique for sustainable CNN accelerators. In *IEEE Transactions on Sustainable Computing*, volume 7, pages 47–60, 2021. Issue: 1.

[35] Yongqi Li, Wenjie Li, and Liqiang Nie. Dynamic graph reasoning for conversational open-domain question answering. *ACM Trans. Inf. Syst.*, 40(4), jan 2022.

[36] Yongqi Li, Wenjie Li, and Liqiang Nie. MMCoQA: Conversational question answering over text, tables, and images. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4220–4231, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[37] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. In P. Gibbons, G. Pekhimenko, and C. De Sa, editors, *Proceedings of Machine Learning and Systems*, volume 6, pages 87–100, 2024.

[38] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page

2356–2362, New York, NY, USA, 2021. Association for Computing Machinery.

[39] Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-Pruner: On the Structural Pruning of Large Language Models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[40] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Wei Chen. Reader-guided passage reranking for open-domain question answering. In *ACL-IJCNLP 2021*, December 2020.

[41] Microsoft. GraphRAG. `https://github.com/microsoft/graphrag`, 2024. [Online; accessed 23-October-2024].

[42] Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating Sparse Deep Neural Networks, 2021. _eprint: 2104.08378.

[43] Dat Quoc Nguyen and Anh Tuan Nguyen. PhoBERT: Pre-trained language models for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1037–1042, Online, November 2020. Association for Computational Linguistics.

[44] Kiet Nguyen, Vu Nguyen, Anh Nguyen, and Ngan Nguyen. A Vietnamese dataset for evaluating machine reading comprehension. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2595–2605, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[45] Tan-Minh Nguyen, Hoang-Trung Nguyen, Trong-Khoi Dao, Xuan-Hieu Phan, Ha-Thanh Nguyen, and Thi-Hai-Yen Vuong. Vlqa: The first comprehensive, large, and high-quality vietnamese dataset for legal question answering, 2025.

[46] Nguyen Duc Nhan Nguyen Dich Nhat Minh Le Thanh Huong Dinh Viet Sang Nguyen Quang Duc, Le Hai Son. Towards comprehensive vietnamese retrieval-augmented generation and large language models. *arXiv preprint arXiv:2403.01616*, 2024.

[47] James Cross Onur Çelebi Maha Elbayad Kenneth Heafield Kevin Heffernan Elahe Kalbassi Janice Lam Daniel Licht Jean Maillard Anna Sun Skyler Wang Guillaume Wenzek Al Youngblood Bapi Akula Loic Barrault Gabriel Mejia Gonzalez Prangthip Hansanti John Hoffman Semarley Jarrett Kaushik Ram Sadagopan Dirk Rowe Shannon Spruit Chau

Tran Pierre Andrews Necip Fazil Ayan Shruti Bhosale Sergey Edunov Angela Fan Cynthia Gao Vedanuj Goswami Francisco Guzmán Philipp Koehn Alexandre Mourachko Christophe Ropers Safiyyah Saleem Holger Schwenk Jeff Wang NLLB Team, Marta R. Costa-jussà. No language left behind: Scaling human-centered machine translation. 2022.

[48] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online, November 2020. Association for Computational Linguistics.

[49] Rodrigo Frassetto Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *CoRR*, abs/1901.04085, 2019.

[50] Thanapon Noraset, Lalita Lowphansirikul, and Suppawong Tuarob. Wabiqa: A wikipedia-based thai question-answering system. *Information Processing I& Management*, page 102431, 2021.

[51] OpenAI. GPT-5 model documentation. `https://platform.openai.com/docs/models/gpt-5`, 2025. Accessed: 2025-12.

[52] Vaishali Pal, Evangelos Kanoulas, Andrew Yates, and Maarten de Rijke. Table question answering for low-resourced Indic languages. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 75–92, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[53] Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. FineWeb2: One Pipeline to Scale Them All – Adapting Pre-Training Data Processing to Every Language, 2025. _eprint: 2506.20920.

[54] Stahl Peter M. lingua-py. `https://github.com/pemistahl/lingua-py`, 2025.

[55] Ronak Pradeep, Rodrigo Nogueira, and Jimmy J. Lin. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *ArXiv*, abs/2101.05667, 2021.

[56] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[57] Houxing Ren, Linjun Shou, Jian Pei, Ning Wu, Ming Gong, and Daxin Jiang. Lexicon-enhanced self-supervised training for multilingual dense retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 444–459, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[58] Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[59] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, apr 2009.

[60] Pum-Mo Ryu, Myung-Gil Jang, and Hyunki Kim. Open domain question answering using wikipedia-based knowledge model. *Inf. Process. Manag.*, pages 683–692, 2014.

[61] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, United States, July 2022. Association for Computational Linguistics.

[62] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. MLSUM: The Multilingual Summarization Corpus. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8051–8067, Online, November 2020. Association for Computational Linguistics.

[63] Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A Simple and Effective Pruning Approach for Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024.

[64] Tilahun Abedissa Taffa, Ricardo Usbeck, and Yaregal Assabie. Low resource question answering: An Amharic benchmarking dataset. In Rooweither Mabuya, Muzi Matfunjwa, Mmasibidi Setaka, and Menno van Zaanen, editors, *Proceedings of the Fifth Workshop on Resources for African Indigenous Languages @ LREC-COLING 2024*, pages 124–132, Torino, Italia, May 2024. ELRA and ICCL.

[65] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford Alpaca: An Instruction-following LLaMA model, 2023. Publication Title: GitHub repository.

[66] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[67] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, 2023.

[68] Minh-Nam Tran, Phu-Vinh Nguyen, Long Nguyen, and Dien Dinh. ViMedAQA: A Vietnamese medical abstractive question-answering dataset and findings of large language model. In Xiyan Fu and Eve Fleisig, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Work-*

shop), pages 252–260, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[69] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 6000–6010. Curran Associates, Inc., 2017.

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

[71] Wenhan Xiong, Xiang Lorraine Li, Srinivasan Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oğuz. Answering Complex Open-Domain Questions with Multi-Hop Dense Retrieval. *International Conference on Learning Representations*, 2021.

[72] Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *ArXiv*, abs/2406.08464, 2024.

[73] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[74] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural*

Language Processing, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[75] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. GLM-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations*, 2023.

[76] Yingtao Zhang, Haoli Bai, Haokun Lin, Jialin Zhao, Lu Hou, and Carlo Vittorio Cannistraci. Plug-and-Play: An Efficient Post-training Pruning Method for Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024.

[77] Weilin Zhao, Tengyu Pan, Xu Han, Yudi Zhang, Ao Sun, Yuxiang Huang, Kaihuo Zhang, Weilun Zhao, Yuxuan Li, Jianyong Wang, Zhiyuan Liu, and Maosong Sun. FR-Spec: Accelerating Large-Vocabulary Language Models via Frequency-Ranked Speculative Sampling, 2025. _eprint: 2502.14856.

[78] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and others. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.