Project title

Your name

Supervisor: Supervisor name

A report submitted in fulfilment of the requirements for the degree of BSc in Computer Science

in the

Department of Computer Science

September 17, 2024

Declaration

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Large language models have recently demonstrated outstanding performance in the field of natural language processing tasks, including machine translation, question answering systems, text generation, and text summarization. These models, with their vast number of parameters and complex self-attention mechanisms, have greatly improved predictive capabilities and driven advancements in related technologies. However, one of the distinctive features of LLMs is their massive parameter size. In comparison to smaller models or traditional neural networks, LLMs contain significantly more parameters, allowing them to capture richer linguistic patterns and improve performance. However, this also brings with it an immense demand for computational resources.

To enhance performance, the scale of LLMs has consistently expanded in recent years. The parameter count has evolved from models with millions of parameters to today's models with hundreds of billions of parameters. For instance, one of the leading models, BLOOM, has a parameter count of 176 billion (Fan et al., 2023). This rapid increase in parameter size has led to notable improvements in performance, but it has also significantly increased the demand for computational resources, creating bottlenecks in computing capacity. Running such large models introduces substantial costs in terms of both computation time and monetary expenses. Despite advancements in hardware, such as GPUs, model size is outpacing the improvements in hardware performance, limiting the widespread deployment of LLMs in environments with restricted computational resources.

Model compression techniques have emerged as a key solution to mitigate these issues by reducing the computational demand while preserving as much of the model's performance as possible.

1.2 Purpose

To address the growing computational demands of large language models, model compression techniques have become a crucial area of research. Model compression aims to reduce the computational load of these models while maintaining high performance, making them more accessible for deployment in resource-limited environments. One of the most effective and widely used compression techniques is pruning, which selectively removes less important parameters from the model, thereby reducing its size and computational cost.

Pruning has gained particular attention due to its balance between simplicity and effectiveness. Unlike methods such as quantization or knowledge distillation, pruning directly reduces the model's parameters, making it an efficient way to compress large models without the need for complex post-processing or retraining stages. Moreover, pruning has shown great potential in preserving model accuracy even at high sparsity levels. By carefully selecting which parameters to remove, pruning can significantly reduce the size of the model while maintaining its ability to perform complex tasks such as natural language understanding and reasoning. This makes pruning a more targeted and effective solution for large-scale models compared to other compression methods.

Recent research has suggested that certain modules, like attention mechanisms, are more critical to model performance than others. If pruning is applied uniformly across all components, it may lead to the over-pruning of crucial modules, such as attention heads, resulting in significant performance degradation. A more refined pruning strategy, which accounts for the varying importance of different modules, could achieve a better balance between model sparsity and performance. This idea of differential pruning, where attention is given to which modules are pruned more or less, has emerged as a promising direction for improving the efficiency of large language models.

Thus, understanding the role of each module within the model, particularly how pruning affects each component, is essential for developing more advanced and efficient compression strategies. This insight motivates the need for more granular, module-specific pruning methods that not only reduce computational costs but also preserve the critical capabilities of large language models.

1.3 Motivation

Traditional pruning techniques often apply the same pruning ratio across all layers and modules of the model, which can lead to suboptimal results. This uniform approach fails to consider the relative importance of different components within the model. For example, attention mechanisms, which enable large language models to handle long-range dependencies in text, may be more sensitive to pruning than other parts of the model, such as MLP layers. Over-pruning these critical components can lead to significant performance loss, especially in tasks that rely heavily on complex reasoning and contextual understanding.

In contrast, module-level pruning allows for more targeted pruning strategies. By applying different pruning ratios to distinct components, such as attention and MLP modules, it becomes possible to retain more critical parameters while pruning less important ones. This granularity enables more control over the model's sparsity, allowing for higher compression rates without sacrificing performance. For instance, recent studies have demonstrated that attention modules, due to their role in capturing dependencies across long sequences, are often more crucial to model performance than MLP layers. A more nuanced pruning strategy, therefore, could preserve attention heads to a greater extent while applying more aggressive pruning to other components.

The motivation behind this research is to explore the impact of such module-level pruning strategies on large language models. By systematically evaluating the effects of pruning different components, we aim to identify the most effective pruning methods for preserving task performance while achieving significant reductions in model size. Our hypothesis is that by focusing on module-level pruning, it is possible to achieve a better balance between model sparsity and task accuracy compared to traditional uniform pruning methods.

This study focuses on two main modules of large language models: the attention mechanism and the MLP layers. Through a series of experiments, we evaluate the effects of pruning these components on the model's performance across a range of natural language processing tasks. The results provide new insights into how pruning can be optimized at a more granular level, demonstrating that module-level pruning is a promising approach for achieving efficient model compression without compromising performance. This research not only advances the understanding of pruning techniques but also offers practical strategies for deploying large models in environments with limited computational resources.

1.4 Report Structure

This report is organized as follows: Chapter 2 reviews the relevant literature on LLMs pruning and model compression techniques. Chapter 3 explains the experimental setup, including the models, datasets, and evaluation metrics used in this study. Chapter 4 presents the results of the experiments, analyzing the effects of pruning on model performance across different modules. Finally, Chapter 5 concludes the report by summarizing the findings and discussing future research directions.

Chapter 2

Literature Review

2.1 Large Language Models

Large Language Models (LLMs) have gained significant attention due to their remarkable performance across a wide array of Natural Language Processing (NLP) tasks. These models, particularly those built upon the Transformer architecture as introduced by Vaswani et al. (2017), such as GPT-3 (Brown et al., 2020) and BLOOM (Fan et al., 2023), have shown exceptional capabilities in text generation, question answering, machine translation, and beyond. By leveraging billions of parameters, LLMs are able to capture complex linguistic patterns and produce human-like text with high accuracy. However, despite their state-of-the-art performance, the massive size of these models poses significant computational challenges, particularly during training and inference stages.

The rapid growth of LLMs has been driven by the pursuit of enhanced performance. Early models like GPT-2 and BERT, which contained hundreds of millions of parameters, have been eclipsed by more recent models such as GPT-4 and BLOOM, which boast hundreds of billions of parameters. This growth has been accompanied by a corresponding increase in the model's ability to handle diverse NLP tasks, from complex reasoning to multi-lingual translation. However, these advancements come at a cost. The computational demands of training and deploying such models have escalated, often requiring powerful hardware resources such as GPU and TPU clusters (Brants et al., 2007; Min et al., 2021). Furthermore, the cost in terms of energy consumption and time has become a bottleneck, limiting the accessibility of these models for widespread real-world use. While LLMs have demonstrated superior capabilities, their enormous resource consumption highlights the need for more efficient solutions.

In addition to the computational demands, the inherent complexity of LLMs also presents challenges in optimizing and fine-tuning these models for specific tasks. Recent studies, such as those focusing on "Chain of Thought" prompting, have demonstrated that by encouraging LLMs to generate intermediate reasoning steps, it is possible to significantly improve their performance on complex reasoning tasks (Wei et al., 2022). While LLMs undoubtedly hold significant potential across a wide range of domains, reducing their computational overhead without sacrificing performance has become a critical research direction.

2.2 Model Compression

Model compression has become a critical area of research in the quest to make Large Language Models (LLMs) more computationally efficient, enabling their deployment on resource-constrained devices and environments. As LLMs continue to scale, with models such as GPT-3 and BLOOM consisting of hundreds of billions of parameters, the computational resources required for both training and inference have increased exponentially. This has led to the exploration of several model compression techniques that aim to reduce the size, memory footprint, and computational demands of these models, while preserving their performance across various tasks.

One of the most prominent compression techniques is quantization, which involves reducing the precision of model weights and activations. Quantization works by mapping floating-point numbers to lower-precision representations, such as 8-bit integers, which can significantly reduce memory usage and computational overhead. Quantization has been particularly effective for deploying LLMs in edge environments, where memory and computational resources are limited. For instance, techniques like Activation-aware Weight Quantization focus on weight-only quantization while maintaining accuracy in downstream tasks, thus enabling on-device inference without substantial performance degradation (Lin et al., 2023). Recent research has shown that quantization techniques can achieve significant compression while preserving performance in zero-shot tasks, making them one of the most viable options for LLM deployment (Jha et al., 2023).

Another significant approach to compression is knowledge distillation, where a smaller "student" model is trained to replicate the behavior of a larger "teacher" model (Hinton et al., 2015). In this process, the student model learns to approximate the output probabilities of the teacher model, thereby transferring knowledge while reducing the model size. This method is effective in creating lightweight versions of large models that perform similarly on specific tasks. However, knowledge distillation often requires significant retraining, which can be computationally expensive and time-consuming. Despite this, knowledge distillation remains an important tool for producing more efficient models, especially when training smaller models from scratch is not feasible.

Pruning, a third major category of compression techniques, aims to remove redundant or less important parameters from the model. By identifying and removing these less critical elements, pruning reduces the overall size of the model while maintaining as much performance as possible. Pruning is particularly appealing because it directly addresses the issue of model sparsity, often without requiring extensive retraining, which can save time and computational resources compared to other methods. Traditional pruning techniques like magnitude-based pruning remove weights based on their absolute values, assuming that weights with smaller magnitudes contribute less to the model's output (LeCun et al., 1990). While effective, this unstructured approach can lead to hardware inefficiencies due to irregular sparse patterns in the pruned model.

To address these limitations, structured pruning methods have been developed. Strucured pruning removes entire neurons, channels, or attention heads, creating more regular sparsity patterns that are better suited for hardware acceleration (Wang et al., 2019). Structured pruning is especially advantageous for modern Transformer-based models, as it maintains the integrity of the model's architecture, allowing it to perform efficiently on parallel hardware such as GPUs and TPUs. By pruning whole components, structured pruning helps models retain their performance while making them more deployment-friendly in real-world scenarios.

In addition to these traditional methods, recent research has introduced binarization techniques, such as Partially Binarized LLMs , which represent a step toward extreme compression. By reducing model weights to binary values (e.g., -1 and +1), PB-LLMs achieve substantial reductions in model size and computational complexity while retaining much of the model's linguistic reasoning capability (Shang et al., 2023). This makes binarization particularly attractive for deploying LLMs in environments with very tight computational and memory constraints, such as mobile devices or IoT platforms.

Moreover, adaptive pruning approaches have emerged, aiming to dynamically adjust the pruning strategy based on model performance. Techniques like Reweighted Proximal Pruning fine-tune the importance of individual parameters during training, allowing for more efficient pruning that retains critical task-specific features (Guo et al., 2019). Similarly, contrastive pruning methods, such as ContrAstive Pruning, employ contrastive learning to ensure that the pruned model retains both generalizable and task-specific knowledge, making it particularly useful for pre-training and fine-tuning paradigms (Xu et al., 2022).

In summary, model compression techniques have evolved significantly in recent years, with methods such as quantization, pruning, and knowledge distillation offering promising solutions for the deployment of LLMs in real-world applications. Each of these methods presents unique trade-offs in terms of accuracy, computational efficiency, and deployment feasibility, and ongoing research continues to explore the best ways to balance these factors. While each method has its advantages, pruning has gained particular attention in the context of LLMs due to its simplicity, effectiveness, and adaptability to different model architectures. Furthermore, adaptive pruning strategies and newer methods like binarization offer exciting opportunities for even more efficient compression, positioning model compression as a critical area of research for the future of LLMs.

2.3 Pruning Methods

Pruning has become one of the most prominent methods for model compression, especially in large language models (LLMs), where computational and memory efficiency are crucial. Pruning techniques aim to remove redundant or less significant parameters in a model, resulting in a more compact model with minimal loss in performance. These methods can be broadly categorized into unstructured pruning, structured pruning, and more recent adaptive pruning techniques.

Unstructured pruning involves removing individual weights based on some measure of importance, typically the magnitude of the weight. One of the earliest and most widely used unstructured pruning techniques is magnitude-based pruning, introduced by LeCun et al. (1990). This method removes weights with the smallest absolute values, assuming that smaller weights contribute less to the overall performance of the model. Despite its simplicity, unstructured pruning often leads to irregular sparsity patterns, which can be inefficient for certain hardware accelerators (Guo et al., 2019). While effective in reducing the number of parameters, unstructured pruning is less efficient in terms of hardware acceleration due to the irregular nature of the pruned weights.

Structured pruning, on the other hand, removes entire structural components such as neurons, channels, or attention heads, which makes the model more hardware-friendly (Wang et al., 2019). By maintaining regular patterns of sparsity, structured pruning allows the model to better exploit parallel processing on devices like GPUs and TPUs. For instance, structured pruning has been successfully applied in reducing attention heads in Transformer-based architectures, where entire heads or even layers can be pruned without significantly affecting performance. This method is particularly advantageous in environments where deployment efficiency is a priority, as it maintains a more predictable and manageable architecture during inference (Wang et al., 2019; Xu et al., 2022).

Beyond these traditional approaches, more recent advances have introduced adaptive pruning methods that adjust the pruning criteria based on the model's performance during training or even after training. For example, Reweighted Proximal Pruning (RPP) is designed to iteratively fine-tune the importance of each weight based on its contribution to the final output (Guo et al., 2019). Similarly, contrastive pruning approaches, such as the ContrAstive Pruning (CAP) method introduced by Xu et al. (2022), focus on retaining both task-agnostic and task-specific knowledge in pruned models. These methods use contrastive learning techniques to ensure that the pruned model retains generalizable features while still excelling at fine-tuned tasks.

Wanda pruning, developed by Sun et al. (2023), offers a more sophisticated technique specifically tailored for LLMs. Unlike magnitude-based pruning, which only considers the size of the weights, Wanda also takes into account the activation values of neurons during inference. This dual consideration helps to better preserve the model's performance even at high levels of sparsity. In particular, by pruning weights that have both low magnitude and low activation, Wanda is able to retain the more critical parameters that are actively contributing to the model's output. This method has proven especially useful for LLMs, where retraining large models after pruning is often impractical. Since Wanda does not require retraining, it is an attractive option for large-scale deployments where computational resources are limited.

SparseGPT, introduced by Frantar et al. (2023), takes a different approach by employing structured pruning specifically for GPT-based models. SparseGPT prunes entire attention heads, channels, or layers rather than individual weights. This form of structured pruning generates more regular sparsity patterns, which are better suited for hardware accelerators. One of the key advantages of SparseGPT is that it is a one-shot pruning method, meaning that the model is pruned in a single pass, without the need for iterative retraining. This makes SparseGPT highly suitable for real-time or on-the-fly pruning scenarios, where models need to be compressed quickly for deployment.

Despite the diversity of pruning techniques, they all share a common limitation: the pruning process is typically applied uniformly across all layers or modules in the model. That is, most pruning strategies adopt a single global pruning ratio that is applied consistently to all components of the model, regardless of their individual importance or function. This uniform pruning approach assumes that all layers or modules contribute equally to the model's performance, which is often not the case. For example, research has shown that attention heads in Transformers are far more critical to maintaining the model's overall performance than other components, such as feedforward layers (Michel et al., 2019). As a result, pruning strategies that apply the same rate to all parts of the model may over-prune important modules, leading to significant performance degradation in certain tasks.

One of the primary goals of more recent pruning methods, such as Wanda and SparseGPT, is to address this limitation by applying pruning in a more targeted and module-specific manner. However, even these methods do not fully exploit the potential benefits of module-specific pruning, where different components of the model—such as attention heads and MLP layers—are pruned at different rates based on their relative importance. By developing more refined pruning strategies that account for the unique characteristics and importance of each module, we can potentially achieve better performance while maintaining higher levels of sparsity. This area of research remains an open challenge and offers significant potential for future advancements in LLM compression.

2.4 Related Work

The exploration of pruning techniques has become central to the effort of making Large Language Models (LLMs) more computationally efficient. Given the massive parameter sizes of these models, it is crucial to understand the varying importance of different model components, such as attention heads and multilayer perceptron (MLP) layers, and how they contribute to the overall performance of the model. Recent research highlights that not all components within an LLM are equally important, and different modules exhibit varying degrees of sensitivity to pruning.

2.4.1 Importance of Different Modules in LLMs

The Transformer architecture, upon which most LLMs are based, is composed primarily of two types of modules: attention mechanisms and feed-forward layers (MLPs). Attention mechanisms, especially multi-head attention, have been shown to be vital in capturing long-range dependencies and maintaining the overall performance of the model (Vaswani et al., 2017). Studies have indicated that some attention heads within the Transformer architecture are more critical than others for certain tasks, suggesting that pruning can selectively target less essential attention heads without significant performance degradation (Michel et al., 2019). Michel et al. (2019) introduced the idea that not all attention heads contribute equally to the model's predictive performance, and their work laid the foundation for more granular pruning strategies that differentiate between essential and non-essential heads.

In contrast, MLP layers, which are responsible for non-linear transformations of the input, have been found to be less critical compared to attention heads. Pruning MLP layers can typically be done more aggressively, as shown by Ji et al. (2023). Their study demonstrated that removing a significant portion of the neurons in the MLP layers often results in only minor performance losses. This suggests that while MLP layers contribute to the overall expressive power of the model, they may not be as sensitive to pruning as attention heads. This is especially relevant when considering the computational and memory trade-offs in deploying pruned models on resource-constrained devices, as MLP layers often contain more parameters than attention heads.

However, despite these insights, uniform pruning strategies that apply the same pruning ratio across all modules still dominate much of the research and practice. This one-size-fits-all approach does not account for the varying importance of different modules. Consequently, more advanced pruning strategies that account for module importance are needed to improve the efficiency of LLM pruning.

2.4.2 Module-Specific Pruning in LLMs

Given the varying sensitivity of different modules to pruning, researchers have explored module-specific pruning strategies that apply different pruning rates to distinct components of the model. One such method is differentiable pruning, which enables fine-tuned control over which modules are pruned more aggressively based on their contribution to task performance. For instance, Xu et al. (2022) proposed Contrastive Pruning (CAP), a method that leverages contrastive learning to retain essential knowledge while pruning less critical components. Their approach demonstrated significant improvements in preserving performance on downstream tasks, particularly for models pre-trained on large corpora and fine-tuned for specific tasks.

Additionally, structured pruning techniques like SparseGPT (Frantar et al., 2023) have introduced more hardware-friendly pruning methods. By targeting entire heads, channels, or layers, structured pruning creates more regular patterns of sparsity, leading to better hardware utilization compared to unstructured pruning methods. SparseGPT focuses on pruning the least important components within the model, which aligns well with module-specific pruning strategies that aim to minimize performance degradation while optimizing the model for efficient inference.

Wanda pruning, another notable method, incorporates both weight magnitudes and activation values to determine which parameters to prune (Sun et al., 2023). While traditional pruning techniques only consider the magnitude of the weights, Wanda goes further by evaluating the activation of each neuron, enabling more informed decisions on which parameters can be removed without negatively impacting model performance. This makes Wanda a particularly effective strategy for pruning large models like GPT-based architectures, as it allows for higher levels of sparsity without significant retraining.

2.4.3 Varying Importance of Modules in Different Tasks

Different tasks require different parts of the model to function optimally. For example, in tasks that require reasoning or complex sentence generation, attention heads are typically more crucial. This is because attention mechanisms capture the long-range dependencies that are essential in maintaining coherence across sentences or paragraphs. On the other hand, tasks that involve more straightforward text classification or fact retrieval may rely more on the MLP layers, making it possible to prune attention heads more aggressively without degrading performance.

Michel et al. (2019) were among the first to empirically demonstrate the importance of attention heads, showing that while some heads contribute significantly to performance, others can be pruned without a noticeable loss in accuracy. This finding has been supported by subsequent studies, such as those by Voita et al. (2019) and Correia et al. (2020), who found that redundant attention heads could be pruned away in certain tasks without impacting the model's ability to generate coherent text. These insights are crucial for designing pruning strategies that selectively target less important components while preserving the overall performance of the model.

Moreover, task-specific pruning has become a focal point of LLM research, where models are pruned differently depending on the nature of the downstream task they are fine-tuned for. For instance, models pruned for language generation tasks may retain more attention heads, as these are critical for maintaining the coherence and flow of generated text. In contrast, for classification tasks, the model can afford to prune more attention heads while keeping the MLP layers relatively intact, as these tasks require less contextual understanding.

2.4.4 Challenges in Module-Specific Pruning

Despite the progress in understanding the importance of different modules, there are several challenges that remain in module-specific pruning. One of the main challenges is identifying the optimal pruning ratios for different modules without extensive hyperparameter tuning. The interactions between attention heads and MLP layers can be complex, and pruning one module can affect the performance of another. Furthermore, different tasks may require different pruning strategies, which complicates the design of a one-size-fits-all solution.

Another challenge is ensuring that the pruned model remains efficient when deployed on real-world hardware. While structured pruning methods like SparseGPT create more hardware-efficient models, unstructured pruning can lead to irregular sparsity patterns that are not well-suited for parallel computation on GPUs or TPUs. Therefore, balancing hardware efficiency with model performance is a critical consideration when designing module-specific pruning strategies.

Finally, the lack of task-specific pruning benchmarks makes it difficult to compare different pruning strategies in a standardized way. While perplexity and accuracy remain the primary metrics for evaluating pruned models, these metrics do not always capture the nuances of task-specific performance. New benchmarks that measure the performance of pruned models on a wider variety of tasks, such as knowledge-intensive tasks or reasoning tasks, are needed to advance the field further.

Chapter 3

Experimental Design

This section describes the experimental setup, including the models (Section 3.1) and datasets (Section 3.3) used for evaluation (Section 3.4). The focus of this work is on pruning specific modules, such as MLPs and attention heads, to reduce model complexity while maintaining performance.

3.1 Model

Baseline Model (full-sized Model) In this study, we employed several pre-trained large language models (LLMs) from Hugging Face's official releases, including Meta's LLaMA-2 (7B) and LLaMA-3 (8B), as well as Microsoft's Phi-2. These models are based on the Transformer architecture, which is widely recognized for its performance across a variety of natural language processing tasks.

The core architecture of these LLMs consists of two primary components: Self-Attention and Multilayer Perceptron (MLP) modules. As shown in the figure below, each layer in the LLaMA model is composed of a multi-head self-attention mechanism and a feedforward MLP. The self-attention mechanism is responsible for capturing long-range dependencies within the input sequence, while the MLP performs local transformations, enhancing the model's expressive power.

The internal structure of the attention mechanism includes components such as Query (Q), Key (K), Value (V), and output projection layers. The MLP module is typically composed of layers such as the up-projection and down-projection, which further process the intermediate representations generated by the attention mechanism.

By focusing on pruning key modules like the self-attention and MLP layers, our study aims to reduce model size and complexity while maintaining performance. The modules highlighted in the figure represent the primary targets for our pruning experiments.

Pruned Model For the pruning process, we employed the Wanda pruning method, as described in Section 2. Wanda efficiently prunes less significant weights by considering both

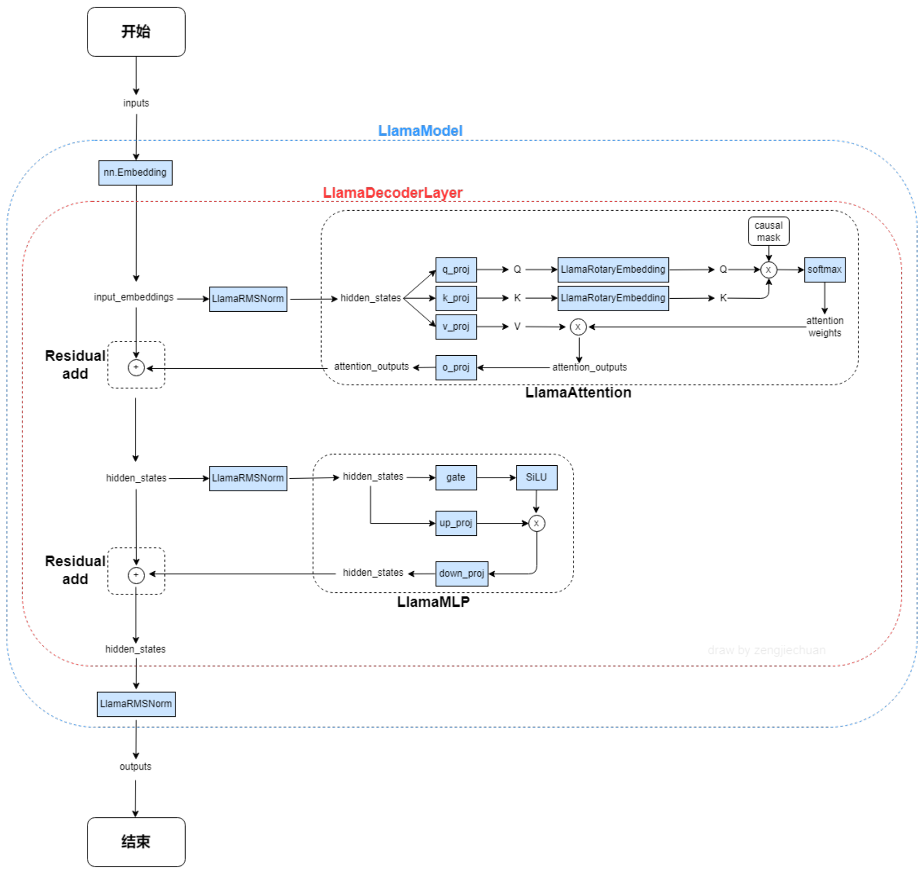

Figure 3.1: LLaMA model architecture

their magnitude and corresponding activation values, thereby inducing sparsity in the model without requiring retraining or significant weight updates.

We applied two primary pruning strategies:

- **Single Module Pruning**: This strategy prunes individual components, such as MLP modules or attention heads, and measures the impact on model performance, particularly using perplexity as a key evaluation metric.
- **Combined Module Pruning**: In this approach, we prune multiple components simultaneously, such as pruning both MLP and attention heads, and analyze how different combinations of pruning rates affect overall model accuracy.

3.2 Datasets

In our experiments, we primarily used the WikiText-2 and C4 datasets to evaluate the performance of the pruned models. These datasets were chosen for their widespread use in language modeling tasks, providing a robust basis for assessing the models' perplexity and other language generation capabilities.

WikiText-2 This dataset is a widely-used corpus for language modeling tasks, containing a rich variety of English texts. WikiText-2 consists of well-curated and clean Wikipedia articles, making it suitable for training and evaluating large-scale language models on text generation tasks.

C4 (Colossal Clean Crawled Corpus) The C4 dataset is a large-scale dataset derived from web pages. It is particularly suited for language models dealing with text generation and comprehension at scale. The diversity of the C4 dataset enables more robust evaluation of the model's ability to generalize across different domains and types of text.

By using these datasets, we are able to assess the model's performance in generating coherent and accurate text. The primary evaluation metric used on these datasets is perplexity (PPL), which measures how well the model predicts the next token in a sequence, reflecting its language modeling capabilities.

3.3 Tasks

To evaluate the performance of the pruned models on downstream tasks, we utilized the *lm eval* framework, a commonly used evaluation toolkit for large language models. *lm eval* provides a standardized set of tasks and metrics for assessing model performance across various natural language processing tasks, ensuring comparability and consistency in evaluation.

The tasks we used cover a range of reasoning, question answering, and factual knowledge assessments, giving a comprehensive evaluation of the pruned models' capabilities. The following table summarizes the tasks and their corresponding model capabilities:

Task	Model Capability
BoolQ	Yes/No Question Answering
RTE	Natural Language Inference
HellaSwag	Commonsense Reasoning
WinoGrande	Pronoun Resolution
ARC-Easy	Science Reasoning (Simple)
ARC-Challenge	Science Reasoning (Challenging)
OpenBookQA	Factual Knowledge

Table 3.1: Downstream tasks and the corresponding model capabilities.

The tasks vary in complexity and the type of reasoning required. For instance, BoolQ involves binary yes/no questions that test a model's understanding of specific contexts, while RTE is a task focused on natural language inference, requiring models to determine whether one statement logically follows from another. Tasks like HellaSwag and WinoGrande emphasize commonsense and contextual reasoning, testing a model's ability to predict probable next steps or resolve pronoun references. ARC-Easy and ARC-Challenge involve multiple-choice science questions of varying difficulty, and OpenBookQA challenges the model's ability to apply factual knowledge in an open-book setting.

These tasks together provide a well-rounded evaluation of the models' capabilities across different reasoning types and domains, helping us understand the impact of pruning strategies on performance in both simple and complex scenarios.

3.4 Evaluation Metrics

To thoroughly evaluate the performance of the pruned models, we employed the following key metrics:

Perplexity This is the primary metric for assessing language modeling tasks. A lower perplexity value indicates better predictive accuracy, as the model is more confident in generating the next token in a sequence.

Accuracy on downstream tasks For downstream tasks, such as text classification and question answering, we used metrics such as accuracy, precision, and recall to gauge the model's performance after pruning.

Sparsity Ratio Although not a direct evaluation metric, the sparsity ratio is critical in understanding the trade-offs between model complexity and performance. By comparing perplexity and task accuracy at different sparsity levels, we can determine how effective the pruning strategy is at maintaining the model's overall performance while reducing its size.

In summary, these evaluation metrics provide a comprehensive view of the impact of pruning on both language modeling and downstream tasks, allowing us to assess the balance between model efficiency and performance.

Chapter 4

Results

4.1 Single Module Pruning Results

The performance of single module pruning is evaluated by analyzing the relationship between pruning rate and perplexity (PPL), as well as the number of pruned weights and PPL. The results demonstrate that pruning different modules leads to varying impacts on the model's perplexity, particularly at higher pruning rates.

Figure 4.1 presents the impact of pruning rates on PPL for LLaMA-2, LLaMA-3, and Phi-2 models. It shows that even though fully pruning certain modules may not significantly reduce the overall parameter count—sometimes resulting in the model's sparsity being less than 10%—the perplexity still increases significantly when the pruning rate exceeds 90%. This suggests that while individual modules may not contribute a large number of parameters, each plays an essential role in maintaining model performance. In other words, despite some modules having a smaller overall size, pruning them at high rates greatly impacts the model's inference capability, demonstrating that each module is indispensable.

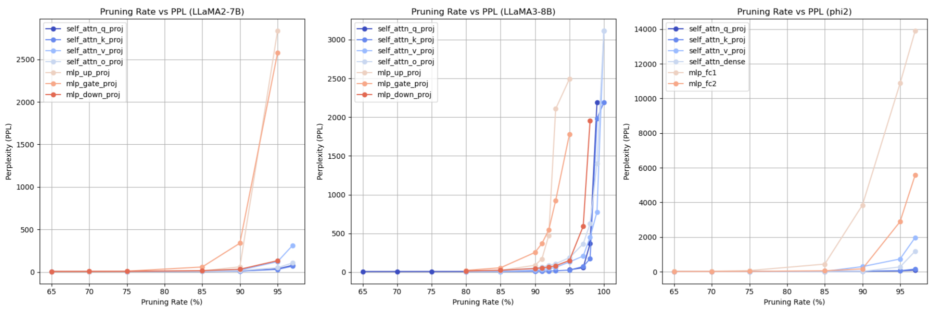

Figure 4.1: Pruning Rate vs Perplexity

However, it is important to note that simply comparing pruning rates does not fully capture the impact on model performance because different modules contribute unequal numbers of parameters to the model. Specifically, MLP modules contain significantly more parameters compared to attention modules, often three to four times as many. This means that when pruning by the same percentage, MLP modules remove more parameters, which can lead to a more substantial drop in performance.

To further investigate this, we analyzed the relationship between the number of pruned weights and PPL, shown in Figure 4.2. As the number of pruned weights increases, we observe that attention modules, particularly V-Proj and O-Proj, cause a sharper increase in PPL when compared to MLP projections. This suggests that while pruning MLP modules reduces a larger number of parameters, attention modules are more sensitive to the loss of parameters on a per-weight basis. Therefore, by analyzing both pruning rate and pruning number, we can conclude that each module has different sensitivities to pruning, depending on both the module type and the amount of parameters removed.

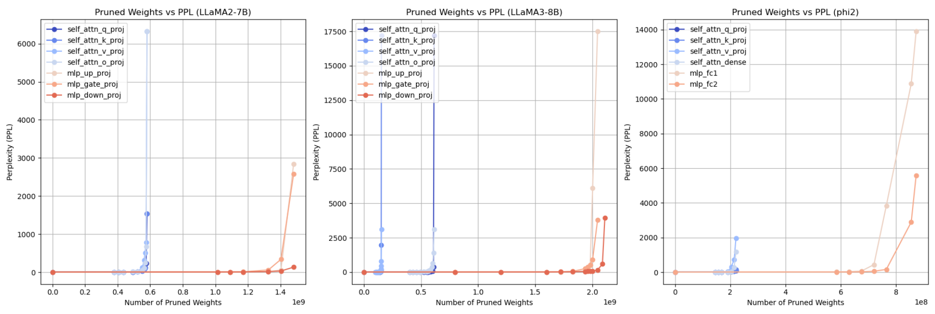

Figure 4.2: Pruned Weights vs Perplexity

4.2 Combined Module Pruning Results

This section evaluates the performance of combined module pruning, focusing on the cumulative effects of pruning both attention and MLP modules. Figure 4.3 replicates the two experiments from the single module analysis, but this time with the combined pruning of attention and MLP modules. The results compare pruning rates and pruned numbers against perplexity (PPL) for the combined modules.

Figure 4.3 shows the comparison between attention and MLP pruning rates. While both types of modules contribute to performance degradation, attention modules appear to handle higher pruning rates more effectively before PPL increases sharply. In contrast, MLP modules show a more significant and faster decline in performance as the pruning rate increases. This observation seems to align with findings from previous studies, such as Ji et al. (1), which suggest that attention modules play a more critical role in preserving model performance. However, this initial observation is complicated by the fact that MLP modules inherently contain more parameters than attention modules, approximately three to four times as many. Therefore, at the same pruning rate, MLP modules lose more parameters, which naturally leads to a more pronounced impact on model performance.

To address this, we analyzed the relationship between the number of pruned weights and PPL for the combined pruning of attention and MLP modules. Figure **??** reveals that when removing an equal number of weights from both module types, attention modules tend to exhibit a larger increase in PPL compared to MLP modules. This result supports the idea that attention modules are more sensitive to weight removal, further highlighting their importance in the overall architecture. Thus, while MLP pruning at higher rates may seem to degrade performance more due to the sheer volume of parameters being pruned, attention modules remain critical for maintaining inference capabilities.

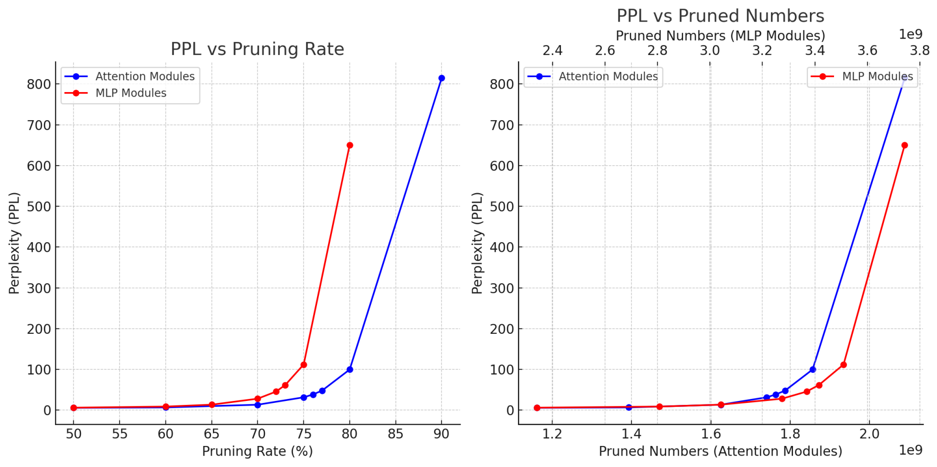

Figure 4.3: Comparison between Attention and MLP Modules

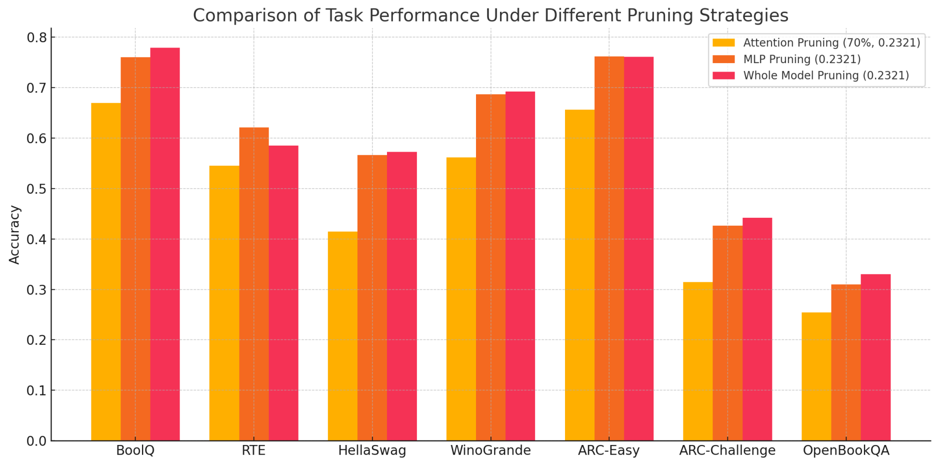

Figure 4.4: Comparison of Downstream Task Performance

Furthermore, we evaluated downstream task performance using the LM Eval framework. Figure 4.4 illustrates the results across various tasks, including reasoning, question answering, and language modeling. The comparison includes three different pruning strategies: attention-only pruning, MLP-only pruning, and uniform pruning across all modules. The results reveal that attention pruning has a noticeable negative effect on performance across all tasks, particularly on reasoning tasks such as ARC-Challenge and RTE. MLP pruning, on the other hand, shows less severe performance drops across most tasks. In fact, in two reasoning tasks, MLP pruning even outperforms uniform pruning, suggesting that MLP modules may not be as crucial for maintaining reasoning performance compared to attention modules. This observation opens up the potential for more optimized pruning strategies where MLP modules can be pruned more aggressively without significantly harming performance, especially in tasks that require reasoning.

Chapter 5

Conclusions and Future Work

This chapter summarizes the findings of the report and outlines potential areas for future research, including more advanced pruning strategies and their application to a wider range of LLM architectures.

Bibliography

[1] Ji, Y., Cao, Y., and Liu, J. Pruning large language models via accuracy predictor. *arXiv preprint arXiv:2309.09507* (2023).

Appendices

Appendix A

An Appendix of Some Kind

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc.

Appendix B

Another Appendix

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc.