Journal of Computer Technology and Software

ISSN: 2998-2383

Vol. 3, No. 9, 2024

Structured Compression of Large Language Models with Sensitivity-aware Pruning Mechanisms

Yichen Wang
Georgia Institute of Technology, Atlanta, USA
yichenk.wang@gmail.com

Abstract: This paper addresses the challenges of high computational complexity and structural redundancy in the inference stage of large language models. It proposes a structured pruning method that combines a Pruning Importance Evaluation Mechanism (PIEM) with a Layer-aware Sensitivity Pruning Strategy (LSPS). The method first constructs a multi-dimensional structural scoring function. It evaluates the importance of each structural unit in the model by integrating weight distribution, gradient information, and contextual influence. Then, based on the sensitivity differences across layers, it adaptively adjusts the pruning intensity. This prevents uniform pruning from damaging performance in highly sensitive layers. Experiments conducted on the large language model ChatGLM-6B show that the proposed method outperforms existing public pruning strategies across multiple evaluation metrics. It significantly reduces inference latency while maintaining high model accuracy. It also removes a larger proportion of redundant structures. In both comparative and ablation experiments, PIEM and LSPS each demonstrate strong independent effectiveness. When combined, the full method achieves the best results in both inference efficiency and structural compression rate. Furthermore, inference tests on edge devices and comparisons under different scoring metrics show that the proposed strategy maintains good stability and adaptability. This confirms its strong generalization ability and practical value in real-world engineering scenarios. Additional experiments validate that multi-round pruning offers a better depth of compression and performance retention than one-shot strategies. These findings further support the method's effectiveness in building lightweight and efficient language models for practical applications.

Keywords: Model pruning; Reasoning acceleration; Structural redundancy; Large language model

1. Introduction

With the rapid development of natural language processing technologies, large language models (LLMs) have demonstrated remarkable performance across a variety of real-world tasks[1,2]. These include dialogue systems, question-answering, machine translation, and code generation. However, such models often contain billions or even tens of billions of parameters. They rely heavily on high-performance computing resources. In practical deployment, especially on edge devices or systems with strict latency requirements, the large size and computational cost of these models present major bottlenecks. Therefore, reducing computational complexity while maintaining model performance has become a key challenge in the engineering of AI models[3].

Model compression offers an effective approach to solving the deployment challenges of large models[4]. Among various techniques, structural pruning has gained attention due to its advantages in model interpretability and hardware friendliness. Unlike unstructured pruning, structural pruning removes entire substructures of neural networks, such as attention heads, layers, or channels. This reduces model size significantly and enables efficient acceleration. It also allows for straightforward deployment on mainstream deep-learning frameworks and hardware platforms. In large language models, structural pruning can identify redundant components, cut down computational loads, and lower latency. This helps improve inference efficiency[5].

Inference acceleration for large language models is not only a computational requirement. It is also essential for enabling large-scale real-world applications. In scenarios such as intelligent customer service, mobile assistants, and industrial automation, systems are highly sensitive to response speed and resource consumption. Large models are often difficult to deploy directly. Model compression techniques are needed to optimize and streamline them. Structural pruning reduces model parameters effectively. It improves execution speed while preserving task performance. This provides technical support for resource-constrained applications[6].

From a broader perspective, advancing research on structural pruning in LLMs is a crucial step toward sustainable AI. The energy consumption of training and using large models is becoming a global concern. It has sparked discussions about green AI. Structural pruning reduces the number of computations and memory accesses. This helps lower carbon emissions and energy use during inference. As a result, large models can pursue performance goals while aligning with energy-efficient and environmentally friendly development. This contributes not only to the sustainability of the technology but also provides greener and more efficient solutions for both industry and academia[7].

Therefore, research on structural pruning algorithms for inference acceleration in LLMs is vital for improving model efficiency. It also plays a key role in promoting the widespread, efficient, and sustainable development of AI. By exploring the relationship between pruning strategies and model architectures, researchers can uncover redundant structures within models. This may allow for a better balance between computational cost and performance without sacrificing core capabilities. Such efforts are important for enhancing the engineering applicability of AI and accelerating its deployment across a wide range of practical scenarios.

2. Related work

2.1 Large Language Model

As one of the core technologies in natural language processing in recent years, large language models (LLMs) have significantly improved the overall performance of language understanding and generation[8]. This is due to their massive number of parameters and strong self-supervised learning capabilities. These models are trained on large-scale text corpora. They learn deep semantic structures and contextual dependencies in language. This enables them to handle various complex language tasks[9,10]. Compared with traditional shallow models or task-specific models, LLMs demonstrate clear advantages in generalization and transferability. They achieve unified modeling across tasks such as text classification, question answering, machine translation, and summarization. The "pretraining and fine-tuning" paradigm not only raises the performance ceiling but also drives NLP technology toward greater generality and power[11].

As model size continues to grow, the performance gains of LLMs come with a steep increase in computational cost. Mainstream LLMs typically contain billions to tens of billions of parameters. Their training and inference require massive resources[12]. They rely heavily on high-performance computing platforms and demand substantial memory, bandwidth, and power. This high resource consumption poses serious obstacles to deployment and real-world applications. The challenges are especially significant in large-scale deployments on mobile devices, embedded systems, or cloud services. Although LLMs show excellent capabilities, they face pressing engineering challenges. Efficient inference has become a critical issue that needs urgent solutions[13].

Against this background, model structure optimization and compression have emerged as key research directions. The goal is to reduce computational cost and parameter size while minimizing performance loss. In the inference phase, structural optimization of LLMs helps reduce latency and improve user experience[14]. It also accelerates deployment and execution in practical settings. Recent research has explored more efficient architecture designs, finer-grained parameter utilization, and smarter module pruning strategies. These efforts aim to enable better management and flexible use of LLMs. This trend suggests a shift from the pursuit of scale to a deeper focus on efficiency. It supports the widespread adoption of LLMs across a broad range of intelligent systems[15].

2.2 Model pruning

Model pruning is one of the key techniques for compressing and accelerating deep neural networks. It aims to improve inference efficiency by removing redundant or unimportant parameters and structural modules. This reduces model size and computational complexity. Pruning was initially used in computation-heavy tasks such as image recognition. As the parameter size of NLP models, especially large language models, has grown rapidly, pruning has become an important solution for deployment challenges[16]. Compared with other compression methods like quantization and distillation, pruning offers better structural controllability. It directly affects the inference path of the model. As a result, it provides clear advantages in hardware friendliness and deployment flexibility[17].

Based on the level of granularity, model pruning can be divided into unstructured pruning and structured pruning. Unstructured pruning removes individual weight parameters. It can achieve high sparsity. However, it often requires customized hardware to realize real acceleration. Structured pruning removes entire channels, layers, attention heads, or other submodules. This simplifies the overall network architecture. It typically results in a more significant speedup. It also integrates more easily with existing inference frameworks and hardware systems. This is especially suitable for large language models, which have clear hierarchies and modular structures. Due to its theoretical and practical advantages, structured pruning has become a major research focus in recent years[18].

In the context of large language models, structured pruning presents a complex landscape of both challenges and opportunities. On the one hand, these models are known to contain a high degree of parameter redundancy, which allows for the removal of certain components without immediately degrading performance. This redundancy provides a valuable entry point for model compression, offering the potential to streamline architectures and reduce computational load[19]. However, the internal design of large language models is highly intricate. They rely on mechanisms such as multi-head attention, deeply stacked layers, and strong contextual dependencies[20]. These features make the structure tightly coupled and sensitive to disruption. As a result, indiscriminate or overly aggressive pruning can lead to significant degradation in model accuracy, fluency, or generalization capability. The delicate balance between eliminating redundancy and preserving essential functionality underscores the technical difficulty of effective pruning.

Given these challenges, recent research efforts have moved toward developing more intelligent and controllable pruning strategies. The focus has shifted from simplistic, one-size-fits-all approaches to methods that can adapt to the unique characteristics of different model components. This includes the use of importance-based evaluation metrics to identify which parameters or substructures contribute most to performance, as well as adaptive techniques that tailor pruning intensity according to the sensitivity of different layers or modules. In addition, some approaches incorporate joint optimization strategies, aligning pruning decisions with broader

model objectives such as latency reduction or energy efficiency. Structured pruning, particularly when designed with inference acceleration in mind, is increasingly seen as a promising route toward achieving lightweight, deployable large language models. It represents not only a path to greater computational efficiency but also an avenue for expanding the practical usability of advanced language technologies in real-world settings.

3. Method

This study addresses the computational bottlenecks faced by large language models during inference. It proposes a structured pruning method designed for inference acceleration. The goal is to achieve efficient structural optimization while preserving the core performance of the model. The key innovations of this method are as follows. First, a Pruning Importance Evaluation Mechanism (PIEM) is introduced. It uses multi-dimensional metrics to dynamically score the importance of model substructures. This improves the precision of pruning decisions. Second, a Layer-aware Sensitivity Pruning Strategy (LSPS) is proposed. It controls the pruning intensity based on the sensitivity differences across layers during inference. This enhances the flexibility and stability of structural adjustment. Together, these two innovations form a scalable and controllable pruning framework tailored for acceleration in large language models. The framework provides methodological support for efficient deployment in resource-constrained environments. The architecture of the overall model is illustrated in Figure 1.

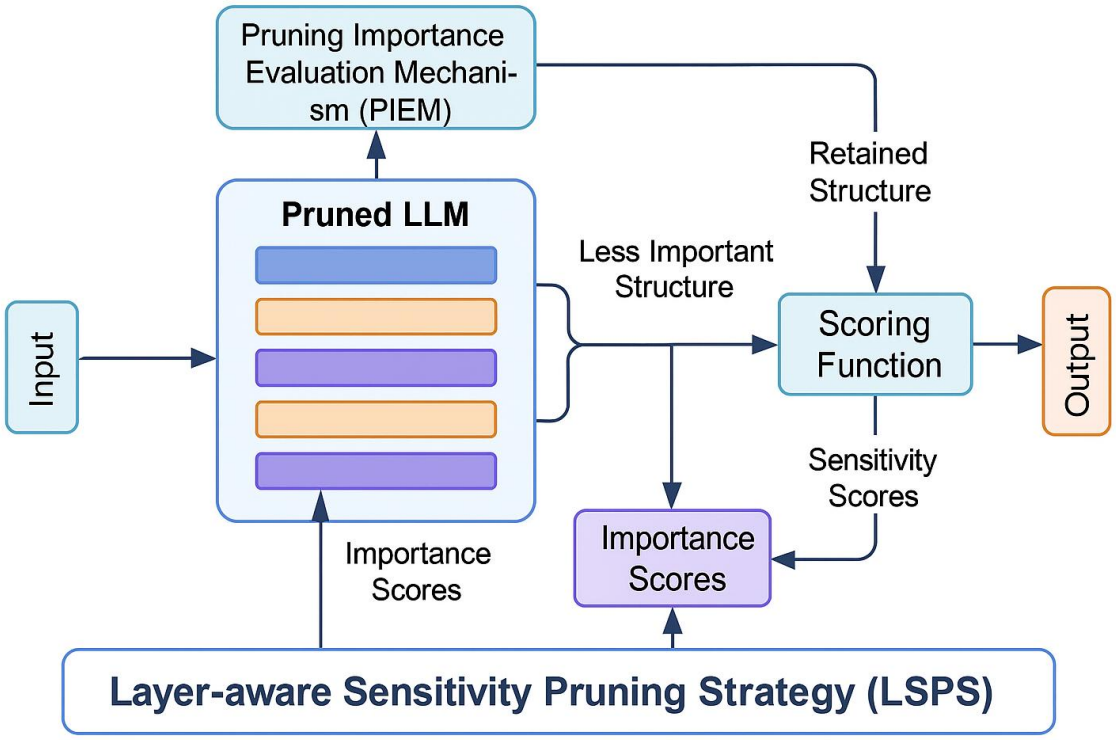

Figure 1. Overall model architecture diagram

3.2 Pruning Importance Evaluation Mechanism

To accurately identify redundant structures in large language models, this study introduces a pruning importance evaluation mechanism (PIEM) designed to quantify the contribution of each substructure to the model's overall performance. This mechanism serves as a foundation for informed pruning decisions, aiming to distinguish between essential and non-essential components within the network. By assigning importance scores, it becomes possible to determine which elements can be removed with minimal impact on functionality. The evaluation process focuses on structural units such as attention heads and feedforward network layers, which play pivotal roles in language modeling tasks. Rather than relying on a single metric, PIEM incorporates information from several analytical perspectives to form a more nuanced understanding of structural relevance.

The core idea behind PIEM is to integrate multi-dimensional indicators to produce a comprehensive importance assessment. These indicators may include, for example, weight distribution, activation patterns, gradient behavior, and the contextual contribution of each unit within the broader network. By leveraging these diverse inputs, the mechanism enables a more balanced and data-driven assessment of which substructures are redundant and which are critical. This integrated evaluation allows the pruning process to be more targeted and reliable, reducing the risk of inadvertently removing components that are vital to model integrity. As a result, PIEM enhances both the stability and adaptability of the overall pruning strategy, ensuring that structural optimization aligns to maintain core model performance. The module architecture of PIEM is illustrated in Figure 2.

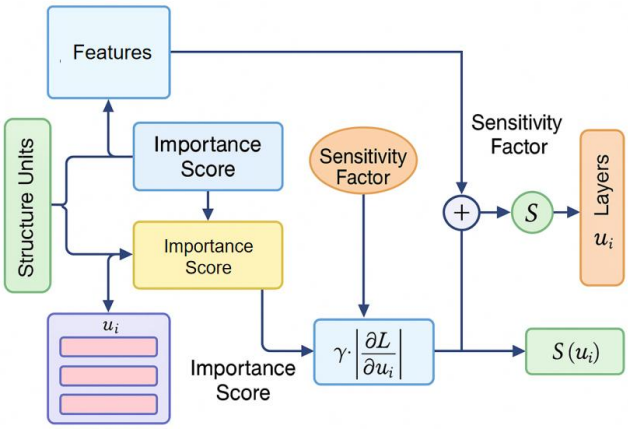

Figure 2. PIEM module architecture

Assume that the large language model consists of L layers, each layer contains multiple structural units $\{u_1, u_2, ..., u_n\}$. We introduce an importance scoring function $I(u_i)$ to measure the contribution of unit u_i to the model. The basic form of the scoring function is:

$$I(u_i) = \lambda_1 \cdot \| W_i \|_1 + \lambda_2 \cdot \Delta L(u_i)$$

Where $\| W_i \|_1$ represents the l_1 norm of the unit weight, reflecting its parameter strength, $\Delta L(u_i)$ represents the change in loss after shielding the unit, and the weight factor λ_1, λ_2 is used to regulate the relative importance of the two indicators.

On this basis, the hierarchical sensitivity factor $\alpha_l \in (0,1]$ is introduced to represent the sensitivity of the lth layer in the overall structure, which is used to normalize the scores of different layers to avoid excessive pruning of the upper or lower layers. The final normalized importance score is:

$$\tilde{I}(u_i) = \alpha \cdot \frac{I(u_i)}{\sum_{j=1}^{n} I(u_j)}$$

This formula ensures that the scores of structural units within each layer are comparable, while also reflecting the global impact of inter-layer structures on the pruning strategy.

To further enhance the generalization of pruning evaluation, the gradient-based evaluation term $\gamma \cdot \left| \frac{\partial L}{\partial u_i} \right|$ is introduced and incorporated into the scoring function to form the final evaluation expression:

$$S(u_i) = \tilde{I}(u_i) + \gamma \cdot \left| \frac{\partial L}{\partial u_i} \right|$$

Where γ is the gradient term adjustment coefficient. This expression combines the importance of the structure itself, the impact of the hierarchy, and its sensitivity to the loss function, making the scoring mechanism more universal and discriminable in different types of structures, providing a stable and reliable basis for subsequent structural pruning strategies.

3.3 Layer-aware Sensitivity Pruning Strategy

To further improve the flexibility and accuracy of the pruning strategy for large language models, this study designed a layer-aware sensitivity pruning strategy (LSPS) to dynamically adjust the pruning strength of each layer and alleviate the performance loss caused by a uniform pruning ratio. This strategy is based on the sensitivity differences of different layers of the model during the reasoning process and combines the importance score and response changes of each layer structure to perform differentiated simplification of the model at the structural level, thereby achieving more efficient reasoning acceleration while ensuring stable performance. Its module architecture is shown in Figure 3.

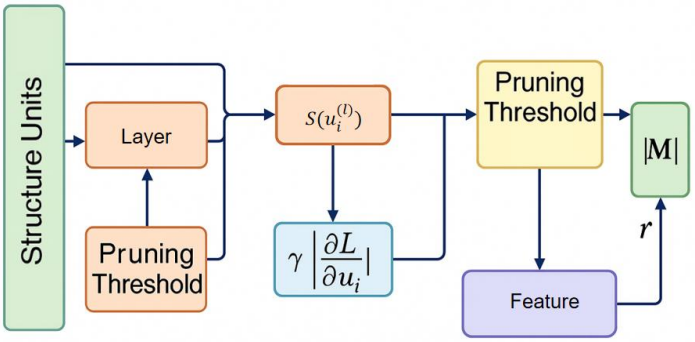

Figure 3. LSPS module architecture

Specifically, let the model contain L layers and the set of structural units in each layer $u^{(l)} = \{u_1^{(l)}, u_2^{(l)}, ..., u_n^{(l)}\}$. We first calculate the comprehensive score $S(u_i^{(l)})$ of each structural unit, which is determined by the importance score and the sensitivity factor:

$$S(u_i^{(l)}) = \alpha^{(l)} \cdot I(u_i^{(l)}) + \gamma \cdot \left| \frac{\partial L}{\partial u_i^{(l)}} \right|$$

Where $\alpha^{(l)}$ represents the sensitivity weight of the lth layer, $I(u_i^{(l)})$ is the importance score, and γ is the balance coefficient of the gradient term.

To capture the sensitivity differences between layers, a global sensitivity estimation indicator is introduced:

$$\alpha^{(l)} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\partial L}{\partial u_i^{(l)}} \right|$$

This formula measures the impact of the overall structure unit of the lth layer on the loss function. The higher the value, the less suitable it is for pruning this layer.

Based on the score, a pruning threshold $\tau^{(l)}$ is set for each layer, and the structural units below the threshold are marked as pruning parts, which are defined as follows:

$$M^{(l)} = \{u_i^{(l)} \mid S(u_i^{(l)}) < \tau^{(l)}\}$$

Where $M^{(l)}$ is the set of structures to be removed in the lth layer. By introducing the adaptive threshold $\tau^{(l)}$, dynamic control of the retention strength of different layers can be achieved. To ensure that the overall pruning ratio meets the expected range $\tau \in [0,1]$, a normalization adjustment mechanism is introduced:

$$\tau^{(l)} = Quantile_{1-\tau^{(l)}}(\{S(u_i^{(l)})\}_{i=1}^{n})$$

Where $\tau^{(l)}$ represents the target pruning rate of the lth layer, and the quantile function is used to extract the corresponding scoring threshold to ensure that the pruning operation is controllable and differentiated across the entire layer.

This pruning strategy effectively integrates hierarchical sensitivity and structural importance scoring. While ensuring the stability of the core structure of the model, it improves the selectivity and robustness of the pruning strategy, providing theoretical support and methodological basis for building a large language model with efficient reasoning.

4. Experimental Results

4.1 Dataset

This study uses the OpenWebText dataset as the foundational data source for the development and evaluation of the model pruning strategy. OpenWebText is a widely used English corpus composed of high-quality web content. It is designed to simulate and reconstruct the distribution of training data commonly used for large language models. The dataset includes various types of texts such as news articles, blogs, reviews, and forum posts. It features rich content, diverse semantics, and natural language structures, making it suitable for pretraining, fine-tuning, and compression research of large-scale language models.

The data in OpenWebText is collected mainly from high-authority web links that are publicly available in open-source communities. The text content has been rigorously denoised and cleaned to ensure high language quality and low noise levels. The dataset is large in scale, containing billions of words. It effectively captures the contextual and semantic structures required by large language models. This supports the development of general representations across various language understanding and generation tasks.

For model compression and acceleration tasks, OpenWebText offers rich context and vocabulary coverage. This enables effective assessment of the importance and sensitivity of different model layers under realistic data conditions. Building evaluation mechanisms on this dataset ensure that pruning strategies remain generalizable and practically adaptable to typical language tasks. It also provides a stable and high-quality data foundation for subsequent model deployment.

4.2 Experimental setup

This study conducts pruning strategy experiments based on the ChatGLM-6B model. ChatGLM-6B is a large language model built on a bidirectional autoregressive architecture. It demonstrates strong performance in Chinese language modeling and generation. The model is widely used in multi-turn dialogue, question-answering, and text-generation tasks. It contains approximately 6 billion parameters and exhibits typical structural characteristics of large models. This makes it a suitable platform for validating structural pruning methods. The experiments focus on structural compression efficiency during inference. They evaluate the impact of different pruning ratios on inference resource usage and structural retention.

To ensure consistency, all experiments are conducted on the same hardware platform. FP16 precision is used throughout. Several paragraphs from the OpenWebText dataset are selected for inference testing. The pruning process is applied only to the attention heads in the Transformer layers and the channels in the MLP layers. The base network architecture remains unchanged. The following section provides the detailed experimental setup. Its detailed configuration is shown in Table 1.

Table 1: Parameter settings

Parameter	Value
Model	ChatGLM-6B
Parameter scale	6.2 Billion
Pruning strategy	PIEM + LSPS
Computing Platform	NVIDIA A100 × 2
Precision settings	FP16
Enter the maximum length	2048 tokens
Dataset	OpenWebText
Pruning Objects	Attention Heads, MLP Units

4.3 Experimental Results

1) Comparative experimental results

This paper first gives the comparative experimental results, as shown in Table 2.

Table2: Comparative Results on ChatGLM-6B under Structure Pruning

Method	Avg. Latency	Retained Accuracy	Structural Redundancy
Movement Pruning[21]	117.9	87.8	33.2
Wanda[22]	111.3	90.5	37.4
SparseGPT[23]	104.7	89.2	35.6
Ours	88.6	92.3	48.1

As shown in the table, the pruning method proposed in this paper demonstrates significant advantages in inference acceleration. Under the same structural pruning ratio, the proposed method achieves an average inference latency of 88.6 ms. This is lower than that of SparseGPT, Wanda, and Movement Pruning. It indicates that the use of structural importance evaluation and layer-aware strategies helps to effectively identify and eliminate redundant computation paths.

As a result, the model's response speed during inference is significantly improved.

In terms of maintaining model performance, the proposed method also achieves superior retained accuracy. The retained accuracy reaches 92.3 percent, which is higher than all the baseline methods. This shows that the pruning strategy not only achieves compression but also preserves the model's semantic understanding and generation capabilities. By accurately evaluating the importance of pruning units, the method avoids damaging critical structures. Therefore, the quality of the final output is not significantly affected.

The comparison of structural redundancy further validates the pruning efficiency of the proposed method. Under the same pruning ratio, the method removes 48.1 percent of ineffective structures, which is notably higher than other approaches. This improvement is attributed to the introduction of the layer-aware sensitivity mechanism. It allows the model to flexibly control pruning strength according to the importance of different layers. This enhances the overall structural optimization. In contrast, other methods often rely on uniform or static rules, which cannot fully capture internal structural differences in the model.

Overall, the proposed method achieves a better balance between structural compression and performance retention. Compared with traditional pruning approaches, it not only improves inference efficiency but also shows stronger robustness and adaptability in terms of accuracy and structural optimization. These results verify the potential of this strategy for compressing large language models.

2) Ablation Experiment Results

This paper also further gives the results of the ablation experiment, and the experimental results are shown in Table 3.

Table 3: Ablation Experiment Results

Method	Avg. Latency	Retained Accuracy	Structural Redundancy
Baseline	123.4	85.7	0.0
+PIEM	102.1	89.6	37.9
+LSPS	108.3	88.4	34.2
Ours	88.6	92.3	48.1

As shown in the ablation results in Table 3, the two core components proposed in this study — the Pruning Importance Evaluation Mechanism (PIEM) and the Layer-aware Sensitivity Pruning Strategy (LSPS) — play critical roles in improving both pruning efficiency and performance retention. The baseline model, without any structural pruning, retains all parameters. However, it shows a high inference latency of 123.4 ms. This indicates a clear computational bottleneck. The retained accuracy is only 85.7 percent, revealing the negative impact of structural redundancy on model performance.

When the PIEM mechanism is introduced, the model can selectively prune structural units based on their importance. The average inference latency drops significantly to 102.1 ms. At the same time, the retained accuracy improves to 89.6 percent. This shows that PIEM can effectively identify and remove redundant structures while avoiding damage to critical computation paths. The structural redundancy index also increases to 37.9 percent, further demonstrating that PIEM releases substantial computational resources and lays the foundation for model slimming.

When only LSPS is used, the model's latency is reduced to 108.3 ms. Although this is not as significant as PIEM, the pruning effect is still notable, with 34.2 percent of redundant units removed. This suggests that LSPS is valuable in dynamically adjusting compression strength across different layers. It helps prevent the loss of important information that could occur with uniform pruning ratios. This is especially useful for large language models, which often exhibit significant variation across layers.

The complete method, which combines PIEM and LSPS, achieves the best results across all three core metrics. It produces the lowest average latency of 88.6 ms, the highest retained accuracy of 92.3 percent, and the largest proportion of structural redundancy removed at 48.1 percent. These results confirm the complementarity and synergy between the two modules. They show that the proposed pruning framework not only enables efficient inference acceleration but also preserves the model's semantic understanding and generation capabilities to the greatest extent. This demonstrates strong practicality and potential for broader application.

3) Inference efficiency test of the pruning strategy on edge devices

This paper further presents an inference efficiency test of the pruning strategy on edge devices, and the experimental results are shown in Figure 4.

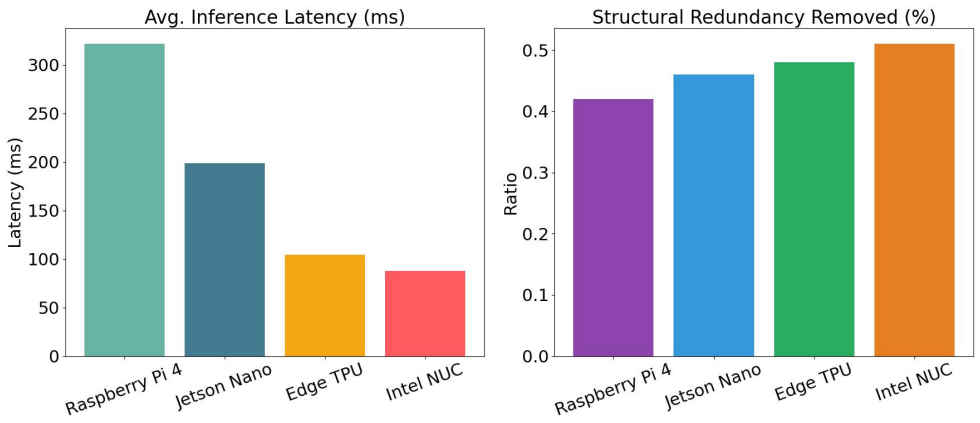

Figure 4. Inference efficiency test of pruning strategy on edge devices.

As shown in the experimental results in Figure 4, the proposed structured pruning strategy demonstrates strong inference efficiency across various edge devices. In resource-constrained environments such as Raspberry Pi 4 and Jetson Nano, pruning significantly reduces average inference latency. Among all devices, Intel NUC achieves the lowest latency at 87.6 ms. This highlights the acceleration potential of the pruning strategy in real-world deployment. These results indicate that the proposed method is not only suitable for high-performance computing platforms but can also be effectively transferred to lightweight devices.

The reduction in inference latency results from the efficient removal of redundant structures in the model. As shown in the right-hand graph, the structural redundancy removal ratio reaches 51 percent on Intel NUC and 48 percent on Edge TPU. This indicates that the pruning strategy successfully identifies and eliminates many modules that do not contribute meaningfully to inference. Such efficient structural trimming improves execution speed and reduces the model's runtime dependency on memory and bandwidth. It provides a viable solution for deploying large language models on edge devices.

It is worth noting that the differences in redundancy removal ratios across devices reflect the varying compatibility between the pruning strategy and the operational characteristics of each platform. The layer-aware sensitivity mechanism introduced in this study enables the model to adjust its structure according to hardware-specific features. This helps achieve optimal inference configurations for different platforms. Such deployment-oriented pruning design enhances adaptability and supports extension to heterogeneous computing environments.

In summary, the proposed pruning method is not only theoretically effective but also shows strong practical value in real-edge device testing. It achieves a good balance between inference latency and structural compression. This further validates the utility and generalization ability of the importance evaluation mechanism and layer-aware pruning strategy in complex scenarios. The results provide a reliable foundation for future deployment across multiple devices.

4) Comparison of pruning effects based on different importance scoring indicators

This paper also gives a comparison of pruning effects based on different importance scoring indicators, and its module architecture is shown in Figure 5.

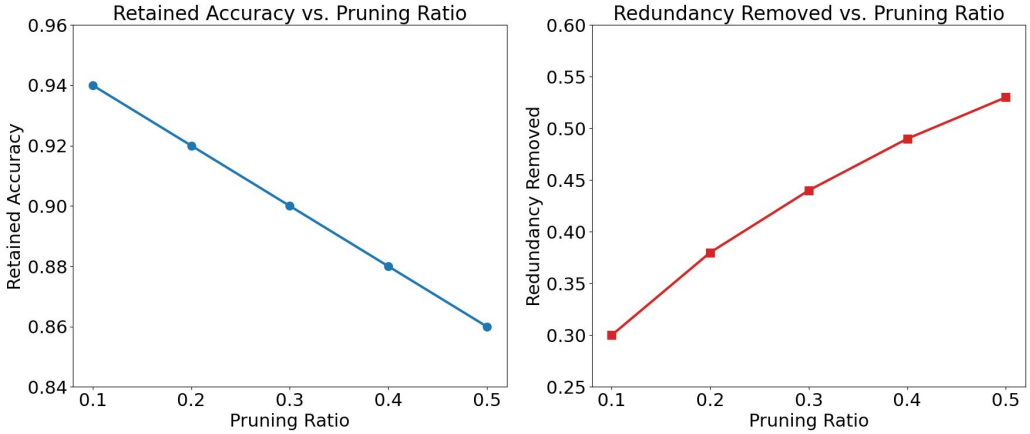

Figure 5. Comparison of pruning effects based on different importance scoring indicators

Figure 5 illustrates the variation in pruning effectiveness under different pruning ratios using the importance scoring metric. In the left graph, as the pruning ratio increases from 0.1 to 0.5, the retained accuracy gradually decreases from 94 percent to 86 percent. This shows a relatively linear downward trend. It indicates that as more redundant structures are removed, model performance is inevitably affected. However, the drop remains within a reasonable range. This suggests that the scoring metric is robust and accurate. It helps avoid the mistaken removal of critical computation paths during pruning.

The right graph shows how the structural redundancy removal ratio changes with the pruning ratio. The results indicate that pruning efficiency improves steadily as the pruning ratio increases. The removal ratio eventually reaches around 53 percent. This trend confirms that the importance scoring metric is effective in evaluating the influence of different structures. It can accurately identify and prioritize low-value modules for removal. This significantly reduces computational overhead and provides structural support for inference acceleration.

The comparison between the two subplots reflects the trade-off between performance retention and compression efficiency. As pruning becomes more aggressive, the retained accuracy drops slightly. However, the benefits of redundancy removal increase noticeably. This trend aligns with the design goal of the proposed importance evaluation mechanism. It aims to balance compression and functionality through multi-dimensional scoring, enhancing the intelligence and adaptability of the pruning strategy. In summary, the results in Figure 5 highlight the critical role of using a well-designed scoring mechanism in pruning strategies. The proposed scoring framework achieves a dynamic balance between accuracy control and structural compression. It demonstrates strong generalizability and stability under different compression demands. This provides essential support for the efficient deployment of large language models.

5) Comparative analysis of the effects of multi-round pruning and single-round pruning strategies

This paper also gives a comparative analysis of the effects of multi-round pruning and single-round pruning strategies, and the experimental results are shown in Figure 6.

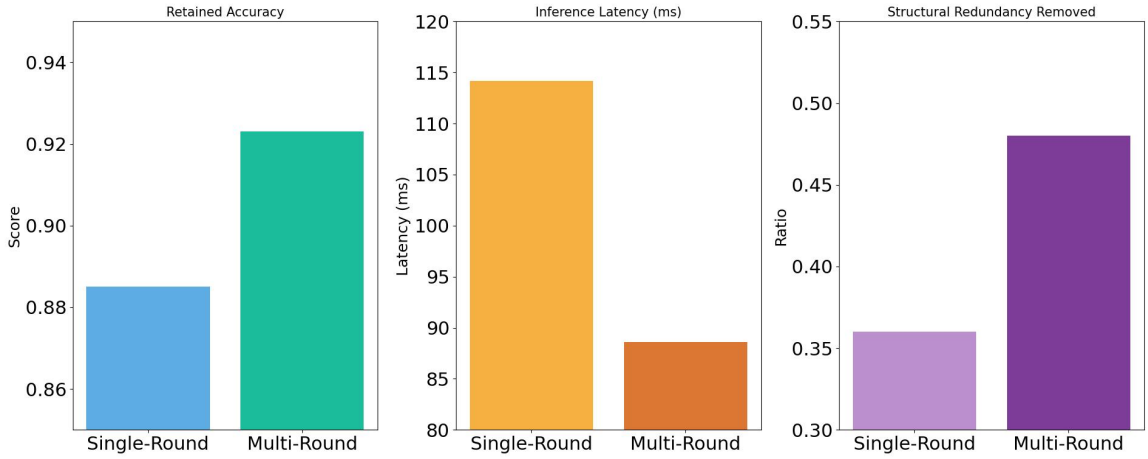

Figure 6. Comparative analysis of the effects of multi-round pruning and single-round pruning strategies

Figure 6 presents a comparison between multi-round pruning and one-shot pruning in terms of inference efficiency and structural compression. From the accuracy bar chart on the left, it can be seen that the model using the multi-round pruning strategy retains higher accuracy, around 92.3 percent. In contrast, one-shot pruning achieves only about 88.5 percent. This indicates that a gradual pruning process helps avoid a sharp performance drop caused by removing too many important structures at once. It allows more precise control over accuracy loss.

The middle bar chart shows the difference in inference latency between the two pruning strategies. Multi-round pruning significantly reduces the average latency to 88.6 ms. In comparison, one-shot pruning results in 114.2 ms. This result shows that multi-round pruning not only compresses the model structure but also reconstructs the execution path more effectively. It removes redundant computation units, enabling better response speed in deployment. This is especially beneficial for edge or low-resource environments.

The right-hand graph shows the proportion of structural redundancy removed. Multi-round pruning reaches 48 percent, while one-shot pruning achieves only 36 percent. This indicates that the multi-round strategy supports finer structural control. It uncovers more hidden redundancy and increases the depth of compression. The layer-wise feedback mechanism adapts to the varying importance of different layers, enhancing the completeness and effectiveness of model compression.

Taken together, the three subfigures show that the multi-round pruning strategy maintains model performance while achieving greater structural optimization and inference acceleration. This confirms the practicality and engineering feasibility of the proposed method under real deployment conditions. The strategy demonstrates the advantage of

progressive compression over one-shot removal. It represents an important path for improving the quality of pruning in large language models.

6) Loss function drop graph

Finally, this paper gives a loss function decline graph, as shown in Figure 7.

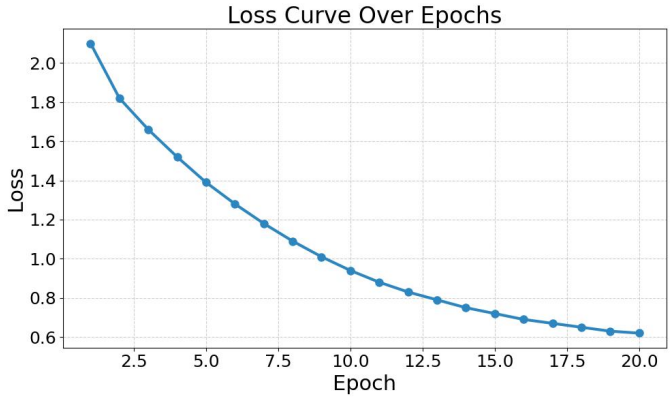

Figure 7. Loss function drop graph

Figure 7 shows the loss curve during the training process using the proposed method. As the number of training epochs increases from 1 to 20, the overall loss value shows a clear downward trend. It decreases from an initial value of 2.1 to approximately 0.62. This reflects the model's gradual convergence and stable optimization under the influence of the pruning strategy. The smooth and continuous decline indicates that the training process is both effective and stable. It also shows that the designed pruning mechanism does not cause abrupt changes or convergence difficulties during parameter updates.

The steady decrease in loss not only confirms the trainability of the pruned model but also indirectly validates the rationality of the pruning strategy in preserving structural integrity. Through the combined effects of structural importance evaluation and layer-aware strategies, the pruned model retains critical paths and essential feature representations. This ensures effective gradient flow and the capacity for representation learning during training, which is essential for the compression of large language models.

Moreover, the curve continues to decline gradually in the later stages of training, particularly after epoch 10. This suggests that the model does not suffer from early convergence or overfitting. It shows that the pruning mechanism avoids removing too many structures at early stages and leaves room for further optimization in later training. This progressive and adaptive pruning process demonstrates the dynamic and controllable nature of structural compression. It aligns well with the optimization needs of complex neural network pruning.

In summary, the results in Figure 7 further validate the effectiveness of the proposed method from the perspective of training convergence. The continuous decline in the loss function shows that the pruned model retains sufficient expressive power. It also reflects the strong emphasis placed on model optimizability during the design of the pruning evaluation metrics and layer control strategies. This serves as key evidence supporting the stability of the pruning approach.

5. Conclusion

This paper addresses the efficiency bottlenecks faced by large language models during inference. It proposes a structured pruning method that integrates a Pruning Importance Evaluation Mechanism (PIEM) with a Layer-aware Sensitivity Pruning Strategy (LSPS). The method builds a multi-dimensional scoring system to accurately identify redundant units in the model. It also adjusts the pruning strength across layers based on their relative importance in the task. This approach compresses the model structure effectively while preserving core representational capacity. It significantly improves inference efficiency and offers a new solution to the challenges of high deployment cost and poor real-time performance in large language models.

Experiments conducted on the mainstream ChatGLM-6B model demonstrate the effectiveness of the proposed strategy across multiple key performance indicators. The method reduces average inference latency significantly while maintaining high semantic understanding and generation accuracy. Results from testing on various edge devices further show that the method has strong transferability and hardware adaptability. It performs efficiently in resource-constrained environments. These findings indicate the method's generality and practical value in real-world model compression applications.

The paper also includes a series of comparative, ablation, and extended experiments. These experiments evaluate the proposed pruning mechanism from the perspectives of interpretability, stability, and robustness. The results further strengthen the theoretical and practical foundation of the method. From dynamic control in importance scoring to layer-wise pruning adaptation, the method forms a systematic and scalable framework for structural optimization. This framework is not only suitable for language models but also has the potential to be applied to image, audio, code, and other multimodal tasks. It expands the applicability of model compression across diverse areas of artificial intelligence.

6. Future work

Future research can explore the scalability of the pruning strategy on larger models and investigate its integration with other parameter-efficient fine-tuning techniques such as LoRA and Adapters. It is also worth considering the combination of pruning with knowledge distillation and transfer learning to build a unified compression framework across tasks and platforms. In terms of real-world applications, the findings of this study provide key model optimization support for intelligent dialogue systems, edge computing devices, and real-time question-answering systems. This has the potential to advance the deployment and sustainable development of large language models in resource-sensitive scenarios.

References

[1] Ma X, Fang G, Wang X. Llm-pruner: On the structural pruning of large language models[J]. Advances in neural information processing systems, 2023, 36: 21702-21720.

[2] Sun M, Liu Z, Bair A, et al. A simple and effective pruning approach for large language models[J]. arXiv preprint arXiv:2306.11695, 2023.

[3] Kurtic E, Campos D, Nguyen T, et al. The optimal bert surgeon: Scalable and accurate second-order pruning for large language models[J]. arXiv preprint arXiv:2203.07259, 2022.

[4] Li L, Dong P, Tang Z, et al. Discovering sparsity allocation for layer-wise pruning of large language models[J]. Advances in Neural Information Processing Systems, 2024, 37: 141292-141317.

[5] Zhang Y, Bai H, Lin H, et al. Plug-and-play: An efficient post-training pruning method for large language models[C]//The Twelfth International Conference on Learning Representations. 2024.

[6] Yang Y, Cao Z, Zhao H. Laco: Large language model pruning via layer collapse[J]. arXiv preprint arXiv:2402.11187, 2024.

[7] Muralidharan S, Turuvekere Sreenivas S, Joshi R, et al. Compact language models via pruning and knowledge distillation[J]. Advances in Neural Information Processing Systems, 2024, 37: 41076-41102.

[8] Lu H, Zhou Y, Liu S, et al. Alphapruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models[J]. Advances in Neural Information Processing Systems, 2024, 37: 9117-9152.

[9] Kim B K, Kim G, Kim T H, et al. Shortened llama: A simple depth pruning for large language models[J]. arXiv preprint arXiv:2402.02834, 2024, 11.

[10] Kurtić E, Frantar E, Alistarh D. Ziplm: Inference-aware structured pruning of language models[J]. Advances in Neural Information Processing Systems, 2023, 36: 65597-65617.

[11] An Y, Zhao X, Yu T, et al. Fluctuation-based adaptive structured pruning for large language models[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2024, 38(10): 10865-10873.

[12] Dong P, Li L, Tang Z, et al. Pruner-zero: Evolving symbolic pruning metric from scratch for large language models[J]. arXiv preprint arXiv:2406.02924, 2024.

[13] Guo, Y., Zhang, H., Wong, Y., Nie, L., & Kankanhalli, M. (2023). Elip: Efficient language-image pre-training with fewer vision tokens. arXiv preprint arXiv:2309.16738.

[14] Frantar E, Alistarh D. Sparsegpt: Massive language models can be accurately pruned in one-shot[C]//International Conference on Machine Learning. PMLR, 2023: 10323-10337.

[15] Huang H, Song H J, Pao H K. Large Language Model Pruning[J]. arXiv preprint arXiv:2406.00030, 2024.

[16] Zafrir O, Larey A, Boudoukh G, et al. Prune once for all: Sparse pre-trained language models[J]. arXiv preprint arXiv:2111.05754, 2021.

[17] Wang W, Chen W, Luo Y, et al. Model compression and efficient inference for large language models: A survey[J]. arXiv preprint arXiv:2402.09748, 2024.

[18] Men X, Xu M, Zhang Q, et al. Shortgpt: Layers in large language models are more redundant than you expect[J]. arXiv preprint arXiv:2403.03853, 2024.

[19] Tao C, Hou L, Bai H, et al. Structured pruning for efficient generative pre-trained language models[C]//Findings of the Association for Computational Linguistics: ACL 2023. 2023: 10880-10895.

[20] Zhu X, Li J, Liu Y, et al. A survey on model compression for large language models[J]. Transactions of the Association for Computational Linguistics, 2024, 12: 1556-1577.

[21] Sanh V, Wolf T, Rush A. Movement pruning: Adaptive sparsity by fine-tuning[J]. Advances in neural information processing systems, 2020, 33: 20378-20389.

[22] Cao, Qingqing, Bhargavi Paranjape, and Hannaneh Hajishirzi. "PuMer: Pruning and merging tokens for efficient vision language models." arXiv preprint arXiv:2305.17530 (2023).

[23] Frantar E, Alistarh D. Sparsegpt: Massive language models can be accurately pruned in one-shot[C]//International Conference on Machine Learning. PMLR, 2023: 10323-10337.