IMPROVED METHODS FOR MODEL PRUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Model pruning is presented as a performance optimization technique for large language and vision models. This technique aims to identify and remove neurons, connections unlikely to lead to the contribution during the machine generation phase. Our goal is to obtain a much smaller and faster foundational model that can quickly generate AIGC content almost as good as those of the unpruned models. Through careful analysis of the weights, bias, activations and other potential indicators, an improved algorithm based on new indicators have been proposed and evaluated. Emprical results show that our proposed algorithm achieves significant improvement in different pruning ranges over previous STOAs.

1 INTRODUCTION

Large language and vision models are facing significant performance challenges due to the massive model size and query loads the system need to support. These models, along with related large production systems, are responbible for crawling, analyzing, and incorporating billions of web pages, videos, and multimodal data into their underling network architectures such as transformer, diffusion et al. One crucial cost factor is the query processing per user, which must scale with both data size and query load. As a result, large foundational models devote substantial hardware and energy resources to this kind of generation task. There has been extensive research on improving query processing performance, including work on various caching techniques, retrieval information systems, and high-performance knowledge representation. A large category of optimization techniques commonly referred to as index or model pruning has emerged in the context of efficient & effective AIGC content generation process. This paper aims to explore and contribute to the understanding and improvement of these pruning techniques to enhance the performance and efficiency of those models.

In this paper, our attention is directed towards a particular optimization technique known as model pruning. In essence, the approach involves conducting a suitable analysis of the learning representation, network designs, and performance study. The objective is to determine those neurons or connections that are highly likely to yield good contributions in reponse to user input. Subsequently, any other neurons that are unlikely to contribute to effective machine generation output are removed from the original neural network. The aim is to obtain a much smaller and faster neural network with a reduced amount of parameters. This pruned network can achieve almost the same quality of machine generation output as the unpruned ones while requiring much less CPU, memory and GPU footprints. Consequently, it leads to faster query processing over a pruned neural network with optimized layers.

(Give an example with numbers to tell the necessaristy for model pruning)

Previous work on model pruning for large language & vision models has primarily focused on approaches such as retaining layers above a global impact threshold or keeping high-scoring neurons in each layer. For detail, we refer to [Cite Wanda paper][Cite SparseGPT paper][Cite Manitude paper]. These efforts have yielded promising results at certain pruning ranges, but obviously there is room for further optimization. The goal of this paper is to build on existing work and develop a methodology (if possible) that combines different indicators to achieve a much better balance between neural network size and generation quality as measured by standard retrieval evaluation metrics. Given a relatively feature rich environment, pruning is considered as a prediction problem to determine, say which neuron, weights or layers to keep.

The remainder of this paper is organized as follows. In Section 2, we provide background information on learning representation, neural networks, and related pruning technique. We summarize our key contributions in Section 3, highlighting the novelty and significance of our approach. We delve into the technical details of our proposed approach in Section 4, providing a comprehensive explanation of the methodology and algorithms developed. We present and explan our experimental results in Section 5, along with some implementation detail and performance analysis. In section 6, we try to give concluding remarks, summarizing the main findings from us and suggesting potential directions for future research.

2 Background and Related Work

In this section, we first provide some background on neural network architectures, user inputs, pruning, and machine generation. We then discuss previous work related to model pruning in the context of large language and vision models. For additional details on general neural network architectures, we refer to [][].

2.1 Background

2.1.1 Nerual Network Architectures

(TODO:)

2.1.2 Human Input & Machine Generation

(TODO:)

2.1.3 Model Quantization & Compression

(TODO:)

2.1.4 Model Pruning & Indicators Discovery

(TODO:)

2.2 Related Work

2.2.1 Typical Pruning Algorithms

There are three typical network pruning algorithms. (1) The magnitude pruning algorithm[18, 19]: Simplest approach: Prunes weights based purely on their absolute magnitude. Threshold-based: A global threshold is determined based on the desired sparsity ratio. Weights below this threshold are set to zero. Unstructured: Can prune individual weights anywhere in the matrix, potentially leading to irregular sparsity patterns that might not be hardware-friendly. Fast but less accurate: Generally the fastest method, but might remove important connections, leading to a larger accuracy drop compared to more sophisticated methods. (2) The WANDA(Weights and Activations) pruning algorithm[3]: Importance-aware: Considers both weight magnitudes and activation statistics to estimate weight importance. Calibration phase: Requires a calibration step where the model processes a small dataset to collect activation data. Row-wise scaling: Normalizes weight magnitudes within each row based on activation statistics, making the pruning less sensitive to weight scale variations across neurons. Unstructured or structured: Can be applied in an unstructured manner (pruning individual weights) or a structured manner (pruning within blocks of weights). Improved accuracy: Often achieves better accuracy-sparsity trade-offs compared to magnitude pruning. (3) The SparseGPT pruning algorithm[5]: Gradient-based: Leverages gradient information during pruning to identify less important connections. Iterative pruning: Prunes the model iteratively, gradually increasing sparsity while minimizing accuracy loss. Block-sparse structure: Encourages a block-sparse structure, which can be more hardware-efficient for some architectures and libraries. Computationally intensive: Can be more computationally expensive than magnitude or WANDA pruning due to the iterative nature and gradient calculations. State-of-the-art results: Often achieves very high sparsity levels with minimal accuracy degradation, making it suitable for compressing large language models. In summary, Magnitude pruning is the simplest and fastest but might be less accurate. WANDA improves upon magnitude pruning by considering activation information, potentially leading to better accuracy. SparseGPT is a more advanced method that uses gradient information and iterative pruning to achieve high sparsity with minimal accuracy loss, but it comes with higher computational cost.

2.2.2 Query Traces & Model Calibration

(TODO:)

2.2.3 Model Entropy & Perplexity Estimation

(TODO:)

2.2.4 Pruning Theory & Mathmatical Induction

(TODO:)

2.2.5 Comparison to Our Work

(TODO:)

3 Our Contributions

In this paper, we study LLM & LVM model pruning that attempt to achieve a good trade-off between network size and generation quality. Our main contributions are as follows:

1. We describe an approach called Movement that can perform much better than previous approaches;

2. We describe several algorithms closely related to pruning and design a unified benchmark for model evaluation;

3. We perform a comprehensive experimental evaluation via the combinations of different datasets, models and evaluation metrics;

4. We compare human designed algorithms with AIGC generated algorithms, demonstrating the pros and cons on both sides in specific domains such as ”code generation”.

4 Our Proposed Pruning Algorithms

4.1 Algorithm Design Principles

When we are designing the algorithms, we take considerations into the following design principles:

1. Target Sparsity Level: What percentage of weights do we aim to prune? Higher sparsity can lead to greater compression and speedups but might sacrifice more accuracy.

2. Quality-Size Trade-off: Finding the right balance between model size & speed & quality is crucial. Some algorithms prioritize accuracy (SparseGPT), while others are more aggressive in pursuing sparsity (magnitude).

3. Pruning Criterion: How do you determine which connections to prune? Options may include: Weights (Magnitude), Activation statistics (WANDA), Gradient (SparseGPT) et al.

4. Structured vs. Unstructured Pruning: The formal method attempts to prune individual weights anywhere, potentially leading to irregular sparsity patterns that might not be hardware-friendly. While the latter method attempts to prune in blocks (e.g., 2:4, 4:8), which can be more efficient for some underline hardware and libs.

5. Pruning Schedule: When and how do we prune? One-shot pruning attempts to prune once at the beginning or after training. While the others attempt to incrementally prune over multiple training epochs.

6. Usage of Calibration Data: Some algorithms like WANDA require a small calibration dataset to collect activation statistics before pruning. The choice of this data can impact pruning effectiveness.

7. Hardware Awareness: Consider the target hardware (CPUs, GPUs, specialized accelerators) and design pruning strategies that align with hardware constraints for optimal efficiency.

8. Layer-Wise Sparsity: Allow different layers to have varying sparsity levels based on their sensitivity. It is well-known that NOT all layers contribute equally to a model's performance.

9. Regularization and Stability: Pruning can always lead to instability during training & prediction. An end-to-end model evaluation is needed in order for final model deployment in production system.

4.2 Human Proposed Algorithms

1. Movement Pruning. The Core Idea of this alg is: Instead of directly removing weights, movement pruning identifies unimportant weights and ”moves” their values to other more significant connections. This helps preserve the overall information flow within the network. (TODO: If the result is good, we will fill in more detail)

5 Experimental Results

From Table 1 presents perplexity results for pruned Llama-7B models using different pruning methods: **Wanda**, **SparseGPT**, **Magnitude**, and **Movement**, across various pruning levels. Below are key observations and conclusions derived from the table:

1. General Trend with Pruning

As the pruning level increases, i.e., a higher fraction of the model's parameters are removed, the perplexity values generally increase for all methods. This trend is expected as a greater loss of parameters typically leads to a degradation in model performance.

2. Low Pruning Levels (0.01 - 0.10)

At lower pruning levels:

- At 0.01 and 0.05 pruning levels, several methods (**Wanda**, **SparseGPT**, **Magnitude**) show “NA” values, while **Movement** retains a low perplexity ($\sim$5.7).
- At 0.10 pruning, all methods show similar perplexity values, ranging from 5.6 to 5.8, indicating minimal performance degradation.

3. Medium Pruning Levels (0.20 - 0.50)

At medium pruning levels:

- Up to the 0.50 pruning level, **SparseGPT** and **Wanda** consistently show lower perplexity values compared to **Magnitude** and **Movement**.
- By 0.50 pruning, there is a noticeable gap, where **SparseGPT** and **Wanda** have perplexities around 7.2, whereas **Magnitude** and **Movement** have perplexities around 17, showing poorer performance.

4. High Pruning Levels (0.60 - 0.90)

At higher pruning levels:

- From 0.60 onward, the differences between the methods become more pronounced. **SparseGPT** and **Wanda** maintain significantly lower perplexity scores compared to **Magnitude** and **Movement**, which exhibit a rapid increase in perplexity.
- At 0.70 pruning, for instance, **SparseGPT** has a perplexity of 27.214, while **Magnitude** and **Movement** reach over 48,000 and 51,000 respectively.

5. Extreme Pruning Levels (0.95 and 0.99)

At extreme pruning levels:

- All methods show significantly higher perplexity values, yet **SparseGPT** and **Wanda** continue to outperform **Magnitude** and **Movement** by a large margin.
- At 0.99 pruning, **SparseGPT** has a perplexity of $\sim$16,869, whereas **Magnitude** and **Movement** exhibit perplexities of $\sim$222,543 and $\sim$214,966 respectively.

Conclusions

- **SparseGPT** and **Wanda** consistently outperform **Magnitude** and **Movement** pruning methods, especially at medium to high pruning levels (0.60 and above).
- **Magnitude** and **Movement** pruning methods exhibit significant degradation in performance (higher perplexity) at more aggressive pruning levels.
- **SparseGPT** is the most resilient pruning method across varying pruning levels, maintaining the lowest perplexity even at extreme pruning levels (0.90 and 0.95).

Thus, **SparseGPT** (and **Wanda** to some extent) seem to be the preferred methods when applying aggressive pruning to large models, as they better preserve performance as indicated by perplexity.

From table 2, we can see that

From table 3, we can see that

From table 4, we can see that the o1 model finds it difficult to generate effective algorithms in one go in the creative application scenario of ”core algorithm generation,” despite our clear understanding of the context and the knowledge domain involved during the experiment. Preliminary experiments show that the o1 model, released on September 12, 2024, did not demonstrate the exceptional capabilities of ”slow thinking,” ”outstanding mathematical logic reasoning,” and ”programming ability” that were emphasized during its promotion and dissemination, at least in our innovation application scenario, as these traits were not significantly quantifiable by scientific metrics. Our future work can focus on the following aspects: (1) Investigating the reasons why the algorithm cannot be generated and successfully run in one go. (2) A horizontal comparison of the effectiveness of AIGC-generated algorithms versus those designed by human algorithm engineers. (3) Expanding the evaluation from ”code generation” by generative AI to more comprehensive assessments such as ”text generation,” ”image generation,” and ”video generation.” (4) Adding a horizontal comparison of models such as GPT-4 and Gemini Pro in vertical domains.

6 Conclusions

In this paper, we have introduced several novel algorithms for model pruning in large language models and large vision models. Through comparison with query wheel and query covering approaches, our methodology, which attempts to estimate the likelihood of neurons resulting in expected results based on diverse neuron features, collections, and query statistics, has demonstrated significant improvement over prior work as evidenced by our experimental results. For future work, we plan several extensions. This includes conducting experiments with other language models that may potentially achieve even better pruning performances. We also aim to optimize our approach further, such as exploring hybrid methods.

Additionally, we plan to study the tradeoff between model size and query cost under different cost models and for actual query processing algorithms. This research holds promise for enhancing the

Table 1: Perplexity on pruned model (Llama-7B) from human domain experts

Pruned Level	Wanda	SparseGPT	Magnitude	Movement
0.01	NA	NA	NA	5.677
0.05	NA	NA	NA	5.714
0.10	5.696	5.696	5.806	5.806
0.20	5.817	5.799	6.020	6.020
0.30	5.999	5.963	6.669	6.668
0.40	6.387	6.311	8.601	8.594
0.50	7.257	7.234	17.285	17.247
0.60	10.691	10.442	559.987	554.727
0.70	84.905	27.214	48414.551	51841.121
0.80	5782.432	182.463	132175.578	135494.797
0.90	19676.668	3198.101	317879.250	301472.500
0.95	28309.178	4088.413	273552.281	273629.750
0.99	108234.484	16869.203	222543.047	214966.484

Table 2: Effectiveness of the weights as a major pruning measure

Pruned Level	Prune by Weights	Prune by -Weights
0.01	NA	24377.635
0.05	NA	25804.920
0.10	5.806	104948.891
0.20	6.020	352772.500
0.30	6.669	335747.406
0.40	8.601	260632.641
0.50	17.285	227413.484
0.60	559.987	185086.078
0.70	48414.551	273153.688
0.80	132175.578	188488.000
0.90	317879.250	185304.016
0.95	273552.281	NA
0.99	222543.047	NA

efficiency and performance of large language and vision models through more effective pruning techniques.

7 CITATIONS, FIGURES, REFERENCES

These instructions apply to everyone, regardless of the formatter being used.

7.1 CITATIONS WITHIN THE TEXT

Citations within the text should be based on the `natbib` package and include the authors' last names and year (with the "et al." construct for more than two authors). When the authors or the publication are included in the sentence, the citation should not be in parenthesis using `\citet{}` (as in "See **?** for more information."). Otherwise, the citation should be in parenthesis using `\citep{}` (as in "Deep learning shows promise to make progress towards AI (**?**).").

The corresponding references are to be listed in alphabetical order of authors, in the REFERENCES section. As to the format of the references themselves, any style is acceptable as long as it is used consistently.

Table 3: Effectiveness of the bias as a major pruning indicator

Pruned Level	Prune by Bias	Prune by -Bias
0.01	NA	NA
0.05	NA	NA
0.10	NA	NA
0.20	NA	NA
0.30	NA	NA
0.40	NA	NA
0.50	NA	NA
0.60	NA	NA
0.70	NA	NA
0.80	NA	NA
0.90	NA	NA
0.95	NA	NA
0.99	NA	NA

Table 4: One pass code generation and effectiveness evaluation

Number	Core Idea	Status	Usage Scenario
01	Gradient Sensitive Pruning	Error	Code Generation
02	L1 Norm Pruning	OK	Code Generation
03	Structured Pruning	OK	Code Generation
04	K-means Clustering Pruning	Error	Code Generation
05	Random Pruning	OK	Code Generation
06	Random Pattern Pruning	OK	Code Generation
07	Variational Dropout Pruning	Error	Code Generation
08	Gradient based Pruning	Error	Code Generation
09	Elastic Weight Consolidation Pruning	Error	Code Generation
10	Dynamic Pruning with Reinforcement Learning	Error	Code Generation

7.2 FOOTNOTES

Indicate footnotes with a number[1] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches (12 picas).[2]

7.3 FIGURES

AUTHOR CONTRIBUTIONS

If you'd like to, you may include a section for author contributions as is done in many journals. This is optional and at the discretion of the authors.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

A APPENDIX

You may include other additional sections here.

[1] Sample of the first footnote
[2] Sample of the second footnote

Table 5: Perplexity on pruned model (llama-7B) from AIGC domain expert (o1)

Pruned Level	**aigc algorithm 2**	**aigc algorithm 3**	**aigc algorithm 6**
0.50	193740.406	266826.094	294350.188
0.60	110879.422	244139.875	138577.469
0.70	174815.859	453267.031	171725.375
0.80	287734.844	570346.750	186493.797
0.90	157028.844	384411.375	298142.469
0.95	90220.781	455298.469	187259.063
0.99	991519.125	206585.391	70452.703

Table 6: Effect of pruned model (OPT-1.3B) applying to downstream task - text generation

Pruned Level	**Perplexity**	**University is**
0.00	*	University is a great place to learn about the world.
0.50	19.191	University is a great place to start a new year.
0.60	23.205	University is a great place to start.
0.70	44.246	University is a good place to get a good place to get a good place to get a good
0.80	364.304	University is a lot lot lot lot lot lot lot lot lot lot lot lot lot lot lot lot
0.90	3772.829	University is.
0.95	8892.167	University is is is is is is is is is is is is is is is is is is
0.99	22548.809	University is is is is is is is is is is is is,,,,,,

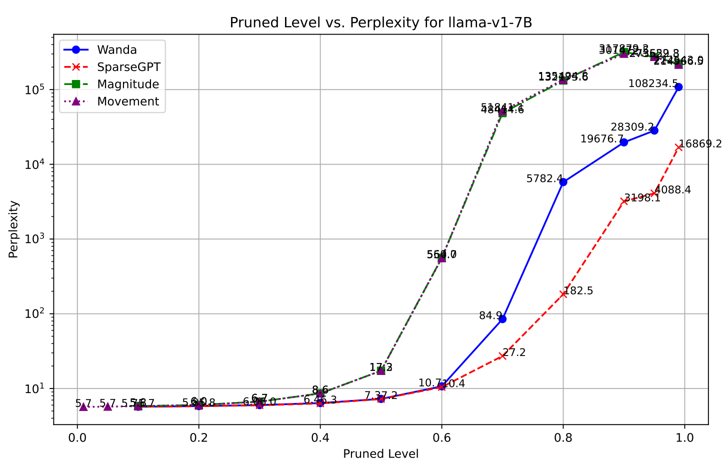

Figure 1: Pruned Level vs. Perplexity for llama-v1-7B

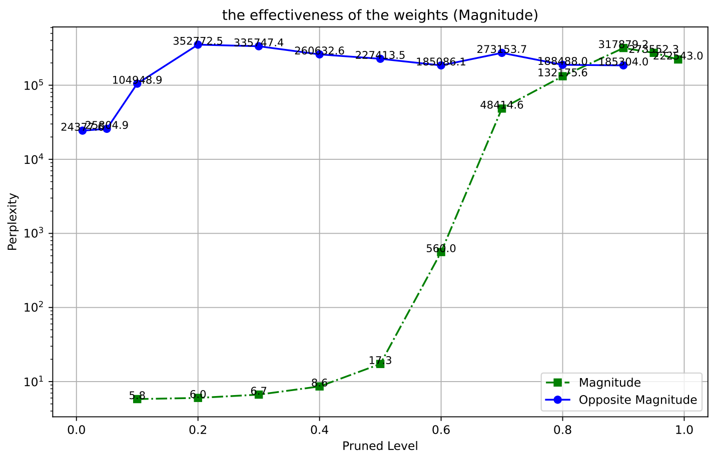

Figure 2: Effectiveness of the weights indicator

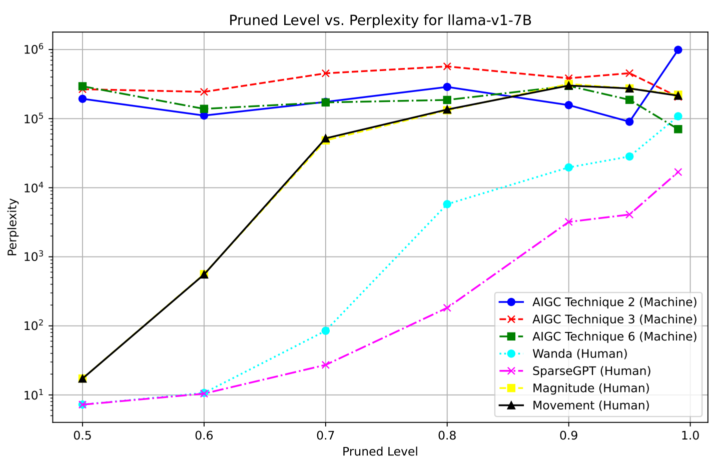

Figure 3: Performance Evaluation between Machine & Human