D^2Prune : Sparsifying Large Language Models via Dual Taylor Expansion and Attention Distribution Awareness

Lang Xiong[1*], Ning Liu[3*], Ao Ren[1†], Yuheng Bai[1], Haining Fang[1], Binyan Zhang[1], Zhe Jiang[1], Yujuan Tan[2†], Duo Liu[1]

[1]Chongqing University, China
[2]National University of Defense Technology, China
[3]Beijing Innovation Center of Humanoid Robotics, China

langxiong@stu.cqu.edu.cn, ningliu1220@gmail.com, {ren.ao, yuheng, haining.fang, zby, jiangzhe}@cqu.edu.cn, tanyujuan@nudt.edu.cn, liuduo@cqu.edu.cn

Abstract

Large language models (LLMs) face significant deployment challenges due to their massive computational demands. While pruning offers a promising compression solution, existing methods suffer from two critical limitations: (1) They neglect activation distribution shifts between calibration data and test data, resulting in inaccurate error estimations; (2) Overlooking the long-tail distribution characteristics of activations in the attention module. To address these limitations, this paper proposes a novel pruning method, D^2Prune. First, we propose a dual Taylor expansion-based method that jointly models weight and activation perturbations for precise error estimation, leading to precise pruning mask selection and weight updating and facilitating error minimization during pruning. Second, we propose an attention-aware dynamic update strategy that preserves the long-tail attention pattern by jointly minimizing the KL divergence of attention distributions and the reconstruction error. Extensive experiments show that D^2Prune consistently outperforms SOTA meth ods across various LLMs (e.g., OPT-125M, LLaMA2/3, Qwen3). Moreover, the dynamic attention update mechanism also generalizes well to ViT-based vision models like DeiT, achieving superior accuracy on ImageNet-1K.

Code & Full Version — https://github.com/cquxl/D2Prune/

Introduction

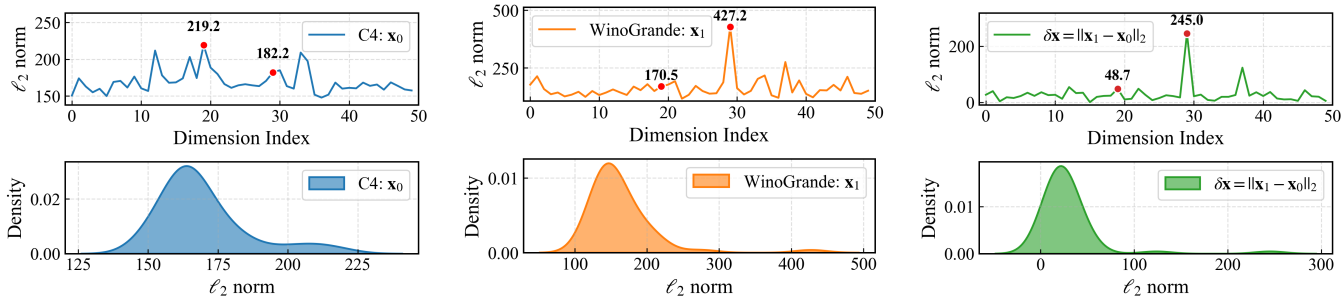
In recent years, Large Language Models (LLMs) (Guo et al. 2025; Dubey et al. 2024; OpenAI 2023) have revolutionized the field of Natural Language Processing (NLP), excelling in complex tasks such as causal reasoning and text generation (Chi et al. 2024; Wang 2024; Mo et al. 2024; Li et al. 2024a). However, their exceptional performance comes at the cost of substantial memory and computational demands, posing significant challenges for deployment on resource-constrained devices (Li et al. 2024b; Kim et al. 2024). Extensive efforts have been made in model pruning techniques, aiming at shrinking the network size by removing redundant weights (Bai et al. 2024; Dong et al. 2024; Sun et al. 2023; Frantar and Alistarh 2023). By introducing sparsity, pruning enhances both memory and computational efficiency and has proven effective in LLMs. Nonetheless, traditional training-based model pruning generally requires substantial computational resources due to processes such as retraining (Liu et al. 2018; Blalock et al. 2020), training from scratch (Hoang and Liu 2023; Sreenivasan et al. 2022), or extensive iterative pruning (Chijiwa et al. 2021; Tanaka et al. 2020). These training-based methods are costly, especially for current large language models (LLMs).

Recently, post-training pruning methods (Kwon et al. 2022; Zhang et al. 2024) have emerged as a more efficient schema for compressing LLMs as they require no additional training resources. These methods typically design weight importance metrics to remove redundant weights with minimal impact on layer-wise output errors, categorized into non-weight-update pruning (e.g., Magnitude (Han et al. 2015), Wanda (Sun et al. 2023), Pruner-Zero (Dong et al. 2024)) and weight-update pruning (e.g., LLM Surgeon (van der Ouderaa et al. 2023), SparseGPT (Frantar and Alistarh 2023), SparseLLM (Bai et al. 2024), ADMM-Grad (Boža 2024)). Through in-depth analysis, we identify two fundamental limitations in current methodologies stemming from inadequate consideration of activation impacts on error propagation.

First, the constant activation assumption in error estimation. Most mainstream pruning methods inherit the "constant activation assumption" from early methods like OBS (Hassibi, Stork, and Wolff 1993), which treat input activations as fixed during pruning. This assumption is valid for small-scale networks, where the train/test data is independent and identically distributed. It fails to hold in modern large language models (LLMs). Our experiments reveal significant shifts in input activation across datasets and layers. For example, Figure 2 illustrates a segment of activation features from the 10th layer of LLaMA-2-7B. The activation norms exhibit significant shifts, indicating a large deviation between the activations across datasets and layers.

Non-weight-update methods, such as Wanda and Pruner-Zero, evaluate importance via the product of weight (or gradient) magnitudes and activation norms, which essentially re-

*These authors contributed equally.
†Corresponding authors



(a) Upstream activation distribution (b) Downstream activation distribution (c) Activation shift distribution

Figure 1: Activation distributions and their shift between upstream and downstream data in LLaMA-2-7B (layer 10). During pruning, a small subset of C4 is used as calibration data (upstream), while downstream tasks such as WinoGrande are evaluated in a zero-shot setting. All inputs are formatted as 128-sample sequences with maximum embedding length. The L_2 norms represent the average activation magnitude across samples.

flects output amplitude rather than true pruning error. Weight-update methods, such as SparseGPT and OBS, explicitly approximate the pruning-induced error via the single-variable-based Taylor expansion on the weight but omit the effect of activation variation between calibration and downstream data, causing inaccurate error modeling. Thus, the fixed activation assumption is fundamentally flawed for LLM pruning. A more accurate modeling of activation shift is required for precise error estimation and robust downstream performance.

Second, negligence of long-tail distribution in attention modules. Multi-head attention, as the core component of Transformers, shows long-tail attention distribution (as noted in several literatures (Zhou et al. 2021; Ji et al. 2021; Chen et al. 2022)) where a few key tokens dominate attention scores, which is pivotal for LLM reasoning. Nonetheless, current pruning methods treat query (Q), key (K), and value (V) weights in attention modules identically to linear layers (Gate, Up, and Down in MLP modules), incurring significant error propagation. Specifically, **the non-weight-update pruning accumulates errors in Q/K/V layers by retaining raw weights, while weight-update pruning disrupts the long-tail attention distribution through global weight adjustments.** As illustrated in Figure 2, direct updates to Q/K/V weights homogenize attention scores from the original long-tail distribution, severely distorting the model's focus on critical tokens. Conversely, skipping updates leads to significant errors in attention outputs due to error accumulation. This "all-or-nothing" approach fails to balance error compensation and distribution preservation, causing catastrophic performance drops, especially at high sparsity.

To tackle the above limitations, we propose D^2Prune, a novel pruning method for large language models via Dual Taylor Expansion and Attention Distribution Awareness. Specifically, to address the first limitation, we propose **Activation-Weight Dual-Sensitive Pruning Mechanism**. By formulating a Dual Taylor Expansion of the error function with respect to both activations and weights, our method jointly models the impacts of activation variations and weight perturbations on output errors, enabling precise pruning mask selection and weight updates. This dual expansion reduces perplexity by approximately 10% compared to conventional single-variable approaches , and it enhances accuracy at high sparsity by up to 40% on downstream tasks with significant distribution shifts . To address the second limitation, we propose **Attention Distribution-Aware Dynamic Weight Update Strategy**. Specifically, we formulate Q/K/V update states as a combinatorial optimization problem and propose a perplexity-guided lightweight adaptive search method that dynamically identifies update configurations. This strategy can effectively preserve the attention distribution while reducing the pruning errors by allowing weight updating. Experimental results demonstrate that **this strategy reduces the KL divergence of the attention distribution and the root-mean-square error (RMSE) of attention outputs by 61% and 43% on average** compared with non-weight-update (Wanda) and weight-update (SparseGPT) baselines. Our contributions are summarized as follows:

- We propose a novel pruning framework (D^2Prune) for compressing LLMs, which simultaneously accounts for activation dynamics and weight sensitivity via Dual Taylor Expansion, enabling effective pruning mask selection and weight updating.
- We propose an attention distribution-aware dynamic weight update method, which dynamically identifies update configurations for q, k, v weights in attention modules, achieving balance in attention distribution preservation and pruning errors compensation.
- Experimental results demonstrate that D^2Prune achieves SOTA performance across varying parameter scales of LLMs (including OPT-125M, LLaMA2/3 models, Qwen3-8/14B.), with substantial improvements in perplexity and zero-shot accuracy across various sparsity levels, especially at high sparsity.

Preliminaries

Related Work

Post-training pruning has emerged as a critical technique for compressing Large Language Models (LLMs) (Kwon et al.

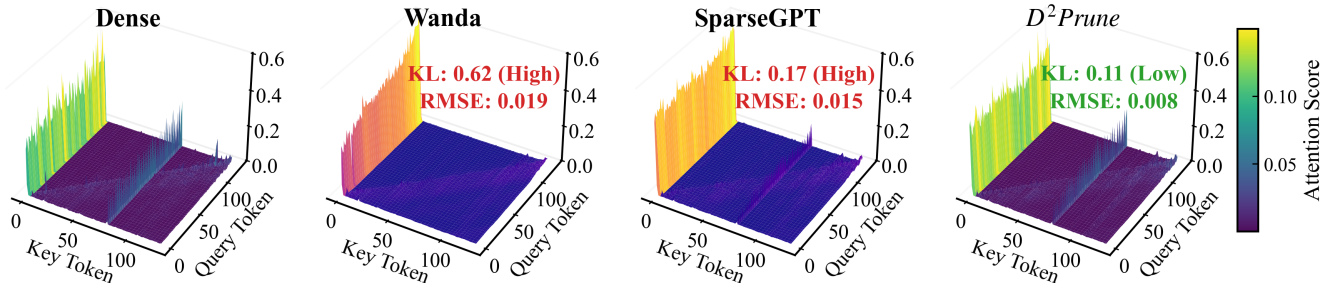


Figure 2: **Visualizing of uniformized multi-head attention in LLaMA-2-7B (80% Sparsity, 128-token sequences from C4 as Calibraion input).** We compare the 3D attention scores of the surfaces in the **final Transformer layer** for dense models against pruned counterparts (Wanda, SparseGPT, and our D^2Prune). D^2Prune selectively updates projections to optimize for distribution consistency, successfully preserving these patterns with minimal distortion (lowest KL/RMSE).

2022; Zhang et al. 2024; Reyhan, Rahnamayan, and Bidgoli 2024). In Appendix G, we review post-training pruning techniques in detail, categorizing them into the non-weight-update and weight-update pruning methods.

Problem Statement

Post-training pruning is performed by decomposing the full-model compression problem into layer-wise pruning subproblems, which can be written as:

$$\arg\min_{\boldsymbol{M}_l} ||\boldsymbol{W}_l \boldsymbol{X}_l - (\boldsymbol{M}_l \odot \boldsymbol{W}_l)\, \boldsymbol{X}_l||_2^2, \tag{1}$$

where the subscript l represents the l_{th} layer, X_l is the activation, W_l is the weight matrix, and $\boldsymbol{M}_l$ is the pruning mask corresponding to the weights, respectively. Furthermore, weight-update methods aim to adjust the remaining unpruned weights to compensate for the pruning-induced error, resulting in an updated weight matrix $\hat{\boldsymbol{W}}_l$. The optimization objective is as follows:

$$\arg\min_{\boldsymbol{M}_l, \hat{\boldsymbol{W}}_l} ||\boldsymbol{W}_l \boldsymbol{X}_l - \left(\boldsymbol{M}_l \odot \hat{\boldsymbol{W}}_l\right) \boldsymbol{X}_l||_2^2. \tag{2}$$

Nonetheless, simultaneously solving for both the sparsity mask and the adjustments to the remaining weights is an NP-hard problem. Mainstream approaches (Frantar and Alistarh 2023; Bai et al. 2024; Hassibi, Stork, and Wolff 1993) typically use a pruning metric to select the pruning mask and then optimize the remaining weights based on that mask. Prior studies mainly adopt the layer-wise pruning error minimization. They usually solve Eq. 1 and Eq. 2 through the single-variable Taylor expansion with respect to weights. OBS and SparseGPT calculate the inverse of the second-order derivative matrix (Hessian) to perform weight mask selection and update. Specifically, the output error change of a layer can be written in the single-variable Taylor expansion form in OBS and SparseGPT:

$$\delta E = \left(\frac{\partial E}{\partial \mathbf{w}}\right)^T \delta \mathbf{w} + \frac{1}{2} \delta \mathbf{w}^T \mathbf{H} \delta \mathbf{w}, \tag{3}$$

where $\mathbf{H} = \partial^2 E / \partial \mathbf{w}^2$, $\delta \mathbf{w} = \mathbf{w} - \mathbf{w}_0$, $\delta E = E(\mathbf{w}) - E(\mathbf{w}_0)$ represents the change of the objective function (error or loss change). Following the principle proposed in Optimal Brain Damage (LeCun, Denker, and Solla 1989), we define the saliency of a parameter as the increase in the objective function (i.e., loss) after removing that parameter. Therefore, the goal of pruning is to minimize the change in error given by Eq. 3, when setting one of the weights $\mathbf{w}$ to zero (represented as w_q). Elimating w_q is expressed as $\delta w_q + w_q = 0$ or expressed in vector form more generally, that is $\mathbf{e}_q^T \mathbf{w} + w_q = 0$, where $\mathbf{e}_q$ is the unit vector in weight space corresponding to scalar w_q. Then, the goal is to solve:

$$\delta \mathbf{w}^* = \arg\min_{\delta \mathbf{w}} \tfrac{1}{2} \delta \mathbf{w}^T \mathbf{H} \delta \mathbf{w} \quad \text{s.t.} \quad \mathbf{e}_q^T \delta \mathbf{w} + w_q = 0. \tag{4}$$

To solve the Eq. 4, we can form a Lagrangian function:

$$L_q = \frac{1}{2} \delta \mathbf{w}^T \cdot \mathbf{H} \cdot \delta \mathbf{w} + \lambda \left(\mathbf{e}_q^T \delta \mathbf{w} + w_q\right), \tag{5}$$

where λ is a Lagrange undetermined multiplier. Finally, we can obtain the optimal weight change $\delta \mathbf{w}$ and the change in error δE:

$$\delta \mathbf{w} = -\frac{w_q}{\left(\mathbf{H}^{-1}\right)_{qq}} \left(\mathbf{H}^{-1}\right)_{:,q}, \delta E = L_q = \frac{w_q^2}{\left(\mathbf{H}^{-1}\right)_{qq}}. \tag{6}$$

Since L_q is the increase in error and is used to reflect the "saliency" of w_q in prior studies, both the weight-update and non-weight-update pruning methods evaluate the importance of w_q utilizing the magnitude L_q or its variants. They rank weights by L_q and then prune the lowest-saliency weights to meet the target sparsity.

Methodology

We propose D^2Prune, a novel pruning framework for LLMs. An overview of D^2Prune is shown in Figure 3 and Algorithm 1 (available in Appendix A).

Dual Taylor Expansion for Mask Selection & Weight Update

Consider a pre-trained large language model where the original input activation of a certain layer is $\boldsymbol{x}_0$, the weight is $\boldsymbol{w}_0$, and the output is $y = \boldsymbol{x}_0{}^T \boldsymbol{w}_0$. Now, suppose we remove a

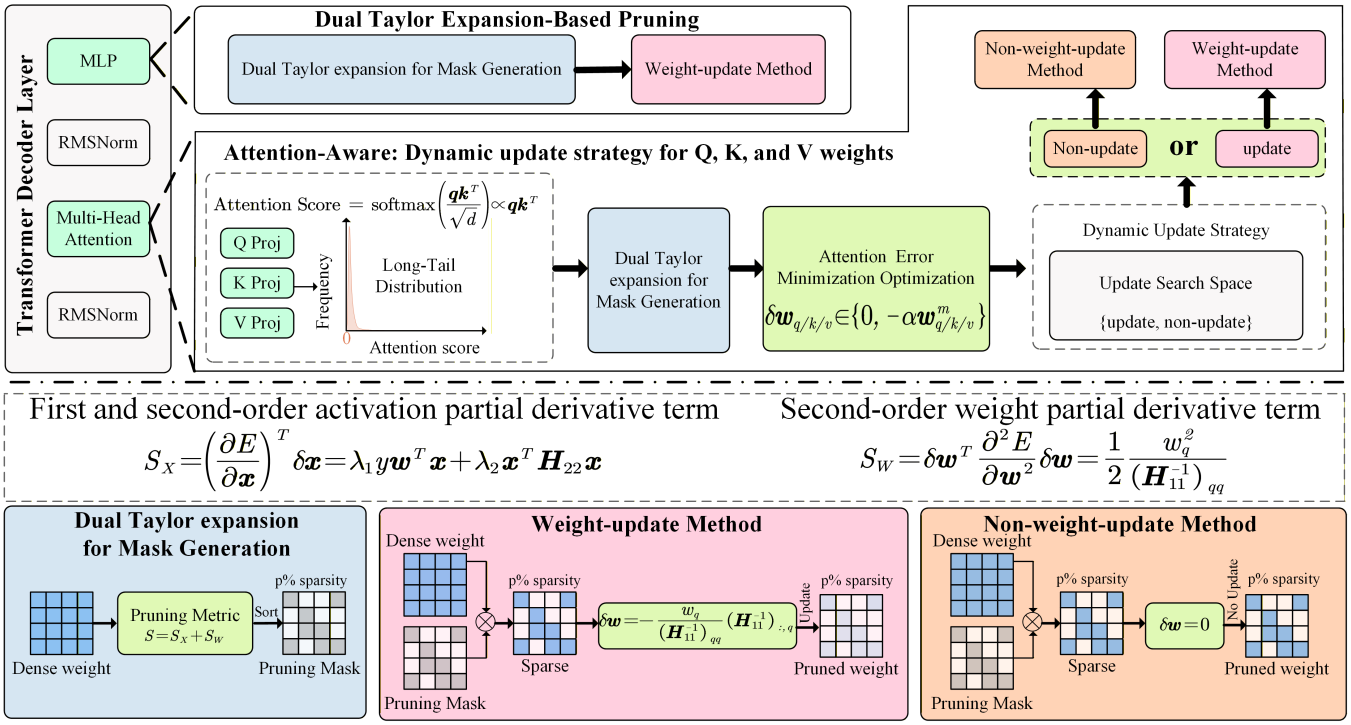


Figure 3: Illustration of the D^2Prune framework. Using the update status of q, k, and v weights in each attention layer as the search space, we take the minimization of ppl as the objective to dynamically adapt the update strategy of the q, k, and v weights.

certain weight such that the error is $E = ||\boldsymbol{x}^T\boldsymbol{w} - \boldsymbol{x_0}^T\boldsymbol{w}_0||_2^2$, where $\boldsymbol{w} = \boldsymbol{w}_0 + \delta\boldsymbol{w}$ denotes the weights retained after pruning, and $\boldsymbol{x}$ is the input activation. **Previous works typically sample a small calibration set and compute layerwise input activations during pruning, assuming the activations remain constant, i.e., $\boldsymbol{x} = \boldsymbol{x}_0$.** However, as demonstrated in Figure 1c, there exists a significant distributional shift between upstream and downstream activations. A more realistic formulation is $\boldsymbol{x} = \boldsymbol{x}_0 + \delta\boldsymbol{x}$, where $\delta\boldsymbol{x}$ is activation shift. Neglecting activation shift leads to inaccurate error estimation in prior pruning methods.

To address this issue, we now consider the error minimization problem as a dual function of both weights and activations. The first-order partial derivative term with respect to the weights is 0 as a network trained to a local minimum in error, and the third and all higher-order terms converge to 0 and can be eliminated. Therefore, following OBS and SparseGPT (refer to Eq. 3), the functional Taylor series of the error change with respect to weights and activations is:

$$\delta E = \left(\frac{\partial E}{\partial \boldsymbol{u}}\right)^T \delta\boldsymbol{u} + \left(\frac{1}{2}\delta\boldsymbol{u}^T\boldsymbol{H}\delta\boldsymbol{u}\right), \tag{7}$$

where $\boldsymbol{u} = (\boldsymbol{w}, \boldsymbol{x})^T, \frac{\partial E}{\partial \boldsymbol{u}} = \left(\frac{\partial E}{\partial \boldsymbol{w}}, \frac{\partial E}{\partial \boldsymbol{x}}\right)^T, \delta\boldsymbol{u} = (\delta\boldsymbol{w}, \delta\boldsymbol{x})^T$, $\boldsymbol{H} = \begin{pmatrix} \partial^2E/\partial\boldsymbol{w}^2 & \partial^2E/\partial\boldsymbol{w}\,\partial\boldsymbol{x} \\ \partial^2E/\partial\boldsymbol{x}\,\partial\boldsymbol{w} & \partial^2E/\partial\boldsymbol{x}^2 \end{pmatrix}$. It is worth noting that, due to the presence of Layer Normalization in Transformers, the input activations of each layer are normalized during the forward propagation on the calibration data. Moreover, since the pruning-induced $\delta\boldsymbol{w}$ is small, the cross terms in the Hessian matrix become higher-order infinitesimals and have a negligible impact on error estimation (a theory derivation and experimental verification detailed in Appendix C). Therefore, it is reasonable to assume that the Hessian is predominantly dominated by its diagonal elements, such that:

$$\delta E = \left(\frac{\partial E}{\partial \boldsymbol{x}}\right)^T \delta\boldsymbol{x} + \frac{1}{2}\delta\boldsymbol{w}^T\boldsymbol{H}_{11}\delta\boldsymbol{w} + \frac{1}{2}\delta\boldsymbol{x}^T\boldsymbol{H}_{22}\delta\boldsymbol{x}, \tag{8}$$

where $\frac{\partial E}{\partial \boldsymbol{x}} = -\boldsymbol{w}\left(\boldsymbol{x}^T\boldsymbol{w} - y\right)$, $\boldsymbol{H}_{11} = \frac{\partial^2 E}{\partial \boldsymbol{w}^2} = \boldsymbol{x}\boldsymbol{x}^T$, $\boldsymbol{H_{22}} = \frac{\partial^2 E}{\partial \boldsymbol{x}^2} = \boldsymbol{w}\boldsymbol{w}^T$, $\delta E = E(\boldsymbol{u}) - E(\boldsymbol{u}_0)$ represents the change of objective function (loss or error). Following the principle proposed in Optimal Brain Damage (Hassibi, Stork, and Wolff 1993), it is reasonable to define the saliency of a parameter as the increase in the objective function (i.e., loss) after removing that parameter, since objective functions play a central role in neural network pruning. Therefore, δE captures the change in the objective function (i.e., network error) caused by perturbations to the weights and activations.

Since the activations of the batch training data are not available for each layer, computing $\delta\boldsymbol{x}$ directly is difficult. Fortunately, as shown in Figure 7-9 (detailed in Appendix C, average cosine similarity across layers reaches 0.96, range: 0.94–0.98), we observe a linear correlation between activation norm shifts across different tasks and the calibration activations across model layers (while the relationship may appear nonlinear at the level of individual weight dimensions (see Figure 1), the layerwise nature of our error estimation makes the linear approximation reasonably valid. As evidenced by the results in Table 12 (detailed in Appendix B.4),

this assumption leads to improved robustness in practice. We leave more fine-grained modeling of $\delta \boldsymbol{x}$ to future work.

Thus, we assume that there exists a perturbation coefficient λ such that $\boldsymbol{\delta x} = \lambda \boldsymbol{x}$ (a theoretical analysis with experimental support results detailed in Appendix C. Substituting this into Eq. 8 and eliminating the constant coefficient term, we can obtain:

$$\begin{aligned}\delta E &= \lambda y \boldsymbol{w}^T \boldsymbol{x} + \left(\frac{1}{2}\lambda^2 - \lambda\right) \boldsymbol{x}^T \boldsymbol{w} \boldsymbol{w}^T \boldsymbol{x} + \frac{1}{2} \delta \boldsymbol{w}^T \boldsymbol{H}_{11} \delta \boldsymbol{w} \\ &= \lambda_1 y \boldsymbol{w}^T \boldsymbol{x} + \lambda_2 \boldsymbol{x}^T \boldsymbol{H}_{22} \boldsymbol{x} + \frac{1}{2} \delta \boldsymbol{w}^T \boldsymbol{H}_{11} \delta \boldsymbol{w}, \end{aligned} \tag{9}$$

where $\lambda_1 = \lambda$, $\lambda_2 = \frac{1}{2}\lambda^2 - \lambda$ denote the perturbation coefficients of the first-order activation bias and the second-order activation bias term, respectively. the goal of pruning w_q should satisfy $w_q + \delta w_q = 0$, i.e., $\boldsymbol{e}_q^T \delta \boldsymbol{w} + \boldsymbol{w} = 0$, the solution of $\delta \boldsymbol{w}$ and δE can be obtained using the Lagrange multiplier method as follows:

$$\begin{aligned}\delta \boldsymbol{w} &= -\frac{w_q}{\left(\boldsymbol{H}_{11}^{-1}\right)_{qq}} \left(\boldsymbol{H}_{11}^{-1}\right)_{:,q}, \\ L_q &= \lambda_1 y \boldsymbol{w}^T \boldsymbol{x} + \lambda_2 \boldsymbol{x}^T \boldsymbol{H}_{22} \boldsymbol{x} + \frac{1}{2} \frac{w_q^2}{\left(\boldsymbol{H}_{11}^{-1}\right)_{qq}}. \end{aligned} \tag{10}$$

Finally, we use L_q as the pruning metric S to select the pruning mask and update the weights with $\delta \boldsymbol{w}$. It is worth noting that, unlike pruning methods such as SparseGPT, Wanda, and Pruner-Zero, our pruning metric S explicitly incorporates the output activation (i.e., y, the input activation of the next layer), demonstrating its ability to model cross-layer dependency.

Attention Distribution-Aware Dynamic Weight Update Strategy

For the q/k/v weights in the attention module, under the pruning mask obtained by dual error minimization and after pruning a certain weight w_m, the weights are updated as $w_i = w_0^i - \delta w_i \, (i \neq m)$, where w_0^i denotes the original i-th weight, $\delta w_i = \alpha_i \, w_q (\alpha_i = (H^{-1})_{qi} / (H^{-1})_{qq})$. The original attention of the i-th token is:

$$\text{Attention}\,(Q, K, V)_i = \sum_{j=1}^{n} \text{soft}\max\left(\frac{\boldsymbol{q}_i \boldsymbol{k}_j^T}{\sqrt{d}}\right) \boldsymbol{v}_j, \tag{11}$$

where $(Q, K, V)_i$ corresponds to the query, key, and value of the i-th token. n denotes the number of tokens and d is the feature dimension. The softmax operator computation result is proportional to the product of $\boldsymbol{q}_i$ and the transposed $\boldsymbol{k}_j$ and the attention is proportional to the elements of $\boldsymbol{v}_j$. The long-tail distribution of multi-head attention is critical for semantic understanding in LLMs, where a small fraction of positions carry disproportionately large, critical weights. Unconstrained linear updates to the remaining q/k/v weights shift these relative magnitudes, distorting the original long-tail pattern and undermining attention fidelity. As illustrated in Figure 2, the non-weight-update method can better maintain the long-tail distribution while achieving worse model performance.

However, by entirely refraining from updating pruned q/k/v weights, local errors propagate and accumulate across layers. To address this issue, we propose an attention distribution-aware dynamic weight update strategy that explicitly preserves key attention patterns via a distribution-constrained optimization objective. After pruning off the m-th weight for q, k,v weights of multi-head attention (MHA) and under the pruning mask selection based on the dual error minimization method, the restoration of q/k/v weights is equivalent to solving the following optimization problems:

$$\begin{aligned}&\min_{\delta w_{q/k/v}} \mathcal{L}_{q/k/v} + \rho \cdot \text{KL}\left(\text{MHA}(\tilde{w}_{q/k/v}) \,||\, \text{MHA}(w_{q/k/v})\right), \\ &\text{s.t.} \quad \delta w_{q/k/v} \in \{0, -\alpha_{q/k/v} w_{q/k/v}^m\}, \\ &\qquad \mathcal{L}_{q/k/v} = \sum_{*\in\{q,k,v\}} \|w_* x_* - (M_* \odot \tilde{w}_*) x_*\|_2^2, \\ &\qquad \tilde{w}_{q/k/v} = w_{q/k/v} + \delta w_{q/k/v}, \end{aligned} \tag{12}$$

where $L_{q/k/v}$ measures the output error from pruning for q, k, and v layer. KL is the Kullback–Leibler divergence enforcing the pruned attention distribution to mimic the original long-tail pattern. ρ is the Lagrangian multiplier, which balances error minimization and distribution preservation. $\delta \boldsymbol{w}_{q/k/v}$ represents $\sigma \boldsymbol{w}_q$, $\sigma \boldsymbol{w}_k$ or $\sigma \boldsymbol{w}_v$, $\alpha_{q/k/v}$ denotes α_q, α_k or α_v. The essence of the above optimization problem is to determine whether the q, k, and v weights need to be updated.

To solve this problem, we explicitly formulate the update strategy of q, k, and v weights as a constrained optimization problem that balances pruning error minimization and attention distribution preservation. Specifically, we design the search space using the binary update status $\{\text{update}, \text{non-update}\}$ for each of the q, k, and v weights in attention layers: "update" enables weight adjustment to compensate for pruning-induced errors, while "non-update" preserves the original weight structure to maintain the long-tail attention distribution. To identify the optimal configuration, we adopt perplexity (ppl) as the objective metric, which comprehensively reflects both the model's language modeling performance (linked to pruning error $L_{q/k/v}$) and the fidelity of attention patterns (linked to KL divergence of distributions). This dynamic adaptation process leverages the cross-layer consistency of outlier ratios in $q/k/v$ weights (e.g., 98% layers in LLaMA-2-13B show the highest outlier ratio in k matrix, see Figure 10 in Appendix D) to reduce the search space, ensuring efficiency while aligning with the dual optimization goal of minimizing error and preserving distribution.

Experiments

Experimental Settings

We compare D^2Prune with three main baselines, including weight-update pruning methods such as SparseGPT (Frantar and Alistarh 2023) and non-weight-update pruning methods such as Wanda (Sun et al. 2023) and Pruner-Zero (Dong et al. 2024), and all experiments are conducted on an NVIDIA A40 GPU. Moreover, given the potential of our dynamic attention update mechanism for global modeling, we also compare our method with a global pruning approach, SparseLLM (Bai

Sparsity	Method	OPT-125M		LLaMA-2-7B		LLaMA-2-13B		LLaMA-2-70B		LLaMA-3-8B	
		PPL ↓	ACC ↑	PPL ↓	ACC ↑	PPL ↓	ACC ↑	PPL ↓	ACC ↑	PPL ↓	ACC ↑
0	Dense	27.66	39.68	5.12	64.38	4.57	67.06	3.12	71.50	5.54	68.40
50	SparseGPT	36.85	39.82	6.52	60.35	5.63	64.87	3.98	71.35	8.56	63.13
	Wanda	38.88	39.75	6.44	60.46	5.58	64.17	3.98	70.96	9.06	61.02
	Pruner-Zero	38.80	39.09	6.43	59.26	5.57	64.21	-	-	8.52	60.28
	D^2Prune **(Ours)**	**34.98**	**40.39**	**6.36**	**61.06**	**5.53**	**65.90**	**3.93**	**71.60**	**8.34**	**63.58**
60	SparseGPT	59.46	39.37	9.56	55.34	7.77	60.26	4.98	70.09	14.40	55.39
	Wanda	74.39	39.45	9.89	53.91	7.87	59.57	4.99	69.04	22.80	48.12
	Pruner-Zero	68.07	39.32	10.34	52.52	7.82	58.12	-	-	20.30	50.65
	D^2Prune **(Ours)**	**52.10**	**40.14**	**9.05**	**56.08**	**7.49**	**60.81**	4.88	**70.65**	**13.44**	**56.95**
70	SparseGPT	218.29	36.31	29.62	44.75	18.20	48.12	8.61	63.49	38.85	43.56
	Wanda	347.42	35.43	84.92	36.68	44.86	39.75	40.27	60.44	114.30	37.07
	Pruner-Zero	317.87	36.88	151.92	38.09	44.76	43.33	-	-	280.33	36.17
	D^2Prune **(Ours)**	**160.81**	**38.09**	**21.10**	**47.97**	**16.51**	**48.32**	**8.17**	**64.06**	**33.37**	**44.82**
80	SparseGPT	2140.55	35.08	102.43	36.23	99.14	38.28	25.86	47.71	178.01	36.95
	Wanda	1920.63	34.93	5107.20	33.72	1384.40	34.67	156.68	37.98	2245.91	34.87
	Pruner-Zero	1251.38	35.06	10244.70	34.83	2040.65	35.09	-	-	10420.01	35.31
	D^2Prune **(Ours)**	**1038.87**	**36.29**	**92.68**	**39.09**	**76.80**	**39.42**	**21.37**	**48.09**	**151.47**	**38.73**
2:4	SparseGPT	59.76		10.18		8.39		5.32		14.16	
	Wanda	79.80	-	11.35	-	8.37	-	5.18	-	22.86	-
	Pruner-Zero	70.92		11.16		8.05		-		23.56	
	D^2Prune **(Ours)**	**59.43**		**10.00**		**8.02**		**5.12**		**14.10**	
3:4	SparseGPT	1365.58		154.23		147.23		54.84		281.74	
	Wanda	2497.68	-	3111.14	-	5815.71	-	386.57	-	13054.17	-
	Pruner-Zero	2946.15		7913.18		4134.56		-		854085.30	
	D^2Prune **(Ours)**	**1346.36**		**136.89**		**143.69**		**48.06**		**190.35**	

Table 1: Pruning comparison across language modeling and zero-shot tasks at different sparsity levels (middle: 50%, 60% and 2:4, high: 70%, 80% and 3:4).

et al. 2024), which iteratively updates activations and weights in the MLP modules to minimize global error detailed results in the Appendix B.5). In addition, we further evaluate our method on the latest large language models Qwen3-8B and 14B (Bai et al. 2023), as well as the vision transformer model DeiT (Touvron et al. 2021) built on the ViT (Dosovitskiy et al. 2020) architecture detailed in Appendix B.3). Following prior studies, we assess and compare various pruning methods based on language modeling with the perplexity metric (ppl) and the zero-shot accuracy for downstream tasks. For all pruning algorithms, we use the C4 (Raffel et al. 2020) training set as the calibration dataset, and we use 128 samples and segment the pre-trained LLMs according to their maximum embedding dimensions. For the language modeling comparison, we assess the perplexity on the WikiText2 (Merity et al. 2016) test set. For the zero-shot comparison, we use seven benchmark tasks including BoolQ (Clark et al. 2019), HellaSwag (Zellers et al. 2019), WinoGrande, RTE (Wang et al. 2018), ARC-c and ARC-e (ARC Easy and Challeng (Clark et al. 2018)), and OBQA (Mihaylov et al. 2018) from the EleutherAI LM Harness (Gao et al. 2023). All the datasets are sourced from the HuggingFace Datasets library.

Experimental Results

Unified Evaluation on Language Modeling and Zero-Shot. Following SparseGPT, Wanda and Pruner-Zero, we provide a unified pruning performance evaluation result in Table 1 on both language modeling (in terms of perplexity, PPL↓) and zero-shot tasks (measured by average accuracy, ACC↑) under various sparsity levels and model sizes. Overall, D^2Prune consistently outperforms existing pruning baselines across both moderate (50%–60%, 2:4) and high sparsity (70%–80%, 3:4) regimes. Compared to weight-update methods (e.g., SparseGPT), D^2Prune achieves on average a 3.1% accuracy gain and 16% lower perplexity. Against non-weight-update methods (e.g., Wanda, Pruner-Zero), the perplexity reduction reaches up to 86%. Notably, D^2Prune surpasses the dense model on LLaMA-2-70B at 50% sparsity in zero-shot accuracy, demonstrating strong generalization. Due to resource limitations, Pruner-Zero fails to perform pruning on LLaMA-2-70B even with a single A40 or A100 GPU. Therefore, the corresponding entries are marked as "–" in the table. For semi-structured pruning (e.g., 2:4 and 3:4 patterns), we report only the perplexity results, while accuracy is marked as "–" to indicate that it was not evaluated. The detailed zero-shot results on seven downstream tasks are shown in Appendix B.1, providing a comprehensive understanding of the zero-shot capabilities for the discussed models.

Expanding Experiment Results. (1) We further compare with the more computationally intensive **global pruning** method SparseLLM (Bai et al. 2024) (detailed in Appendix B.5 and (2) validate D^2Prune on **vision transformers** (e.g., DeiT (Touvron et al. 2021)) and **recent LLMs like Qwen3 (Bai et al. 2023)** (detailed in Appendix B.3, confirming its broad applicability across architectures and tasks.

Method	50	60	70	80
SparseGPT	8.56	14.40	38.85	178.01
D^2-Sparsegpt	**8.48**	**13.66**	**38.64**	**170.92**
Wanda	9.06	22.80	114.30	2245.91
D^2-Wanda	**8.92**	**22.54**	**106.58**	**1414.32**

Table 2: Ablations study of dual Taylor expansions on LLaMA-3-8B (WikiText2 perplexity, unstructured 50%-80% sparsity, Dense: 5.54). "D^2-" indicates the first- and second-order activation partial derivative term S_X (see Algorithm 1, detailed in Appendix A).

(3) In addition, to comprehensively evaluate the reasoning capabilities of pruned models in real-world scenarios, we conduct **in-context learning (ICL) experiments on the GSM8K** benchmark (Cobbe et al. 2021) (detailed in Appendix B.2). The results demonstrate that in the few-shot setting, D^2Prune consistently outperforms baselines. (3) Finally, we provide **pruning efficiency and speedup** experiment results in the Appendix E.

Ablation study

Effectiveness of the Dual Taylor Expansion for mask selection & weight update. To validate the effectiveness of the dual Taylor expansion for mask selection and weight update, we incorporated the first and second order activation partial derivatives to SparseGPT and Wanda for mask selection and weight update, and we denote these methods with the prefix "D^2-". As shown in Table 2, D^2-SparseGPT consistently outperforms SparseGPT across all models and sparsity levels, and D^2-Wanda consistently outperforms Wanda, demonstrating the effectiveness of the dual Taylor expansion. Besides, we perform the ablation study on key hyperparameters λ_1, λ_2 and s for dual Taylor expansion in Appendix B.4.

Effectiveness of Attention Distribution-Aware Dynamic Weight Update Strategy. As shown in Table 3, we perform unstructured pruning on the LLaMA-2-7B. Specifically, when either the q, k, or v weight is not updated (denoted as w/o q, w/o k, and w/o v), the pruning performance outperforms Wanda and is competitive with SparseGPT. Furthermore, our attention distribution-aware dynamic weight update strategy dynamically sets one of the q, k, and v weights to the non-update state while keeping the others in the update state, leading to optimal performance for all LLMs. This strongly supports the effectiveness of the attention distribution-aware dynamic weight update strategy.

Analysis

Impact of sparsity on the update configurations of q, k, v weights. When searching for optimal q, k, and v update configurations, we evaluated various settings under different sparsity levels. The configuration yielding the lowest perplexity was selected as the optimal setup. The results in Table 3 and Appendix D reveal a key insight: the optimal q, k, and v update configuration remains consistent across all sparsity levels for each model, indicating that it is independent of sparsity.

Method	Weight-Update Layers	50	60	70	80
SparseGPT	ALL Layers	6.52	9.56	29.62	102.43
Wanda	None	6.44	9.89	84.92	5107.20
Pruner-Zero	None	6.43	10.34	151.92	10244.70
D^2Prune	ALL Layers	6.46	9.46	23.45	101.88
	None	6.43	9.75	81.62	3561.51
	w/o q	6.50	9.60	25.16	104.63
	w/o k	6.49	9.55	24.04	118.79
	w/o v	6.36	9.05	21.10	92.68
	Dynamic Update	**6.36**	**9.05**	**21.10**	**92.68**

Table 3: Ablations study of dynamic weight update for attention modules (q, k, v) on LLaMA-2-7B (WikiText2 perplexity, unstructured 50%-80% sparsity, Dense: 5.12).

Method	Fine-tuning	60		70	
		PPL	ACC	PPL	ACC
SparseGPT	✗	9.56	55.34	29.62	44.75
	LoRA	7.18	57.82	10.18	49.73
Wanda	✗	9.86	53.91	84.92	36.68
	LoRA	7.26	56.45	11.81	46.43
D^2Prune	✗	**9.05**	**56.06**	**21.10**	**47.97**
	LoRA	**7.05**	**60.59**	**9.70**	**51.37**

Table 4: Fine-tuning can mitigate the perplexity gap to dense LLM. D^2Prune significantly outperforms all other pruning approaches even after LoRA fine-tuning.

LoRA Fine-tuning. We evaluated the impact of fine-tuning on the pruned models. Specifically, we employ LoRA (Hu et al. 2022) (r=8, α=16) for fine-tuning with one epoch on the C4 training datasets (training sample size is 30k) using one GPU (40G memory) and 15 hours. We conducted experiments on LLaMA-2-7B, involving unstructured pruning at 60% and 70% sparsity levels. Table 4 reports the perplexity on the WikiText2 dataset and mean accuracy of zero-shot tasks. We can observe that LoRA fine-tuning can effectively restore the performance of all pruned models, and D^2Prune consistently outperforms all other pruning methods in both accuracy and perplexity. This further demonstrates the effectiveness of D^2Prune.

Conclusion

We propose D^2Prune, a novel pruning algorithm for compressing large language models (LLMs). By extending error minimization with dual Taylor expansion and attention distribution awareness, D^2Prune improves the accuracy of pruning mask selection and weight updates. It explicitly models the dual effects of activation and weight variations, and introduces a dynamic weight update strategy for Transformer attention modules to reduce attention distribution errors. Extensive experiments show that D^2Prune consistently outperforms recent post-training pruning methods.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Nos. 62572085 and 62472058).

References

Bai, G.; Li, Y.; Ling, C.; Kim, K.; and Zhao, L. 2024. SparseLLM: Towards global pruning of pre-trained language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; Hui, B.; Ji, L.; Li, M.; Lin, J.; Lin, R.; Liu, D.; Liu, G.; Lu, C.; Lu, K.; Ma, J.; Men, R.; Ren, X.; Ren, X.; Tan, C.; Tan, S.; Tu, J.; Wang, P.; Wang, S.; Wang, W.; Wu, S.; Xu, B.; Xu, J.; Yang, A.; Yang, H.; Yang, J.; Yang, S.; Yao, Y.; Yu, B.; Yuan, H.; Yuan, Z.; Zhang, J.; Zhang, X.; Zhang, Y.; Zhang, Z.; Zhou, C.; Zhou, J.; Zhou, X.; and Zhu, T. 2023. Qwen Technical Report. *arXiv preprint arXiv:2309.16609*.

Blalock, D.; Gonzalez Ortiz, J. J.; Frankle, J.; and Guttag, J. 2020. What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2: 129–146.

Boža, V. 2024. Fast and effective weight update for pruned large language models. *arXiv preprint arXiv:2401.02938*.

Chen, J.; Agarwal, A.; Abdelkarim, S.; Zhu, D.; and Elhoseiny, M. 2022. Reltransformer: A transformer-based long-tail visual relationship recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19507–19517.

Chi, H.; Li, H.; Yang, W.; Liu, F.; Lan, L.; Ren, X.; Liu, T.; and Han, B. 2024. Unveiling causal reasoning in large language models: Reality or mirage? *Advances in Neural Information Processing Systems*, 37: 96640–96670.

Chijiwa, D.; Yamaguchi, S.; Ida, Y.; Umakoshi, K.; and Inoue, T. 2021. Pruning randomly initialized neural networks with iterative randomization. *Advances in neural information processing systems*, 34: 4503–4513.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dong, P.; Li, L.; Tang, Z.; Liu, X.; Pan, X.; Wang, Q.; and Chu, X. 2024. Pruner-Zero: Evolving Symbolic Pruning Metric from scratch for Large Language Models. *arXiv preprint arXiv:2406.02924*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Frantar, E.; and Alistarh, D. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 10323–10337. PMLR.

Gao, L.; Tow, J.; Abbasi, B.; Biderman, S.; Black, S.; DiPofi, A.; Foster, C.; Golding, L.; Hsu, J.; Le Noac'h, A.; et al. 2023. A framework for few-shot language model evaluation, 12 2023. *URL https://zenodo. org/records/10256836*, 7.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*.

Han, S.; Pool, J.; Tran, J.; and Dally, W. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

Hassibi, B.; Stork, D. G.; and Wolff, G. J. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, 293–299. IEEE.

Hoang, D. N.; and Liu, S. 2023. Revisiting pruning at initialization through the lens of ramanujan graph. *ICLR 2023*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Ji, T.; Jain, S.; Ferdman, M.; Milder, P.; Schwartz, H. A.; and Balasubramanian, N. 2021. On the distribution, sparsity, and inference-time quantization of attention values in transformers. *arXiv preprint arXiv:2106.01335*.

Kim, B.-K.; Kim, G.; Kim, T.-H.; Castells, T.; Choi, S.; Shin, J.; and Song, H.-K. 2024. Shortened llama: A simple depth pruning for large language models. *arXiv preprint arXiv:2402.02834*, 11.

Kwon, W.; Kim, S.; Mahoney, M. W.; Hassoun, J.; Keutzer, K.; and Gholami, A. 2022. A fast post-training pruning framework for transformers. *Advances in Neural Information Processing Systems*, 35: 24101–24116.

LeCun, Y.; Denker, J.; and Solla, S. 1989. Optimal Brain Damage. In Touretzky, D., ed., *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann.

Li, J.; Tang, T.; Zhao, W. X.; Nie, J.-Y.; and Wen, J.-R. 2024a. Pre-trained language models for text generation: A survey. *ACM Computing Surveys*, 56(9): 1–39.

Li, L.; Dong, P.; Tang, Z.; Liu, X.; Wang, Q.; Luo, W.; Xue, W.; Liu, Q.; Chu, X.; and Guo, Y. 2024b. Discovering sparsity allocation for layer-wise pruning of large language models. *Advances in Neural Information Processing Systems*, 37: 141292–141317.

Liu, Z.; Sun, M.; Zhou, T.; Huang, G.; and Darrell, T. 2018. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Mo, Y.; Qin, H.; Dong, Y.; Zhu, Z.; and Li, Z. 2024. Large language model (llm) ai text generation detection based on transformer deep learning algorithm. *arXiv preprint arXiv:2405.06652*.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140): 1–67.

Reyhan, Z. A.; Rahnamayan, S.; and Bidgoli, A. A. 2024. Novel Post-Training Structure-Agnostic Weight Pruning Technique for Deep Neural Networks. In *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2206–2212. IEEE.

Sreenivasan, K.; Sohn, J.-y.; Yang, L.; Grinde, M.; Nagle, A.; Wang, H.; Xing, E.; Lee, K.; and Papailiopoulos, D. 2022. Rare gems: Finding lottery tickets at initialization. *Advances in neural information processing systems*, 35: 14529–14540.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Tanaka, H.; Kunin, D.; Yamins, D. L.; and Ganguli, S. 2020. Pruning neural networks without any data by iteratively conserving synaptic flow. *NeurIPS*, 33: 6377–6389.

Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; and Jégou, H. 2021. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, 10347–10357. PMLR.

van der Ouderaa, T. F.; Nagel, M.; Van Baalen, M.; Asano, Y. M.; and Blankevoort, T. 2023. The llm surgeon. *arXiv preprint arXiv:2312.17244*.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Wang, Z. 2024. CausalBench: A Comprehensive Benchmark for Evaluating Causal Reasoning Capabilities of Large Language Models. In *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, 143–151.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Zhang, Y.; Bai, H.; Lin, H.; Zhao, J.; Hou, L.; and Cannistraci, C. V. 2024. Plug-and-play: An efficient post-training pruning method for large language models. In *The Twelfth International Conference on Learning Representations*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 11106–11115.