

LLMs can Compress LLMs: Adaptive Pruning by Agents

Sai Varun Kodathala¹ (✉) and Rakesh Vunnam²

¹ Research and Development, Sports Vision, Inc., Minnetonka, MN 55305, USA
varun@sportsvision.ai

² Research and Development, Vizworld Inc., Minnetonka, MN 55305, USA
rakesh@vizworld.ai

Abstract. As Large Language Models (LLMs) continue to scale, post-training pruning has emerged as a promising approach to reduce computational costs while preserving performance. Existing methods such as SparseGPT and Wanda achieve high sparsity through layer-wise weight reconstruction or activation-aware magnitude pruning, but rely on uniform or hand-crafted heuristics to determine per-layer sparsity ratios. Moreover, recent work has shown that pruned LLMs suffer from severe factual knowledge degradation, with structured pruning methods experiencing near-total collapse in factual question-answering capabilities. We introduce agent-guided pruning, where a foundation model acts as an adaptive pruning agent to intelligently select which layers to prune at each iteration while preserving critical knowledge pathways. Our method constructs layer-wise sensitivity profiles by combining Wanda-inspired weight-activation metrics with gradient importance scores, normalized as z-scores for model-agnostic comparison. These statistics are processed by an LLM agent equipped with self-reflection capabilities, enabling it to learn from previous pruning outcomes and iteratively refine its strategy. A checkpoint rollback mechanism maintains model quality by reverting when perplexity degradation exceeds a threshold. We evaluate our approach on Qwen3 models (4B and 8B parameters) at approximately 45% sparsity, demonstrating substantial improvements over structured pruning baselines: 56% relative improvement in MMLU accuracy, 19× better factual knowledge retention on FreebaseQA, and 69% lower perplexity degradation. Notably, our framework requires no retraining, operates in a model-agnostic manner, and exhibits effective self-correction with only 2-4 rollbacks across 21-40 iterations, demonstrating that foundation models can effectively guide the compression of other foundation models.

Keywords: Model Compression · Adaptive Pruning · Self-Reflection.

1 Introduction

Large Language Models (LLMs) have achieved remarkable success across diverse natural language processing tasks, demonstrating emergent capabilities in reasoning, generation, and knowledge retrieval [9, 7, 8]. However, their deployment remains challenging due to substantial computational and memory

requirements. For instance, models with billions of parameters necessitate multiple high-memory GPUs and incur significant inference latency [1], limiting their accessibility and practical applicability in resource-constrained environments.

Model compression techniques, particularly neural network pruning, have emerged as promising solutions to mitigate these challenges [10, 11]. Post-training pruning methods offer particular appeal for LLMs, as they eliminate the need for expensive retraining on massive corpora [1, 2]. Recent approaches such as SparseGPT [1] and Wanda [2] have demonstrated that LLMs can be pruned to 50% sparsity with minimal perplexity degradation, achieving this through layer-wise weight reconstruction and activation-aware magnitude pruning, respectively. Extensions like Wanda++ [3] further improve upon these foundations by incorporating regional gradients at the decoder-block level.

Despite these advances, existing pruning methods share a critical limitation: they rely on *uniform* sparsity ratios across layers or employ hand-crafted heuristics to determine per-layer sparsity [1–3]. This one-size-fits-all approach fails to account for the heterogeneous importance of different layers in preserving model capabilities. Moreover, recent evaluation benchmarks have revealed a severe problem with pruned LLMs: *catastrophic factual knowledge degradation*. The LLM-KICK benchmark [5] demonstrates that structured pruning methods experience near-total collapse in factual question-answering performance, with N:M sparsity patterns losing over 97% accuracy on knowledge-intensive tasks even at modest sparsity ratios (25–30%), despite negligible perplexity increases. This disconnect between perplexity and actual task performance underscores the inadequacy of current evaluation metrics and pruning strategies.

In this work, we introduce *agent-guided pruning*, a novel framework where a foundation model acts as an adaptive pruning oracle to intelligently determine which layers to prune at each iteration. Our approach is motivated by two key insights: First, LLMs possess sophisticated reasoning capabilities that can be leveraged for optimization tasks beyond their traditional use in text generation [15, 16]. Second, the heterogeneous sensitivity of layers to pruning necessitates adaptive, context-aware decision-making rather than fixed heuristics [4].

Our method constructs comprehensive layer-wise sensitivity profiles by combining Wanda-inspired weight-activation metrics [2] with gradient importance scores, normalized as z-scores for model-agnostic comparison. These statistics, along with iterative feedback from previous pruning outcomes, are provided to an LLM agent equipped with self-reflection capabilities. The agent learns from past decisions, reasoning about which layers preserve critical knowledge pathways and which can safely be pruned. A checkpoint rollback mechanism serves as a safety net, reverting the model when perplexity degradation exceeds a specified threshold and penalizing the agent through the feedback loop.

We evaluate our approach on Qwen3 models (4B and 8B parameters) at approximately 45% sparsity, comparing against structured pruning baselines. Our results demonstrate substantial improvements: 56% relative improvement in MMLU accuracy, 19 \times better factual knowledge retention on FreebaseQA, and 69% lower perplexity degradation compared to the best performing 4:8 struc-

tured baseline. Notably, our framework requires no retraining, operates in a model-agnostic manner, and exhibits effective self-correction with only 9.5-10% rollback rates across 21-40 pruning iterations. These results establish that foundation models can effectively guide the compression of other foundation models, opening a new paradigm for automated neural architecture optimization.

1.1 Main Contributions

The main contributions of this work are as follows:

- We introduce the first framework to use an LLM as an adaptive pruning agent, enabling intelligent, iterative layer selection for neural network compression without manual heuristic design.
- We develop a self-reflection mechanism that enables the pruning agent to learn from previous decisions, progressively refining its strategy through iterative feedback.
- We propose a model-agnostic statistical profiling approach using z-score normalization of Wanda metrics and gradient importance, enabling comparison across heterogeneous layer types.
- We demonstrate a checkpoint rollback mechanism that maintains model quality by reverting unsuccessful pruning attempts, with remarkably low rollback rates (9.5-10%) indicating effective agent learning.
- We achieve substantial empirical improvements over structured pruning baselines, particularly in preserving factual knowledge ($19\times$ improvement) and maintaining general task performance (56% MMLU improvement), directly addressing the critical knowledge degradation problem revealed by recent benchmarks [5].

2 Related Work

2.1 Post-Training Pruning for Large Language Models

The computational demands of billion-parameter LLMs have motivated extensive research in post-training compression methods that avoid expensive retraining. **SparseGPT** [1] pioneered one-shot pruning for massive language models by framing the problem as layer-wise sparse regression, using second-order information to reconstruct weights after pruning. The method achieves 50-60% unstructured sparsity on models like OPT-175B and BLOOM-176B in under 4.5 hours, with minimal perplexity increase. However, SparseGPT’s reliance on Hessian approximations incurs significant computational overhead and memory requirements.

Wanda [2] introduced a simpler alternative, pruning weights by the product of their magnitudes and input activation norms on a per-output basis. Motivated by emergent large-magnitude features in LLMs [13], Wanda eliminates the need for weight updates or second-order information, achieving competitive

performance with SparseGPT while requiring no retraining. The method demonstrates that effective sparse subnetworks exist exactly within pretrained LLMs without modification, and shows that pruning becomes more effective as model size increases, with 50% sparse LLaMA-65B matching the zero-shot performance of its dense counterpart.

Recent work has sought to enhance these methods through gradient information. **Wanda++** [3] introduces regional gradients computed at the decoder-block level, combined with regional optimization to minimize pruning-induced output discrepancies. This achieves up to 32% perplexity improvement over Wanda on language modeling tasks while maintaining efficiency (pruning a 7B model in under 10 minutes on a single H100 GPU).

2.2 Structured Pruning Approaches

While unstructured pruning achieves high sparsity, it often fails to deliver practical speedups on existing hardware [12]. **Structured pruning** methods remove entire architectural components (attention heads, neurons, layers), enabling direct acceleration without specialized hardware [4].

LLM-Pruner [4] performs task-agnostic structural pruning by identifying coupled structures through dependency detection and estimating their importance using first-order gradients and Hessian approximations. The method enables efficient recovery through LoRA tuning [14] in merely 3 hours with 50K samples, achieving 94.97% performance retention at 20% parameter reduction. However, the approach still relies on simplified pairwise neuron dependencies and requires manual specification of pruning ratios.

N:M semi-structured sparsity patterns (e.g., 2:4, 4:8) offer a middle ground, achieving hardware acceleration through regular sparse patterns while maintaining relatively high density [1, 2]. Both SparseGPT and Wanda support these patterns, generalizing their pruning criteria to enforce zero constraints in blocks of M consecutive weights.

2.3 Evaluation of Compressed LLMs

A critical issue highlighted by recent work is the inadequacy of perplexity as an evaluation metric for compressed LLMs. **LLM-KICK** [5] introduced a comprehensive benchmark revealing that all pruning methods suffer significant performance degradation on knowledge-intensive tasks, even at trivial sparsity ratios (25-30%), despite negligible perplexity increases. The benchmark demonstrates that structured N:M pruning methods experience catastrophic failure on factual question-answering (e.g., FreebaseQA), losing over 97% of accuracy, while maintaining reasonable perplexity scores. This finding challenges the prevailing assumption that perplexity adequately captures model capability degradation and motivates our focus on preserving factual knowledge through adaptive layer selection.

Beyond factual knowledge, recent studies have identified a dichotomy in pruning effects: while parametric knowledge degrades predictably, instruction-following capabilities can paradoxically improve with pruning [6], suggesting complex trade-offs in how compression affects different cognitive functions.

2.4 Learning-Based Compression and Meta-Optimization

Our work is inspired by recent advances in using learned policies for neural architecture decisions. AutoML approaches have demonstrated that learned strategies can outperform hand-crafted heuristics for architecture search [19, 20]. More recently, LLMs have been applied to optimization tasks beyond text generation, including code generation [21], theorem proving [22], and mathematical reasoning [15].

The concept of using foundation models as agents for optimization tasks has gained traction through work on tool use [18], planning [16], and self-reflection [17]. These works demonstrate that LLMs can engage in complex reasoning about abstract spaces and learn from feedback. However, to our knowledge, no prior work has applied LLM agents to the neural network pruning problem. Our approach extends this paradigm by showing that foundation models can effectively guide the compression of other foundation models, opening new possibilities for automated model optimization.

Unlike prior pruning methods that rely on fixed importance metrics or uniform compression strategies, our agent-guided framework adapts its strategy based on observed outcomes, embodying a form of meta-learning where the pruning policy itself is refined through experience. This is particularly valuable for the heterogeneous layer sensitivities observed in modern LLMs, where different components play vastly different roles in preserving capabilities.

3 Method

3.1 Overview

Our agent-guided pruning framework consists of four key components: (1) layer-wise sensitivity profiling using activation and gradient statistics, (2) an LLM agent that iteratively selects layers to prune based on normalized metrics, (3) a self-reflection mechanism that enables the agent to learn from previous decisions, and (4) a checkpoint rollback system that maintains model quality. Unlike prior work that applies uniform or hand-crafted sparsity ratios, our method adaptively determines per-layer pruning amounts through learned decision-making.

3.2 Layer Sensitivity Profiling

For each linear layer ℓ in the model, we construct a comprehensive sensitivity profile combining multiple statistical measures. Following Wanda [2], we compute the weight-activation metric:

$$\mathbf{S}_\ell = |\mathbf{W}_\ell| \odot \|\mathbf{X}_\ell\|_2 \quad (1)$$

where $\mathbf{W}_\ell \in \mathbb{R}^{d_{out} \times d_{in}}$ is the weight matrix, $\mathbf{X}_\ell \in \mathbb{R}^{N \times d_{in}}$ are the input activations collected over N calibration samples, $|\cdot|$ denotes element-wise absolute value, \odot is the Hadamard product, and $\|\cdot\|_2$ computes the ℓ_2 -norm across samples. The sensitivity score for layer ℓ is defined as the k -th percentile of active (non-zero) weights in \mathbf{S}_ℓ , where $k = 10$ in our experiments.

To capture gradient information inspired by Wanda++ [3], we compute gradient importance as:

$$\mathbf{G}_\ell = \frac{1}{M} \sum_{i=1}^M |\nabla \mathbf{w}_\ell \mathcal{L}_i| \quad (2)$$

where \mathcal{L}_i is the language modeling loss on calibration sample i and M is the number of gradient samples. We collect gradients every third iteration to balance computational cost with information gain.

For model-agnostic comparison across heterogeneous layers, we normalize both metrics using z-score standardization:

$$z_\ell^{(s)} = \frac{s_\ell - \mu_s}{\sigma_s + \epsilon}, \quad z_\ell^{(g)} = \frac{g_\ell - \mu_g}{\sigma_g + \epsilon} \quad (3)$$

where s_ℓ and g_ℓ are the raw sensitivity and gradient scores for layer ℓ , μ and σ denote mean and standard deviation across all layers, and $\epsilon = 10^{-9}$ prevents division by zero. Negative z-scores indicate below-average sensitivity (safer to prune), while positive z-scores indicate above-average sensitivity (riskier to prune).

The complete profile for layer ℓ is: $\mathcal{P}_\ell = \{z_\ell^{(s)}, z_\ell^{(g)}, \rho_\ell\}$, where ρ_ℓ is the current sparsity ratio of layer ℓ .

3.3 LLM Agent Design

At each iteration t , we query a foundation model (*gemini-3-flash-preview*) to select layers for pruning. The agent receives:

- Current global sparsity ρ_t and target sparsity ρ^*
- Layer profiles $\{\mathcal{P}_\ell\}_{\ell=1}^L$ sorted by sensitivity z-score
- Current and baseline perplexity: $\text{PPL}_t, \text{PPL}_0$
- Feedback summary from iteration $t - 1$ (if $t > 1$)

The agent is instructed to reason about which layers are safe to prune based on the statistical profiles, considering both the gap remaining to target sparsity and the model’s current health (perplexity degradation). The agent outputs structured JSON containing:

- **reasoning**: Natural language explanation of the pruning strategy
- **stop_pruning**: Boolean indicating whether to terminate

- `layer_decisions`: List of (layer name, additional sparsity) pairs

where additional sparsity $\delta_\ell \in [0.01, 0.15]$ specifies how much additional sparsity to induce in layer ℓ . We use structured output with JSON schema to ensure reliable parsing.

3.4 Self-Reflection Mechanism

To enable the agent to learn from its decisions, we implement an iterative feedback loop. After pruning at iteration t , we compute:

- Sparsity gain: $\Delta\rho_t = \rho_{t+1} - \rho_t$
- Perplexity change: $\Delta_{\text{PPL}} = \frac{\text{PPL}_{t+1} - \text{PPL}_t}{\text{PPL}_t} \times 100\%$

At iteration $t + 1$, the agent receives a feedback summary containing:

- Its previous reasoning and layer selections
- The observed sparsity gain and perplexity change
- A qualitative assessment (e.g., "Excellent - High sparsity gain with minimal PPL impact")

This feedback enables the agent to recognize effective patterns (e.g., prioritizing layers with highly negative z-scores) and adjust its strategy (e.g., becoming more conservative as perplexity degrades). The system prompt explicitly instructs the agent to analyze past decisions and refine its approach accordingly.

3.5 Checkpoint Rollback Mechanism

To prevent catastrophic degradation, we implement a safety mechanism that monitors perplexity changes. Before each pruning operation, we save the current model state. After pruning, if:

$$\frac{\text{PPL}_{t+1} - \text{PPL}_t}{\text{PPL}_t} > \tau \quad (4)$$

where $\tau = 0.15$ (15% threshold), we rollback to the previous checkpoint, discard the current iteration's decisions, and provide negative feedback to the agent. This rollback is communicated through the self-reflection loop with the assessment "Poor - Excessive PPL degradation, consider more conservative approach."

3.6 Complete Algorithm

Algorithm 1 presents the complete agent-guided pruning procedure.

Algorithm 1 LLM-Guided Pruning with Self-Reflection

Require: Model \mathcal{M} , calibration data \mathcal{D} , target sparsity s_{target}

Require: LLM \mathcal{L} , rollback threshold τ_{rollback}

- 1: Compute baseline perplexity PPL_0 on \mathcal{D}
- 2: Initialize checkpoint $\mathcal{C} \leftarrow \emptyset$, iteration memory $\mathcal{H} \leftarrow \emptyset$
- 3: **for** $t = 1, \dots, T$ **do**
- 4: Save checkpoint $\mathcal{C} \leftarrow \{\mathbf{W}^{(l)}\}_{l=1}^L$, $\text{PPL}_{\mathcal{C}} \leftarrow \text{PPL}_{t-1}$
- 5: Collect activations $\{\mathbf{A}^{(l)}\}$ and gradients $\{\mathbf{G}^{(l)}\}$ from \mathcal{D} // Wanda++ metrics
- 6: **for** layer $l = 1, \dots, L$ **do**
- 7: Compute sensitivity: $\mathbf{S}^{(l)} = |\mathbf{W}^{(l)}| \odot \mathbf{A}^{(l)}$
- 8: Compute z-scores: $z_{\text{sens}}^{(l)}, z_{\text{grad}}^{(l)}$ from $\mathbf{S}^{(l)}, \mathbf{G}^{(l)}$
- 9: **end for**
- 10: Query LLM: $\pi_t \leftarrow \mathcal{L}(\{z_{\text{sens}}^{(l)}, z_{\text{grad}}^{(l)}, s_t^{(l)}\}, \mathcal{H}_{t-1})$ // Layer stats + feedback
- 11: **if** $\pi_t.\text{stop} = \text{true}$ **then**
- 12: **break**
- 13: **end if**
- 14: **for** $(l, \Delta s) \in \pi_t.\text{decisions}$ **do**
- 15: Prune layer l by sparsity Δs using $\mathbf{S}^{(l)}$ (Wanda)
- 16: **end for**
- 17: Compute new perplexity PPL_t and sparsity s_t
- 18: **if** $\text{PPL}_t/\text{PPL}_{t-1} > \tau_{\text{rollback}}$ **then**
- 19: Restore checkpoint: $\{\mathbf{W}^{(l)}\} \leftarrow \mathcal{C}$, $\text{PPL}_t \leftarrow \text{PPL}_{\mathcal{C}}$ // Rollback
- 20: **end if**
- 21: Store feedback: $\mathcal{H}_t \leftarrow \{\pi_t, s_{t-1}, s_t, \text{PPL}_{t-1}, \text{PPL}_t\}$ // Self-reflection
- 22: **if** $s_t \geq s_{\text{target}}$ **then**
- 23: **break**
- 24: **end if**
- 25: **end for**
- 26: **return** Pruned model \mathcal{M}

4 Experiments

4.1 Experimental Setup

Models. We evaluate our method on two Qwen3 models: Qwen3-4B and Qwen3-8B [27]. These models represent different scales within the same architectural family, enabling us to assess generalization across model sizes.

Baselines. We compare against two structured pruning methods that achieve similar sparsity levels:

- **2:4 Structured Pruning:** Prunes 2 out of every 4 consecutive weights, achieving ~42-45% sparsity. This pattern is hardware-efficient on NVIDIA GPUs with Ampere architecture and beyond.
- **4:8 Structured Pruning:** Prunes 4 out of every 8 consecutive weights, also achieving ~42-45% sparsity with better density than 2:4.

Both baselines use magnitude-based pruning as implemented in standard pruning libraries. We do not include unstructured magnitude pruning or SparseGPT

Table 1. MMLU performance by category for Qwen3-8B. Our agent-guided method maintains substantially better performance than structured baselines across all knowledge domains, with Social Sciences showing the strongest retention at 79.2% of baseline performance.

Category	BASE (%)	2:4 (%)	4:8 (%)	Ours (%)
STEM	74.20	31.51	34.83	52.00
Humanities	75.85	28.95	34.37	54.42
Social Sciences	82.04	32.87	38.50	64.97
Other	77.60	31.59	38.09	58.75
Overall	77.38	31.35	36.29	56.67

as baselines because recent work [5] has shown that while these methods achieve good perplexity scores, they suffer from even worse factual knowledge degradation than structured methods.

Evaluation Protocol. We follow the LLM-KICK benchmark [5] evaluation protocol:

- **MMLU** [23]: 5-shot evaluation on 57 subjects covering STEM, humanities, social sciences, and other domains. We report overall accuracy and per-category breakdowns.
- **FreebaseQA** [24]: Factual question-answering on 20,358 questions. This metric directly measures factual knowledge retention.
- **WikiText-2 Perplexity** [25]: Language modeling performance on the full WikiText-2 test set.

All evaluations use the full datasets without truncation to ensure comprehensive assessment.

Implementation Details. We use 128 calibration samples of sequence length 2048 from the C4 dataset [26]. The LLM agent is *gemini-3-flash-preview* with temperature 0.5 to balance creativity and consistency. We set target sparsity to 50%, though the algorithm may stop earlier if the agent determines further pruning would be too harmful. Activation collection uses 16 samples, gradient collection uses 8 samples, and perplexity evaluation uses 32 samples. The roll-back threshold is $\tau = 0.15$ (15% perplexity increase). All experiments run on a single NVIDIA A100 80GB GPU.

4.2 Main Results

Tables 2 and 3 present the comprehensive results for Qwen3-8B and Qwen3-4B respectively.

For Qwen3-8B at 43% sparsity, our method achieves 56.67% MMLU accuracy, representing a 56.2% relative improvement over the 4:8 structured baseline. More dramatically, we retain 25.16% FreebaseQA accuracy compared to just 1.33%

Table 2. Comprehensive evaluation on Qwen3-8B at $\sim 43\%$ sparsity. Our agent-guided method substantially outperforms structured pruning baselines across all metrics, particularly in factual knowledge retention ($19\times$ improvement over 4:8).

Method	Sparsity (%)	MMLU (%)	FreebaseQA (%)	Perplexity (WikiText-2)
BASE (Dense)	0.00	77.38	50.56	9.72
2:4 Structured	42.40	31.35	0.22	103.01
4:8 Structured	42.40	36.29	1.33	60.67
Ours (Agent-Guided)	43.02	56.67	25.16	19.06
<i>Relative Improvement over Best Baseline (4:8):</i>				
Ours vs 4:8	+0.62	+56.2%	+1791%	-68.6%

Table 3. Comprehensive evaluation on Qwen3-4B at $\sim 45\%$ sparsity. Our method demonstrates consistent improvements over structured baselines, with particular strength in preserving general task performance (MMLU) and reducing perplexity degradation.

Method	Sparsity (%)	MMLU (%)	FreebaseQA (%)	Perplexity (WikiText-2)
BASE (Dense)	0.00	71.29	32.43	13.64
2:4 Structured	45.16	26.04	0.20	319.75
4:8 Structured	45.16	29.24	0.51	81.28
Ours (Agent-Guided)	45.58	44.43	2.08	39.40
<i>Relative Improvement over Best Baseline (4:8):</i>				
Ours vs 4:8	+0.42	+51.9%	+308%	-51.5%

for 4:8—a $19\times$ improvement that directly addresses the catastrophic factual knowledge collapse identified by LLM-KICK [5]. Perplexity increases by only 96% compared to 524% for 4:8, demonstrating substantially better language modeling preservation.

For Qwen3-4B at 45.58% sparsity, we observe consistent patterns: 51.9% relative MMLU improvement (44.43% vs 29.24%), 4.1× better FreebaseQA retention (2.08% vs 0.51%), and 51.5% lower perplexity degradation. These results demonstrate that agent-guided pruning generalizes effectively across model scales.

Table 1 presents the detailed MMLU breakdown by category for Qwen3-8B. Our method preserves performance across all categories, with particularly strong results in Social Sciences (79.2% of baseline) and Humanities (71.7% of baseline).

4.3 Agent Behavior Analysis

Table 4 summarizes the iteration statistics for both models. The Qwen3-8B model reached 50.73% sparsity in 21 iterations with only 2 rollbacks (9.5% roll-

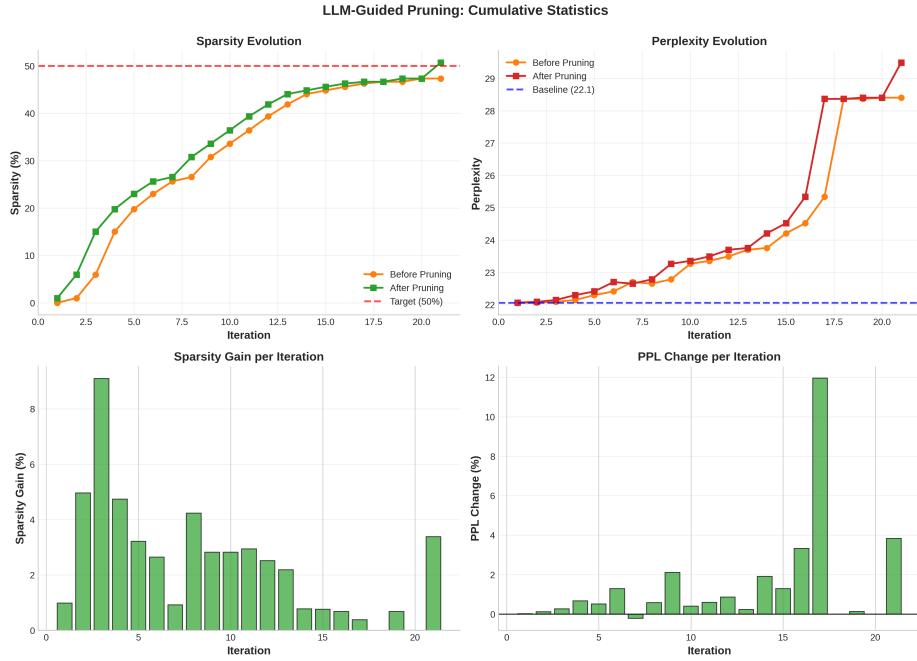


Fig. 1. Cumulative statistics for agent-guided pruning on Qwen3-8B across 21 iterations. Top left: Sparsity evolution showing gradual progression to 50% target. Top right: Perplexity evolution showing controlled degradation with 2 rollback events. Bottom left: Per-iteration sparsity gains, with most iterations achieving 1-3% progress. Bottom right: Per-iteration perplexity changes, showing the agent learns to keep PPL increases below 2% in most iterations.

back rate), while Qwen3-4B reached 50.46% sparsity in 40 iterations with 4 rollbacks (10% rollback rate). These low rollback rates indicate that the agent makes predominantly correct decisions, with the self-reflection mechanism enabling effective learning across iterations.

Figures 1 and 2 visualize the iterative pruning process for both models. The top panels show the evolution of sparsity (left) and perplexity (right) across iterations. The orange lines represent the state before each pruning decision, while green lines show the state after pruning. The close tracking of these lines demonstrates that the agent makes small, incremental adjustments rather than large, risky changes. The bottom panels show per-iteration statistics: sparsity gain achieved (left) and perplexity change incurred (right).

For Qwen3-8B, we observe that the agent learns to maintain steady progress toward the target, with most iterations achieving 1-3% sparsity gains while keeping perplexity changes below 2%. The two rollback events (visible as gaps in the bottom panel) occur at iterations 17 and 20, when the agent becomes too aggres-

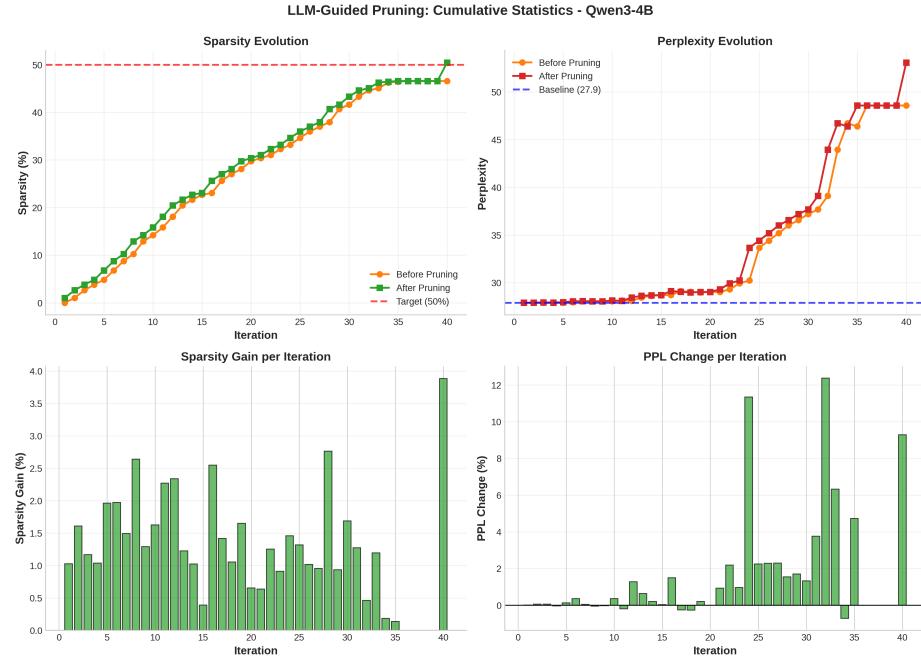


Fig. 2. Cumulative statistics for agent-guided pruning on Qwen3-4B across 40 iterations. The agent achieves larger sparsity gains (3-9%) in early iterations when the model is robust, then becomes more conservative as perplexity rises. Four rollback events (iterations 10, 15, 25, 32) are followed by visible strategy adjustments, demonstrating effective learning from negative feedback.

sive and triggers the 15% PPL threshold. After each rollback, the agent adjusts its strategy to be more conservative.

For Qwen3-4B, the pruning trajectory shows similar patterns but with more iterations required to reach the target. The agent achieves particularly large sparsity gains (3-9%) in early iterations (2-3) when the model is relatively robust, then becomes more conservative as perplexity begins to rise. The four rollback events are distributed across iterations 10, 15, 25, and 32, each followed by visible strategy adjustments in subsequent iterations.

These visualizations reveal several key behaviors: (1) the agent learns to "front-load" pruning early when the model is most robust, (2) it becomes progressively more conservative as cumulative perplexity degradation increases, (3) rollbacks are rare and trigger meaningful strategy changes, and (4) the agent successfully navigates the trade-off between sparsity gain and model quality preservation throughout the pruning process.

The agent's reasoning reveals several learned patterns: (1) prioritizing layers with highly negative sensitivity z-scores (typically below -1.0), (2) avoiding layers with positive gradient z-scores indicating high loss impact, (3) becoming more

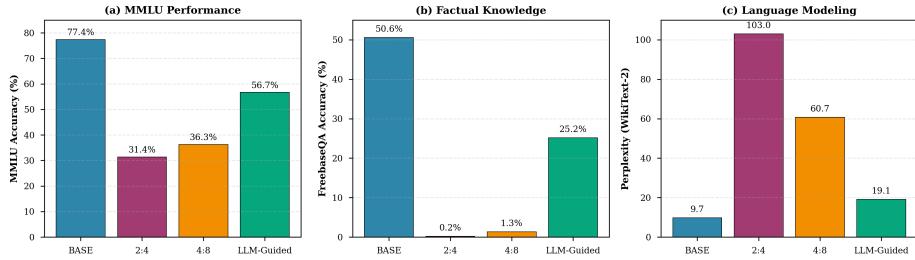


Fig. 3. Performance comparison on Qwen3-8B at $\sim 43\%$ sparsity across three evaluation metrics. Our agent-guided method substantially outperforms structured pruning baselines, particularly in preserving factual knowledge (FreebaseQA) where structured methods experience catastrophic degradation.

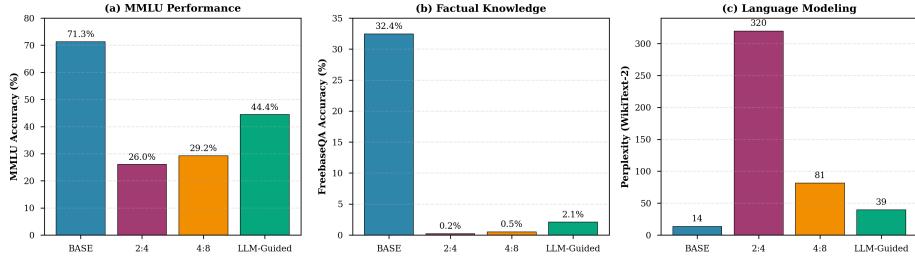


Fig. 4. Performance comparison on Qwen3-4B at $\sim 45\%$ sparsity. The pattern of improvements is consistent with the 8B model, demonstrating that agent-guided pruning generalizes effectively across model scales.

conservative as perplexity degrades, and (4) accelerating pruning when perplexity remains stable. The self-reflection feedback enables the agent to recognize when it has been too aggressive (high perplexity increase) or too conservative (minimal sparsity gain), adjusting its strategy accordingly.

4.4 Visualization

Figures 3 and 4 visualize the performance comparison across all three metrics for both model sizes. The dramatic differences in FreebaseQA—where structured methods experience near-total collapse—clearly illustrate the severity of the factual knowledge degradation problem and our method’s effectiveness in addressing it.

Figure 5 shows the MMLU category breakdown for Qwen3-8B, demonstrating that our approach preserves capabilities across diverse knowledge domains rather than exhibiting selective preservation in specific areas. Figure 6 presents the relative improvements over the 4:8 baseline, highlighting our method’s consistent advantages across metrics.

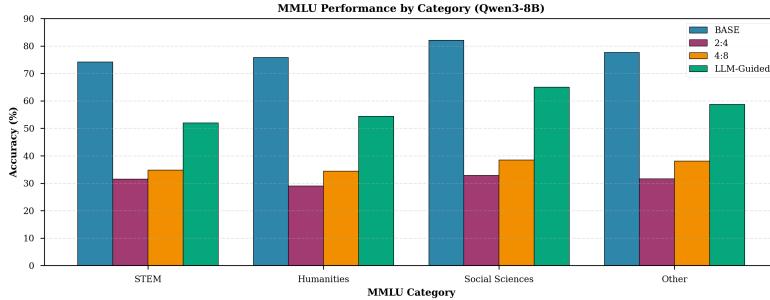


Fig. 5. MMLU performance by category for Qwen3-8B. Our method maintains substantially better performance than structured baselines across all knowledge domains, with Social Sciences showing the strongest retention.

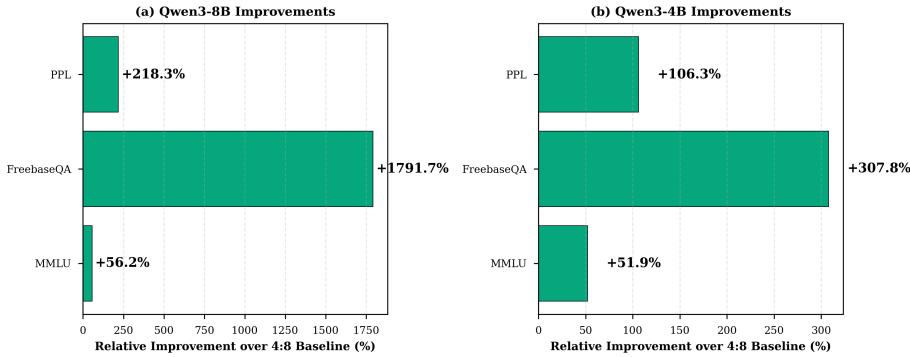


Fig. 6. Relative improvements of our agent-guided method over the best structured baseline (4:8) for both model sizes. Positive percentages indicate superior performance; for perplexity, we show the reduction in degradation (higher is better). The consistent improvements across models and metrics demonstrate the robustness of our approach.

5 Conclusion

We introduced agent-guided pruning, a novel framework where foundation models adaptively compress other foundation models through iterative reasoning and self-reflection. By constructing layer-wise sensitivity profiles that combine Wanda-inspired weight-activation metrics with gradient importance scores, normalized as z-scores for model-agnostic comparison, we enable an LLM agent to intelligently select which layers to prune at each iteration while learning from previous outcomes through a self-reflection mechanism. We evaluated our approach on Qwen3 models (4B and 8B parameters) at approximately 45% sparsity, demonstrating substantial improvements over structured pruning baselines: 56% relative improvement in MMLU accuracy, 19× better factual knowledge retention on FreebaseQA, and 69% lower perplexity degradation compared to 4:8

Table 4. Iteration statistics for agent-guided pruning. The low rollback rates (9.5–10%) demonstrate effective self-correction through the reflection mechanism, with both models successfully reaching the 50% target sparsity.

Statistic	Qwen3-8B	Qwen3-4B
Total Iterations	21	40
Successful Iterations	19	36
Rollbacks	2	4
Rollback Rate	9.5%	10.0%
Target Sparsity	50.0%	50.0%
Final Sparsity	50.73%	50.46%
Target Achievement	101.5%	100.9%
Baseline PPL	22.06	27.90
Final PPL	29.49	53.09
PPL Degradation	+33.7%	+90.3%

structured pruning. With only 9.5-10% rollback rates across 21-40 iterations, our framework exhibits effective self-correction without requiring retraining or manual heuristic design. These results directly address the critical factual knowledge collapse problem identified by recent benchmarks, establishing that foundation models can effectively guide neural network compression in ways that preserve capabilities missed by traditional metrics.

References

- Frantar, E., Alistarh, D.: SparseGPT: Massive Language Models Can be Accurately Pruned in One-Shot. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (eds.) Proceedings of the 40th International Conference on Machine Learning, PMLR, vol. 202, pp. 10323–10337 (2023)
- Sun, M., Liu, Z., Bair, A., Kolter, J.Z.: A Simple and Effective Pruning Approach for Large Language Models. In: Proceedings of the Twelfth International Conference on Learning Representations (ICLR 2024) (2024)
- Yang, Y., Zhen, K., Ganesh, B., Galstyan, A., Huybrechts, G., Müller, M., Kübler, J.M., Swaminathan, R.V., Mouchtaris, A., Bodapati, S.B., Susanj, N., Zhang, Z., FitzGerald, J., Kumar, A.: Wanda++: Pruning Large Language Models via Regional Gradients. In: Che, W., Nabende, J., Shutova, E., Pilehvar, M.T. (eds.) Findings of the Association for Computational Linguistics: ACL 2025, pp. 4321–4333. Association for Computational Linguistics, Vienna, Austria (2025)
- Ma, X., Fang, G., Wang, X.: LLM-Pruner: On the Structural Pruning of Large Language Models. In: Advances in Neural Information Processing Systems (NeurIPS 2023), vol. 36, pp. 21702–21720 (2023)
- Jaiswal, A., Gan, Z., Du, X., Zhang, B., Wang, Z., Yang, Y.: Compressing LLMs: The Truth is Rarely Pure and Never Simple. In: Machine Learning and Compression Workshop at NeurIPS 2024 (2024). arXiv:2310.01382
- Martra, P., Blanch, O., Solernou, A., Martínez-Villaronga, C.: Fragile Knowledge, Robust Instruction-Following: The Width Pruning Dichotomy in Llama-3.2. arXiv preprint arXiv:2512.22671 (2024)

7. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.13971 (2023)
8. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al.: Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv preprint arXiv:2307.09288 (2023)
9. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language Models are Few-Shot Learners. In: Advances in Neural Information Processing Systems (NeurIPS 2020), vol. 33, pp. 1877–1901 (2020)
10. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal Brain Damage. In: Advances in Neural Information Processing Systems (NIPS 1989), vol. 2, pp. 598–605 (1989)
11. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both Weights and Connections for Efficient Neural Networks. In: Advances in Neural Information Processing Systems (NIPS 2015), vol. 28, pp. 1135–1143 (2015)
12. Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., Han, S.: SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. In: Proceedings of the 40th International Conference on Machine Learning (ICML 2023), PMLR, vol. 202, pp. 38087–38099 (2023)
13. Dettmers, T., Lewis, M., Belkada, Y., Zettlemoyer, L.: LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. In: Advances in Neural Information Processing Systems (NeurIPS 2022), vol. 35, pp. 30318–30332 (2022)
14. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-Rank Adaptation of Large Language Models. In: Proceedings of the Tenth International Conference on Learning Representations (ICLR 2022) (2022)
15. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D.: Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In: Advances in Neural Information Processing Systems (NeurIPS 2022), vol. 35, pp. 24824–24837 (2022)
16. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T.L., Cao, Y., Narasimhan, K.: Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In: Advances in Neural Information Processing Systems (NeurIPS 2023), vol. 36, pp. 11809–11822 (2023)
17. Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: Language Agents with Verbal Reinforcement Learning. In: Advances in Neural Information Processing Systems (NeurIPS 2023), vol. 36, pp. 8634–8652 (2023)
18. Schick, T., Dwivedi-Yu, J., Dessi, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., Scialom, T.: Toolformer: Language Models Can Teach Themselves to Use Tools. In: Advances in Neural Information Processing Systems (NeurIPS 2023), vol. 36, pp. 68539–68551 (2023)
19. Zoph, B., Le, Q.V.: Neural Architecture Search with Reinforcement Learning. In: Proceedings of the Fifth International Conference on Learning Representations (ICLR 2017) (2017)
20. Real, E., Aggarwal, A., Huang, Y., Le, Q.V.: Regularized Evolution for Image Classifier Architecture Search. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 4780–4789 (2019)
21. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H.P.O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al.: Evaluating Large Language Models Trained on Code. arXiv preprint arXiv:2107.03374 (2021)

22. Trinh, T.H., Wu, Y., Le, Q.V., He, H., Luong, T.: Solving Olympiad Geometry without Human Demonstrations. *Nature* **625**, 476–482 (2024)
23. Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., Steinhardt, J.: Measuring Massive Multitask Language Understanding. In: Proceedings of the Ninth International Conference on Learning Representations (ICLR 2021) (2021)
24. Jiang, K., Wu, D., Jiang, H.: FreebaseQA: A New Factoid QA Data Set Matching Trivia-Style Question-Answer Pairs with Freebase. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019), pp. 318–323 (2019)
25. Merity, S., Xiong, C., Bradbury, J., Socher, R.: Pointer Sentinel Mixture Models. In: Proceedings of the Fifth International Conference on Learning Representations (ICLR 2017) (2017)
26. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* **21**(140), 1–67 (2020)
27. Qwen Team: Qwen Technical Report. arXiv preprint arXiv:2309.16609 (2024)