DART-ing Through the Drift: Dynamic Tracing of Knowledge Neurons for Adaptive Inference-Time Pruning

Anonymous ACL submission

Abstract

Large Language Models (LLMs) rely on sparsification to reduce inference latency and memory usage by eliminating redundant parameters. However, existing pruning methods suffer from three key limitations. First, they rely on calibration datasets, which introduce domain bias and limit generalization. Second, they allocate sparsity using layer-local activation statistics, without considering how FFN layers jointly shape the residual stream and token embeddings across depth. Third, they depend on static masks that fail to adapt to the evolving semantic state during autoregressive generation, a failure mode we term knowledge drift. As generation progresses, evolving attention patterns induce knowledge drift, causing fixed allocations to become suboptimal and degrading output quality. We introduce DART, a training-free framework for online inference-time sparsification using only forward-pass activations, requiring no calibration data or auxiliary predictors. DART dynamically refreshes stale masks during autoregressive decoding and allocates sparsity non-uniformly across layers based on their embedding transformation sensitivity. Across 11 benchmarks and 10 models from the LLaMA-2, LLaMA-3, DeepSeek-R1, Qwen3 and Mistral families, DART retains over 92% of dense-model accuracy at 50% Feed Forward Network (FFN) sparsity. On our custom multi-domain prompts, which force cross-domain transitions that no existing benchmark exercises, DART retains 83%, improving over the static baselines by 60%, while being upto $3.54\times$ end-to-end speedup without specialized kernels.

1 Introduction

LLMs have demonstrated strong capabilities across diverse applications, including code generation (Chen, 2021; DeepSeek-AI, 2025), mathematical reasoning (Wei et al., 2022; Qwen Team, 2025), biomedical diagnostics (Singhal et al., 2023; Li et al., 2025), and embodied robotics (Driess et al., 2023). These capabilities are largely driven by scale (Kaplan et al., 2020; Hoffmann et al., 2022), producing models with massive memory and computation overheads (Chowdhery et al., 2023) and motivating sparsification techniques that suppress inactive FFN neurons to sustain high-throughput, low-latency deployment under dynamic workloads (Yu et al., 2022; Wan et al., 2023).

Background: Existing pruning technique, implicitly or explicitly, selects an active neuron subspace, the subset of FFN parameters permitted to participate in computation. SparseGPT (Frantar and Alistarh, 2023) and Wanda (Sun et al., 2024) select this subspace once from a calibration set and fix it globally. DLP (Chen et al., 2025) and OWL (Yin et al., 2024) refine it per layer using calibration-derived importance scores, but the subspace remains static at inference. DejaVu (Liu et al., 2023) selects a per-token subspace via auxiliary predictors, but it too requires dataset-specific training. Griffin (Dong et al., 2024) commits to the prefill-selected subspace for the entire generation and does not account for differing layer sensitivities to pruning. Despite their differences, all these methods share a common assumption that the active subspace selected at or before generation onset remains valid throughout.

Motivation: This assumption breaks down under autoregressive generation. (Geva et al., 2021; Dai et al., 2022; Meng et al., 2022) show that different knowledge domains activate distinct neuron subspaces. We observe that in realistic generation settings, a single prompt often traverses multiple domains within the same response. For example, a query such as "Tell me things to do in Paris" may invoke neurons associated with restaurants, museums, history, sports, and transportation during decoding. Consequently, the active neuron subspace evolves throughout generation rather than remaining fixed, where attention continuously shifts across seman-

tic regions of the context, even under aggressive domain-specific pruning. We term this mismatch *knowledge drift*. The evolving attention state moves beyond the representational support of the fixed active subspace and the FFN can no longer express the knowledge the model is attempting to retrieve. As illustrated in Figure 2b, this drift is not silent. When a domain transition occurs, attention outputs change direction measurably before generation quality degrades, providing a detectable early warning signal.

Effective subspace selection is further complicated by layer heterogeneity. We observe that middle FFN layers exhibit the highest semantic specialization and cross-domain activation variance, making them the primary source of knowledge drift, while late layers contain a small set of domain-invariant neurons and early layers remain highly sensitive to perturbations. Consequently, uniform sparsity budgets and fixed subspace selections fail to adapt to evolving generation dynamics.

Our Contributions: We introduce **Dart** (Dynamic Attention-Guided Runtime Tracing), a training-free, calibration-free framework that formulates FFN sparsification as an online subspace tracking problem. Dart consists of two components: (i) a *Knowledge Pruner* that performs input-conditioned subspace selection with non-uniform layer-wise sparsity allocation, and (ii) a *Knowledge Tracer* that monitors attention trajectories to detect subspace drift and trigger mask refreshes before generation degrades. Our contributions are:

❶ We identify *knowledge drift*, a previously uncharacterized failure mode where the active FFN subspace becomes misaligned with the model's evolving generation target, causing degenerate outputs in long-horizon tasks.

❷ We present Dart, a training-free framework for online FFN subspace tracking. It dynamically reallocates sparsity across layers and refreshes stale masks using attention-based drift detection.

❸ We introduce a template-based multi-domain benchmark in which structured prompt templates with typed subject slots are instantiated across curated domain taxonomies, requiring explicit knowledge-domain transitions within a single generation, exposing failure modes overlooked by existing single-domain benchmarks.

Across 11 benchmarks spanning zero-shot commonsense reasoning, 5-shot domain-specific evaluation, and language modeling, we evaluate Dart on 10 LLMs from the LLaMA-2, LLaMA-3, DeepSeek-R1, Qwen3, and Mistral families. At 50% FFN sparsity, Dart consistently outperforms both static and dynamic pruning baselines, retaining over 92% of dense-model accuracy on LLaMA-2/3 models. Across 10 summarization benchmarks, Dart consistently outperforms static pruning on every dataset; on open-ended generation, it retains up to 93% of dense quality versus 76% for static pruning. On our multi-domain benchmark, static pruning degenerates in 98% of samples while Dart retains 83% of dense quality, delivering up to $3.54\times$ end-to-end speedup without specialized kernels.

2 Preliminaries

We consider a transformer-based LLMs with L layers, where each layer consists of Multi-Head Attention (MHA) and a gated FFN (Shazeer, 2020) with residual connections. Modern LLMs employ Grouped-Query Attention (GQA) (Ainslie et al., 2023), where multiple query heads share key and value projections, which reduces attention parameter count relative to standard MHA and concentrates the computational cost in the FFN. In LLaMA-3.1-8B (Grattafiori et al., 2024), FFN layers account for $\sim$5.6B of the 6.9B non-embedding parameters (81%), versus $\sim$1.3B for attention (19%). In Mixture of Experts(MoE) architectures such as Mixtral 8×7B (Jiang et al., 2024), the expert FFNs account for 96% of the total paramters, while attention contributes under 4% making FFN pruning the dominant lever for efficiency in both dense and sparse LLMs.

Feed-Forward Network. Given $\mathbf{y}_t^{(l)} \in \mathbb{R}^d$ as input at layer l and time step t, the gated FFN computes an intermediate representation via up- and gate-projection matrices $\mathbf{W}_{\text{up}}^{(l)}, \mathbf{W}_{\text{gate}}^{(l)} \in \mathbb{R}^{d\times m}$:

$$\mathbf{u}_t^{(l)} = \mathbf{y}_t^{(l)}\mathbf{W}_{\text{up}}^{(l)} \odot \text{SiLU}\left(\mathbf{y}_t^{(l)}\mathbf{W}_{\text{gate}}^{(l)}\right), \quad (1)$$

followed by a down-projection $\mathbf{W}_{\text{down}}^{(l)} \in \mathbb{R}^{m\times d}$:

$$\mathbf{z}_t^{(l)} = \mathbf{u}_t^{(l)}\mathbf{W}_{\text{down}}^{(l)}. \quad (2)$$

Given a binary mask $M^{(l)} \in \{0,1\}^m$, the pruned FFN selectively zeroes neurons before the down-projection:

$$\hat{\mathbf{z}}_t^{(l)} = \left(M^{(l)} \odot \mathbf{u}_t^{(l)}\right)\mathbf{W}_{\text{down}}^{(l)}, \quad (3)$$

where $\odot$ denotes the Hadamard product.

Knowledge neurons and contextual sparsity. The FFN operates as a key-value memory (Geva et al., 2021), where each neuron in $\mathbf{u}_t^{(l)}$ acts as a key that retrieves domain-specific knowledge stored as values in $\mathbf{W}_{\text{down}}^{(l)}$. Given token embeddings within a knowledge domain, only domain-relevant neurons activate strongly (Meng et al., 2022), making FFN activations inherently contextually sparse. In contrast, GQA forces query heads to operate within a shared key-value subspace, reducing the intrinsic head-level sparsity of MHA (Liu et al., 2023). Together, these properties make the FFN the primary target for runtime pruning.

3 Probing Knowledge Dynamics in LLMs

We present two structural observations about LLM generation dynamics that jointly motivate DART's design: the heterogeneous functional roles of layers across the network depth, and the decoupling of attention-driven semantic navigation from FFN-based knowledge realization under pruning.

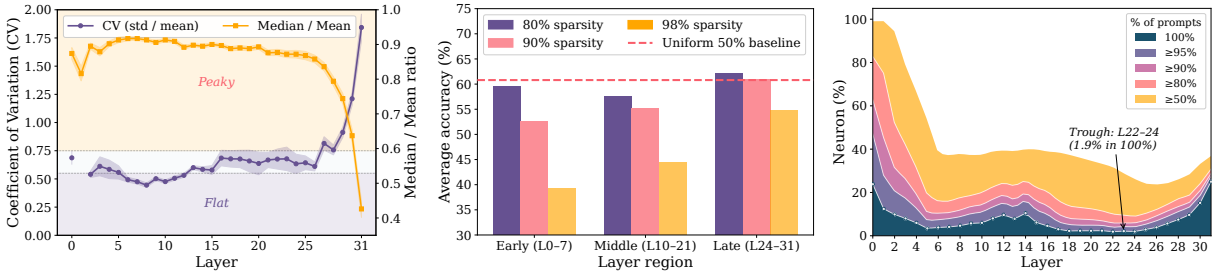
Layer-Wise Functional Specialization. Prior work has shown that different layers in LLMs serve distinct functional roles (Geva et al., 2021; Dai et al., 2022; Meng et al., 2022). As shown in Figure 1a, early layers exhibit relatively flat activation distributions, indicating broad neuron utilization across domains, whereas later layers show highly peaked activations dominated by a sparse subset of specialized neurons. To evaluate their functional importance, we progressively prune low-activation neurons at different depths. Figure 1b shows that pruning early layers causes significant accuracy degradation even at modest sparsity, while later layers tolerate aggressive pruning with minimal impact. Since perturbations in early layers propagate downstream, these layers are particularly sensitive. We also observe that every layer requires a minimum retained neuron capacity below which accuracy drops sharply. Middle and later layers exhibit different sparsification behavior. Late layers rely on a few dominant high-contribution neurons, enabling aggressive pruning once these neurons are preserved. In contrast, middle layers contain neurons with relatively similar activation strengths yet still sustain high sparsity. As shown in Figure 1c, this is because middle layers are highly domain-specialized and share substantially fewer neurons across domains, making only a small subset relevant for a given input. Finally, the cosine similarity between a layer's input and output embeddings closely tracks these behaviors as seen in Figure 2a. Early layers induce larger embedding transformations and are correspondingly more sensitive to pruning, whereas middle layers produce smaller embedding changes consistent with their domain-specialized behavior. A detailed study of the per layer importance exists in Appendix D. We formalize this intuition as a runtime sensitivity signal in Section 4.1.

Semantic Navigation Persists Under Pruning Since FFN layers encode domain-specific knowledge (Geva et al., 2021), pruning them with respect to a given domain would be expected to confine generation to that domain. Contrary to this expectation, we observe that even under aggressive domain-specific FFN pruning, attention consistently shifts toward tokens associated with secondary semantic domains implied by the context, even when such tokens constitute less than 5% of the context as seen in Figure 2c. The model repeatedly attempts to transition into these domains despite insufficient FFN capacity to support them. Failed transitions redirect generation back to the original domain, followed by subsequent re-attempts. This oscillatory behaviour intensifies over time, producing repetitive or semantically inconsistent outputs. Detailed examples exist in Appendix E.

To quantify this effect, we track cosine similarity between n-gram representations of attention outputs and a reference vector from the mask construction phase. As shown in Figure 2b, the unpruned model exhibits a sharp similarity drop at tokens 425-475, indicating a domain shift, while the pruned model maintains high similarity (> 0.6), confirming failed transitions despite attention attempting to shift context. This creates a fundamental tension: attention continues to drive semantic navigation across domains, while FFN pruning restricts the capacity to realize it. Effective pruning must therefore dynamically adapt FFN neurons to match the evolving semantic path - motivating the Knowledge Tracer introduced in Section 4.3.

4 DART

We first formalize the *Knowledge Pruner*, a lightweight, calibration-free mechanism that selects a per-layer FFN subspace conditioned on the current context, motivated by the observations in Section 3.1. We then introduce the *Knowledge Tracer*, motivated by the drift observations in Section 3.2, which monitors model dynamics and triggers a subspace update when either detector



(a) (b) (c)

Figure 1: (a) Layer-wise neuron activation statistics showing the coefficient of variation (CV) and Median/Mean ratio of neuron activations across layers. (b) Accuracy sensitivity to layer-wise sparsity when the least-activated neurons are progressively pruned based on activation magnitude. (c) Domain-invariant neurons across layers, where the y-axis shows the percentage of neurons appearing in masks for a different percentage of prompts (legend) across 100 multi-topic prompts.

confirms a domain shift. The Knowledge Tracer comprises a proactive detector and a reactive detector, monitoring attention outputs and generated tokens respectively, as shown in Figure 3a. The overall state machine is illustrated in Figure 3b.

4.1 Context-based Pruning of FFN

Only a subset of FFN neurons is active for any given semantic context (Dong et al., 2024). We capture temporal neuron significance s_i, for the i-th neuron by accumulating activation magnitudes over a window of τ tokens:

$$s_i = \sum_{t=1}^{\tau} \left(u_{t,i}^{(l)}\right)^2 .$$

A binary mask $M^{(l)}$ retains the top-k neurons by cumulative score, $M_i^{(l)} = \mathbb{I}[|\{s_j \mid s_j > s_i\}| < k]$, (where $\mathbb{I}$ is the indicator function) removing entire columns from $W_{\text{up}}^{(l)}$, $W_{\text{gate}}^{(l)}$ and entire rows from $W_{\text{down}}^{(l)}$ for all $M_i^{(l)} = 0$.

4.2 Layer-Wise Contextual Sparsity

With intra-layer pruning defined, we allocate inter-layer sparsity via two factors derived from the observations in section 3.

Layer Sensitivity. We hypothesize that a layer is functionally important if its output significantly alters the direction or magnitude of the input state. Thus, we quantify the functional impact of FFN$^{(l)}$ on the t-th token by a sensitivity score combining directional and magnitude change:

$$S_t^{(l)} = \left(1 - \frac{\mathbf{y}_t^{(l)} \mathbf{z}_t^{(l)}}{\|\mathbf{y}_t^{(l)}\| \cdot \|\mathbf{z}_t^{(l)}\|}\right) \cdot \frac{\|\mathbf{z}_t^{(l)} - \mathbf{y}_t^{(l)}\|}{\|\mathbf{y}_t^{(l)}\|}, \tag{4}$$

where the first term measures the change in direction (via cosine similarity), while the second term measures the intensity of the update relative to the input. Averaged over τ tokens and normalized across layers, the relative importance score is

$$I_{t,\tau}^{(l)} = 1 - \frac{\bar{S}_{t,\tau}^{(l)}}{\sum_{j=0}^{L-1} \bar{S}_{t,\tau}^{(j)}}, \quad \bar{S}_{t,\tau}^{(l)} = \frac{1}{\tau} \sum_{i=0}^{\tau-1} S_{t+i}^{(l)}.$$

Depth-Aware Sparsification. Early layers are functionally denser and less robust to pruning (Gromov et al., 2024; Men et al., 2025), as confirmed by sensitivity results in appendix D.1. For normalized depth $\tilde{l} = l/(L-1)$, we apply a depth-dependent factor:

$$D^{(l)} = \min\Big(1,\ \alpha_e + (1-\alpha_e)\frac{\tilde{l}}{\beta_e},\ \alpha_l + (1-\alpha_l)\frac{1-\tilde{l}}{\beta_l}\Big), \tag{5}$$

where $\alpha_e, \alpha_l \in (0,1)$ scale early/late protection and $\beta_e + \beta_l \leq 1$.

Iterative Redistribution. Given global sparsity ρ, we solve $p^{(l)} \in [p_{\min}, p_{\max}]$ iteratively: at step k, each active layer absorbs a share of the remaining budget Δ_{k-1} proportional to $I^{(l)} \cdot D^{(l)}$,

$$p_k^{(l)} = \text{clip}\left(p_{k-1}^{(l)} + \frac{\Delta_{k-1} \cdot I_{t,\tau}^{(l)} \cdot D^{(l)}}{\sum_{j \in A_{k-1}} I_{t,\tau}^{(j)} \cdot D^{(j)}},\ p_{\min}, p_{\max}\right),$$

$$\Delta_k = \Delta_{k-1} - \sum_{j \in A_{k-1}} \left(p_k^{(j)} - p_{k-1}^{(j)}\right),$$

$$A_k = \left\{j \mid p_k^{(j)} \in (p_{\min}, p_{\max})\right\}.$$

Detailed ablations exist in Appendix D.2 for reference.

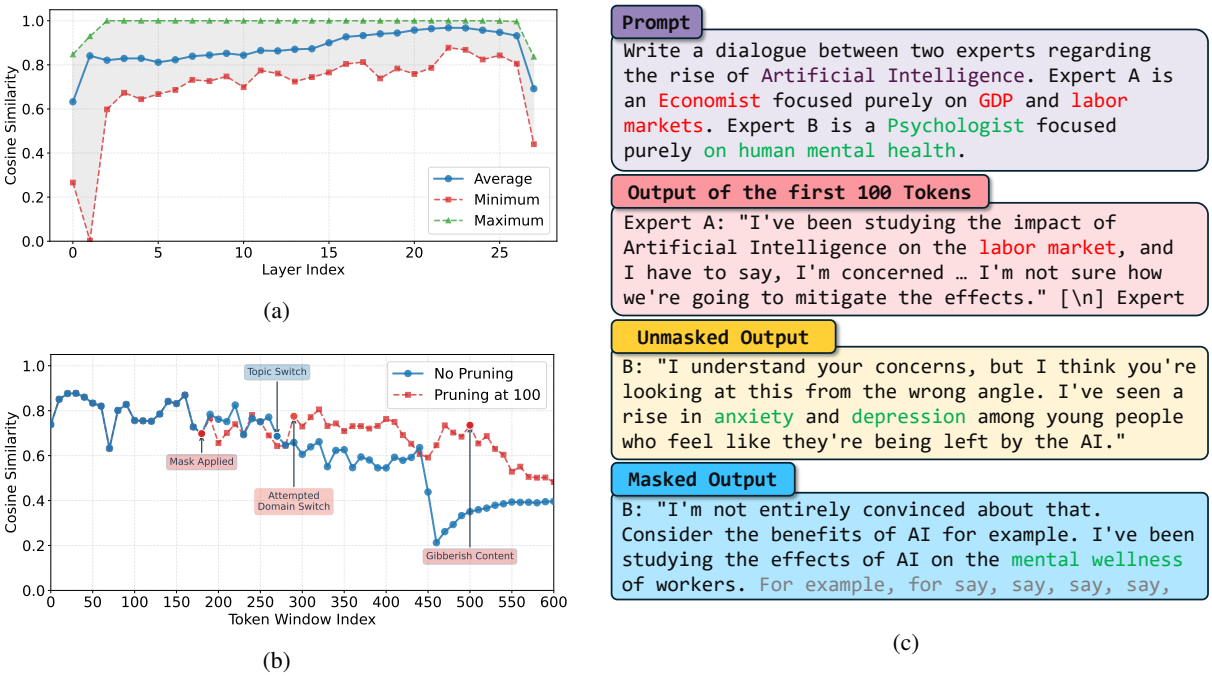


Figure 2: (a) Layer-wise cosine similarity between FFN input and output embeddings, showing that layers with larger embedding transformations are more sensitive to pruning. (b) Cosine similarity between attention outputs during generation and the masking reference representation. The pruned model fails to transition into new semantic domains despite attention attempting to shift context. (c) Example generations from unpruned and statically pruned models, where failed semantic transitions under pruning lead to repetitive outputs.

4.3 Knowledge Drift

Having derived context-conditioned masks, we address their limitation under autoregressive generation. DART employs two complementary detectors, a proactive monitor on attention geometry and a reactive fallback on generated tokens, triggering re-masking only when drift is confirmed.

Proactive Drift Detection. We define knowledge drift as a deviation in the semantic trajectory of the final-layer attention outputs. Let $\mathbf{y}_t^{(L-1)} \in \mathbb{R}^d$ denote the attention output of the last layer, $L-1$ at token t. Over a window of τ_w tokens, we compute the centroid

$$\bar{\mathbf{y}}_{t,\tau_w} = \frac{1}{\tau_w} \sum_{i=0}^{\tau_w - 1} \mathbf{y}_{t+i}^{(L-1)}.$$

Setting τ_w too small yields noisy centroids as the standard error falls as $O(1/\sqrt{\tau_w})$ while too large and drift onset mid-window goes undetected. We use $\tau_w = 20$ based on the ablation in appendix F. Let t_0 denote the start of the scoring phase, T its duration, and $K = \lfloor T/\tau_w \rfloor$ the number of reference windows. During mask construction, we compute cosine similarities between each reference window and the full scoring-phase centroid as $a_i = \cos(\bar{\mathbf{y}}_{t_0+i\tau_w,\tau_w}, \bar{\mathbf{y}}_{t_0,T})$, with $\mu_{t_0,T} = \frac{1}{K}\sum_{i=0}^{K-1} a_i$ and $\sigma^2_{t_0,T} = \frac{1}{K}\sum_{i=0}^{K-1}(a_i - \mu_{t_0,T})^2$. Drift is detected when $a_{t',\tau_w} = \cos(\bar{\mathbf{y}}_{t',\tau_w}, \bar{\mathbf{y}}_{t_0,T}) < a_{\min}$. Further ablations motivating this parameter-free minimum-alignment criterion are provided in appendix F.

To avoid re-masking on transient deviations, we accumulate drift evidence across consecutive windows using a counter:

$$c_t = \begin{cases} c_{t-\tau_w} + 1 & \text{if drift is detected,} \\ \max(0, c_{t-\tau_w} - 1) & \text{otherwise.} \end{cases}$$

Re-masking is triggered when $c_t \geq c_0$. Since smaller τ_w produces noisier centroid estimates, we maintain a fixed confirmation horizon $\tau_w^* = 20$ tokens by setting $c_0 = \left\lceil \frac{\tau_w^*}{\tau_w} \right\rceil$. Upon triggering, the model reverts to full unpruned computation for τ_r tokens to re-score neurons and build a fresh mask. Approximating centroid variance as inversely proportional to the number of observed windows gives $\tau_r \geq C(\sigma_{t_0,T}/\mu_{t_0,T})^2$, where $C = \tau_w/\varepsilon^2$ with $\varepsilon = 0.1$:

$$\tau_r = \text{clip}\left(\left\lfloor C \left(\frac{\sigma_{t_0,T}}{\mu_{t_0,T}} \right)^2 \right\rfloor, \tau_r^{\min}, \tau_r^{\max} \right), \quad (6)$$

$$C = \frac{\tau_w}{\varepsilon^2}.$$

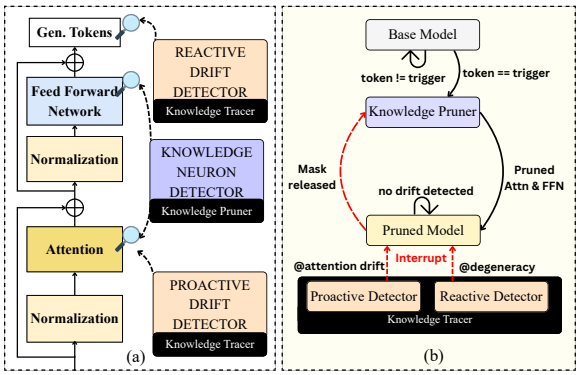


Figure 3: (a) Knowledge Pruner and Knowledge Tracer within a transformer layer. (b) The runtime state machine where at a pre-defined generation step τ(which acts as the trigger point), the Pruner selects an initial FFN subspace conditioned on the current context. The system then continues generating until either detector confirms a shift, at which point the model enters a re-pruning state where the Pruner recomputes a fresh subspace aligned with the evolved context.

Algorithm 1 Reactive Drift Detection

Require: Generated tokens $\mathbf{g}$, window size W
Require: N-gram sizes $\{3, 5, 8, 12\}$
Require: Thresholds $\{4, 3, 2, 2\}$
for each $n \in \{3, 5, 8, 12\}$ **do**
  Extract distinct n-grams $\mathcal{V}_n$ from $\mathbf{g}_{T-W+1:T}$
  Compute

$$R_n = \max_{s \in \mathcal{V}_n} \text{count}(s, \mathbf{g}_{T-W+1:T})$$

  if $R_n \geq \theta_n$ **then**
    return Drift Detected
  end if
end for
return No Drift

We set $\tau_r^{\min} = 20$ and $\tau_r^{\max} = 50$. The reactive drift detector serves as a fallback mechanism when attention geometry remains stable despite degenerate generation behavior, and is summarized in Algorithm 1. Detailed ablations for the Proactive and Reactive detectors exist in Appendix F and Appendix G respectively.

5 Experiments

Accuracy across datasets. We evaluate DART on LLAMA-3-8B and LLAMA-3.1-8B (Grattafiori et al., 2024). Results on other models, *i.e.*, LLAMA-2-8B/13B (Touvron et al., 2023), QWEN3-4B/14B (Yang et al., 2025), DEEPSEEK-R1-QWEN/LLAMA-8B (Guo et al., 2025), and MISTRAL-7B (Jiang et al., 2023), are delayed to the Appendix A. We evaluate our approach on language modeling benchmarks including WIKITEXT (Merity et al., 2016) and eight zero-shot downstream tasks, BOOLQ (Clark et al., 2019), RTE (Wang et al., 2018), HELLASWAG (Zellers et al., 2019), WINOGRANDE (Sakaguchi et al., 2020), ARC-E/C (Clark et al., 2018), PIQA (Bisk et al., 2020) and OBQA (Mihaylov et al., 2018), which together assess general reasoning, entailment, and commonsense robustness. To evaluate prompt-conditioned knowledge preservation, we additionally benchmark on multi-shot domain-specific datasets MMLU (Hendrycks et al., 2021), GPQA (Rein et al., 2024), and MEDMCQA (Pal et al., 2022), which carry domain-specific factual reasoning tasks. Zero-shot and five-shot evaluations are conducted using LM-EVAL-HARNESS (Gao et al., 2024).

We compare DART against a range of state-of-the-art pruning methods. Across both MLP-only and full-model pruning settings (Tables 1 and 2), DART consistently achieves superior accuracy. Additionally, DART yields significantly lower perplexity compared to prior approaches, which we attribute to its continuous drift detection mechanism that dynamically updates neuron selection during generation. We refer the reader to Appendix B for a comparison with state-of-the-art static pruning techniques. For the impact of multi-batching, we refer the reader to Appendix I.

General-Domain Summarization and Generation. We evaluate DART at 50% FFN sparsity on LLaMA-3.1-8B-Instruct across summarization on 10 benchmarks(full table in Appendix J), open-ended generation (AlpacaEval, WritingPrompts, OpenWebText). We report ROUGE-1/2/L and BLEU-4, alongside an LLM-as-judge score on five criteria scored [0,1] (Appendix H). The Knowledge Tracer improves quality across all 13 comparisons. On summarization, drift detection raises the mean judge score from 0.47 to 0.52, with the largest gain on DialogSum (+0.08), where topic shifts cause the active FFN subspace to diverge mid-document. On AlpacaEval, drift detection retains 93% of dense quality (judge 0.84) against 76% under a static subspace (judge 0.68). The largest gains appear in tasks requiring sustained semantic adaptation over long generations, where the initially selected sparse subspace progressively loses alignment with the evolving context.

Knowledge Drift Stress Test. To directly stress temporal non-stationarity in neuron relevance, we construct chained multi-topic prompts containing

Table 1: Comparison of pruning methods at 50% FFN-only sparsity on commonsense reasoning benchmarks. (*) results are extracted from CFSP (Wang et al., 2024).

Method	Sparsity	WinoGrande	PIQA	OBQA	HellaSwag	ARC-e	ARC-c	Mean
Llama-3.1-8B*	0%	74.0	81.0	44.8	78.9	80.9	53.3	68.8
Wanda-SP*	50%	61.9	70.8	36.8	59.6	61.5	36.4	54.5
FLAP*	50%	58.1	60.0	31.0	42.0	39.6	26.0	42.8
CSFP*	50%	65.2	71.2	36.4	61.2	62.5	37.3	55.6
SparseLLM	50%	70.0	74.5	39.5	65.5	63.5	37.0	58.3
DART (ours)	50%	**70.5**	**75.5**	**44.5**	**66.5**	**74.5**	**49.5**	**63.5**

Table 2: Comparison of pruning methods at 40% FFN and Attention sparsity across language modeling and reasoning benchmarks. We prune individual weights instead of neurons for Query, Key and Value matrics as done in WiNA(Chen et al., 2026). (*) results are extracted from Probe Pruning (Le et al., 2025).

Method	Sparsity	WikiText (PPL)	BoolQ	PIQA	WinoGrande	ARC-c	OBQA	Mean
Llama-3-8B*	0%	6.8	85.0	79.5	72.5	47.2	40.2	54.5
Full-Batch Probing*	40%	12.3	73.1	**77.8**	70.3	42.9	**39.8**	**52.7**
Wanda-SP*	40%	18.4	66.6	73.4	63.2	31.8	34.5	48.0
FLAP*	40%	18.5	67.3	73.5	66.7	31.7	34.4	48.7
PP*	40%	14.9	70.3	76.3	67.2	39.0	36.9	50.8
DART (ours)	40%	**11.9**	**79.4**	76.8	**71.3**	**44.0**	32.6	**52.7**

Table 3: End-to-end speedup of Dart under different generation regimes on Llama-3.1-8B. Values are reported as *MLP-only / MLP+Attn* at 50% sparsity (p = *prefill tokens,* d = *decode tokens*).

Workload	Speed-Up
Document ingestion (5000p)	3.38× / 3.54×
Summarization (5000p + 100d)	1.75× / 1.81×
Reasoning (100p + 1000d)	1.30× / 1.31×
Short QnA (2p + 500d)	1.31× / 1.26×

2–4 semantic domain transitions within a single generation trajectory(Appendix K) As shown in Figure 4, static pruning rapidly degrades as topic complexity increases, exhibiting severe repetition and loss of transition quality under cross-domain generation. It enter degeneracy in 98% of the prompts under all three prompt types. Periodic remasking partially mitigates these failures for simpler transitions, but its advantage over static pruning diminishes as the number of topic shifts increases, eventually approaching static behavior under 4-topic chains. This occurs because fixed-interval refreshes become increasingly unlikely to align with the actual semantic transition boundaries within generation. In contrast, drift-aware remasking remains consistently stable across all topic counts, maintaining an overall judge score of approximately 0.64–0.65 while preserving substantially stronger non-repetition behavior than both periodic and static baselines. Although periodic remasking reduces catastrophic degeneration relative to static pruning, it still frequently misses stale-mask onset, leading to repeated looping failures during longer semantic trajectories. Despite operating at lower sparse execution efficiency, drift-aware remasking delivers significantly higher quality per unit of pruned computation than periodic remasking, demonstrating that *when* remasking occurs is substantially more important than *how often* remasking occurs. Overall, Dart's quality reaches 83% of the dense with 61% efficiency, and re-pruning overhead concentrates only where attention direction genuinely shifts across knowledge domains.

Inference Speed-Ups. As shown in Table 3, Dart delivers consistent end-to-end speedups across diverse generation regimes, with the largest gains in compute-bound prefill workloads. For long document ingestion (5000p), Dart achieves up to $3.54\times$ speedup, while summarization workloads reach up to $1.81\times$ acceleration. The gains arise from reducing computation in structured FFN and attention projections. However, overall latency remains partially constrained by non-prunable components such as KV-cache operations, attention softmax, sampling, and runtime overheads, which become more dominant during long autoregressive decoding. Consequently, decode-heavy workloads

Table 4: Automated and LLM-as-a-Judge evaluation across summarization and generation benchmarks. Efficiency denotes the percentage of generated tokens executed under the pruned model configuration relative to the total generated tokens.

Configuration		Automated Metrics			LLM-as-Judge						Efficiency
Dataset	Mode	R1	RL	BLEU	Cov	Coh	Spec	Trans	NRep	Overall	Eff.%
CNN_DAILYMAIL	Dart	.183	.134	.024	.75	.85	.80	.75	.90	.81	85.4
	static	.167	.121	.022	.70	.80	.75	.70	.80	.75	100.0
XSUM	Dart	.158	.111	.006	.65	.85	.70	.65	.85	.74	85.2
	static	.138	.103	.005	.60	.80	.65	.60	.80	.69	100.0
SAMSUM	Dart	.247	.186	.013	.75	.90	.50	.60	.85	.72	89.9
	static	.241	.189	.021	.70	.85	.45	.55	.80	.67	100.0
DIALOGSUM	Dart	.196	.152	.008	.80	.90	.60	.65	.85	.76	95.4
	static	.183	.136	.009	.70	.85	.50	.60	.75	.68	100.0
GOV_REPORT	Dart	.073	.076	.004	.15	.55	.45	.30	.60	.41	97.0
	static	.059	.066	.002	.10	.40	.40	.25	.45	.32	100.0
ALPACA_EVAL	dense	—	—	—	.90	.95	.85	.85	.95	.90	NA
	Dart	.849	.830	.763	.85	.90	.80	.80	.85	.84	79.3
	static	.444	.388	.228	.70	.75	.60	.65	.70	.68	100.0
WRITINGPROMPTS	dense	—	—	—	.40	.45	.35	.30	.40	.38	NA
	Dart	.887	.875	.850	.35	.40	.30	.25	.35	.33	38.4
	static	.652	.629	.550	.30	.35	.25	.20	.25	.27	100.0
OPENWEBTEXT	dense	—	—	—	.85	.90	.80	.80	.85	.84	NA
	Dart	.964	.963	.910	.80	.85	.75	.75	.80	.79	88.8
	static	.642	.617	.455	.75	.80	.65	.70	.70	.72	100.0

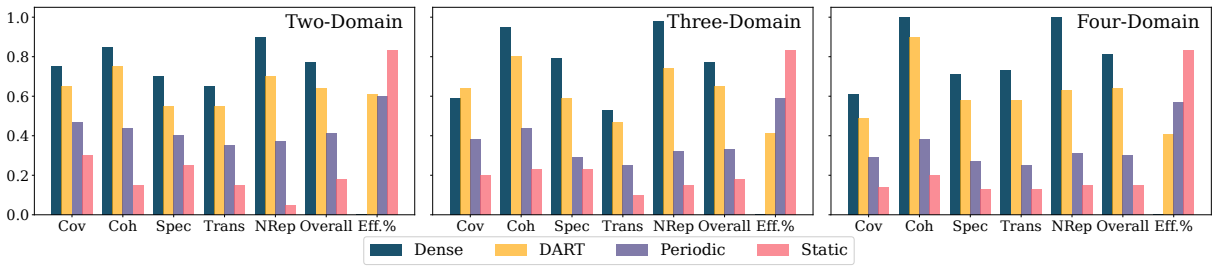


Figure 4: Coverage, coherence, specificity, transition quality, and non-repetition of generated text as evaluated by an LLM-as-a-Judge on chained prompts containing 2-, 3-, and 4-domain semantic transitions within a 500-token generation window. The periodic approach refreshes the active neuron set every 100-tokens.

such as reasoning and short QnA observe smaller but consistent gains of $1.25\times$–$1.31\times$.

6 Conclusion

Sparsity in LLMs is fundamentally limited by the non-stationary nature of neuron relevance during autoregressive generation. We show that a fixed sparse subspace, even when calibrated from datasets statically or derived from semantically aligned prompts, progressively loses alignment as the generated content transitions across knowledge domains. To address this, we introduce Dart, a drift-aware structured pruning framework that dynamically re-estimates sparse neuron subspaces only when semantic drift is detected. Dart performs content-aware remasking conditioned on changes in attention dynamics, enabling the active sparse subspace to track evolving knowledge requirements throughout generation. Across reasoning, summarization, and open-ended generation benchmarks, Dart consistently improves robustness and output quality while maintaining high effective sparsity and accelerator-friendly dense matrix execution. Static mask approaches degenerate in 98% of our tested multi-domain prompts, and only retain 23% of the model quality which is far lesser as compared to over 83% quality retention as compared to the dense baselines by Dart. These results suggest that adaptive sparsity mechanisms will be critical for deploying highly sparse LLMs in long-context and multi-domain generation settings.

Limitations

LLMs themselves exhibit highly heuristic internal behavior, shaped by shared architectural priors, autoregressive training objectives, and broad internet-scale datasets. Consequently, no single sparsification or drift-detection policy is expected to be universally optimal across all prompts, domains, or model families. Rather than assuming exact neuron-level consistency, DART is designed around higher-level empirical observations that remain stable across modern decoder-only transformers: neuron relevance evolves during generation, semantic transitions induce subspace misalignment, and stale sparse masks lead to degeneration and repetition. While the specific drift statistics and optimal thresholds may vary slightly across architectures and workloads, our results demonstrate that the underlying behavior generalizes consistently across multiple LLaMA, QWEN, DEEPSEEK, and MISTRAL models. MoE architectures already perform dynamic content-aware routing through expert selection, but DART could potentially further identify high-value neurons within each expert FFN. However, optimizing such sparsity is challenging because individual experts are already relatively small, leaving limited redundancy for stable structured pruning and remasking. Additionally, the current implementation focuses on FFN sparsity during inference and does not dynamically sparsify KV-cache representations, or training-time execution. Extending drift-aware sparse execution to hardware-native sparse accelerators, multimodal transformers, and adaptive training-time sparsification remains future work.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback and suggestions. We also acknowledge the open-source community for providing model implementations, evaluation frameworks, and datasets used throughout this work, including the developers of the LLaMA, Qwen, Mistral, and DeepSeek model families. Computational resources used in this work were supported by our laboratory infrastructure and GPU research cluster.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore. Association for Computational Linguistics.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Mark Chen. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Sihan Chen, Dan Zhao, Jongwoo Ko, Colby Banbury, Huiping Zhuang, Luming Liang, Pashmina Cameron, and Tianyi Chen. 2026. Wina: Weight informed neuron activation for accelerating large language model inference. *Preprint*, arXiv:2505.19427.

Yuli Chen, Bo Cheng, Jiale Han, Yingying Zhang, Yingting Li, and Shuhao Zhang. 2025. Dlp: Dynamic layerwise pruning in large language models. *arXiv preprint arXiv:2505.23807*.

Eunyeong Cho, Jehyeon Bang, Ranggi Hwang, and Minsoo Rhu. 2026. PASCAL: A Phase-Aware Scheduling Algorithm for Serving Reasoning-based Large Language Models . In *2026 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 1–16, Los Alamitos, CA, USA. IEEE Computer Society.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Harry Dong, Beidi Chen, and Yuejie Chi. 2024. Prompt-prompted adaptive structured pruning for efficient llm generation. *arXiv preprint arXiv:2404.01365*.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, and 3 others. 2023. Palm-e: an embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pages 10323–10337. PMLR.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts. 2024. The unreasonable ineffectiveness of the deeper layers. *arXiv preprint arXiv:2403.17887*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Qi Le, Enmao Diao, Ziyan Wang, Xinran Wang, Jie Ding, Li Yang, and Ali Anwar. 2025. Probe pruning: Accelerating llms through dynamic pruning via model-probing. *Preprint*, arXiv:2502.15618.

Fenglin Li and 1 others. 2025. Medical reasoning in the era of LLMs: A systematic review of enhancement techniques and applications. *arXiv preprint arXiv:2508.00669*.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, and 1 others. 2023. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2025. Shortgpt: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20192–20204.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Xupeng Miao, Chunan Shi, Jiangfei Duan, Xiaoli Xi, Dahua Lin, Bin Cui, and Zhihao Jia. 2024. Spotserve: Serving generative large language models on preemptible instances. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ASPLOS '24, page 1112–1127, New York, NY, USA. Association for Computing Machinery.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR.

Qwen Team. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, and 1 others. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, and 1 others. 2023. Efficient large language models: A survey. *arXiv preprint arXiv:2312.03863*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Yuxin Wang, Minghua Ma, Zekun Wang, Jingchang Chen, Huiming Fan, Liping Shan, Qing Yang, Dongliang Xu, Ming Liu, and Bing Qin. 2024. Cfsp: An efficient structured pruning framework for llms with coarse-to-fine activation information. *Preprint*, arXiv:2409.13199.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Gen Li, Ajay Jaiswal, Mykola Pechenizkiy, Yi Liang, Michael Bendersky, Zhangyang Wang, and Shiwei Liu. 2024. Outlier weighed layerwise sparsity (owl): a missing secret sauce for pruning llms to high sparsity. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. 2022. Orca: A distributed serving system for {Transformer-Based} generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 521–538.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

A Performance across models

To validate Dart's generalizability, we evaluate its performance across architecturally diverse state-of-the-art models. We target 70% FFN sparsity and compare sparse models against their dense counterparts on seven zero-shot tasks and three multi-shot domain-specific benchmarks. Results are summarized in Table 5.

Across both zero-shot and multi-shot benchmarks, dense models consistently outperform their sparse counterparts, as expected under aggressive 70% FFN pruning. The magnitude of degradation reveals patterns tied to architecture, model scale, and task type.

Architectural Effects on Sparsity Resilience. Models with attention normalization mechanisms exhibit relatively stable neuron activations. Qwen3-14B maintains strong performance on multi-shot

Table 5: Comparison of dense and sparse performance across multiple LLM architectures at 70% sparsity.

Benchmark	Mode	Qwen3 -14B	Qwen3 -4B	DeepSeek-R1 -Qwen-8B	DeepSeek-R1 -Llama-8B	Llama -3.1-8B	Llama -3.2-3B	Mistral -7B	Mean
Zero-Shot Tasks									
BoolQ	Dense	86.70	83.33	84.92	83.43	83.09	74.04	86.42	83.13
	Sparse	65.69	57.40	68.65	75.47	66.21	53.24	78.87	66.50
RTE	Dense	76.53	76.53	74.37	72.20	71.12	54.15	75.81	71.53
	Sparse	54.15	52.71	63.54	62.45	55.23	55.23	63.18	58.07
WinoG.	Dense	74.43	70.96	67.96	68.35	74.51	69.46	74.59	71.47
	Sparse	59.04	55.49	56.67	63.46	65.98	60.22	68.82	61.38
ARC-e	Dense	81.90	75.93	76.52	66.20	82.53	72.05	81.73	76.69
	Sparse	66.71	49.20	61.11	55.43	59.43	50.38	65.07	58.19
ARC-c	Dense	59.22	51.11	54.78	42.66	54.95	46.33	60.07	52.73
	Sparse	44.03	35.49	38.91	39.08	38.99	33.96	46.42	39.55
HellaSwag	Dense	81.44	73.75	75.72	74.78	79.31	74.13	83.43	77.51
	Sparse	68.40	53.66	59.49	58.63	64.58	52.77	74.12	61.66
OBQA	Dense	48.60	40.80	42.60	44.80	45.60	41.00	48.60	44.57
	Sparse	38.20	34.60	34.80	35.60	37.60	36.80	41.20	36.97
Multi-Shot Tasks (5-shot)									
MMLU	Dense	82.15	75.56	72.29	57.84	66.61	58.14	62.54	67.30
	Sparse	52.33	40.03	44.49	39.26	34.15	28.51	43.35	40.30
MedMCQA	Dense	68.09	59.60	55.89	43.44	56.66	49.18	49.29	54.59
	Sparse	44.75	36.12	36.89	29.98	29.52	29.60	35.17	34.58
GPQA	Dense	40.34	36.64	35.18	25.57	31.38	29.63	31.80	32.93
	Sparse	30.59	30.97	27.00	22.77	25.90	24.13	26.16	26.79

tasks (e.g., 52.33% on MMLU vs. 82.15% dense), suggesting normalization stabilizes critical neuron pathways. Mistral-7B shows resilience on zero-shot tasks (e.g., 78.87% on BoolQ) because sliding-window attention encourages localized, non-overlapping neuron activations compatible with structured pruning.

DeepSeek-R1 models show mixed outcomes: the Qwen variant performs slightly better than the LLaMA variant on several zero-shot tasks, indicating that RL-optimized reasoning circuits distribute information differently across neurons depending on attention architecture.

Model Size and Neuron Redundancy. Larger models (Qwen3-14B, Mistral-7B) retain more performance under 70% sparsity because FFN layers contain redundant neurons encoding overlapping representations. Smaller models (LLaMA-3.2-3B) degrade faster, as each neuron carries multi-domain knowledge; pruning removes multiple capabilities at once.

Task-Type Sensitivity. General-domain zero-shot tasks (BoolQ, Winogrande, HellaSwag) exhibit moderate drops, as representations are distributed across many neurons. Dart preserves high-contribution neurons per layer, maintaining global representational structure. Domain-specific multi-shot tasks (MMLU, MedMCQA, GPQA) experience larger reductions, since specialized knowledge occupies a small subset of neurons; larger models mitigate this via redundancy, while smaller models show sharper drops.

B Comparison with Existing Static Pruning Techniques

Structured vs. Unstructured Pruning Objective. SparseGPT and WANDA perform unstructured weight pruning, eliminating individual parameters across both attention and MLP sublayers. In contrast, Dart performs structured neuron pruning and restricts pruning to the FFN (MLP) sublayers, keeping attention intact. This choice is principled: the attention mechanism governs token interactions and mediates shifts in knowledge domain during generation. Pruning attention weights risks degrading the model's ability to track contextual transitions, whereas FFN pruning primarily reduces representational capacity without directly interfering with domain routing.

Unstructured methods possess a structural advantage in that they remove fine-grained redundancy throughout the network, particularly in attention projections (QKV and output matrices), where pa-

Table 6: Zero-shot performance of LLAMA-2-7B across different pruning methods and sparsity allocation strategies. DART prunes FFN with the specified sparsity (70%, 80%, 85%). All other approaches prune both Attention and FFN layers at 70% sparsity.

METHOD	SPARSITY	BOOLQ	RTE	HELLASWAG	WINOGRANDE	ARC-E	ARC-C	OBQA	MEAN
DENSE	-	77.71	62.82	76.00	69.30	74.58	46.33	44.20	64.42
MAGNITUDE	UNIFORM	37.95	53.07	26.36	49.33	27.86	26.96	28.00	35.65
	OWL	40.03	52.35	30.10	48.54	30.72	26.37	27.00	36.44
	DLP	46.51	52.71	37.93	51.78	37.58	28.58	30.80	40.84
SPARSEGPT	UNIFORM	65.35	53.43	41.07	58.01	40.66	24.74	29.80	44.72
	OWL	67.92	53.07	47.97	62.04	47.31	26.02	31.80	48.02
	DLP	71.25	53.79	50.23	62.19	49.07	27.65	33.40	49.65
WANDA	UNIFORM	48.23	52.71	30.28	49.96	30.30	21.42	26.40	37.04
	OWL	62.11	52.71	37.46	56.27	42.05	24.06	30.20	43.55
	DLP	62.29	52.71	44.19	58.80	46.97	25.77	33.00	46.25
DART	70%	67.83	57.40	65.04	64.64	61.27	37.12	39.80	56.16
	80%	63.58	49.81	53.79	59.58	56.06	31.91	35.60	50.05
	85%	59.42	51.62	44.05	59.51	48.15	30.37	34.20	46.76

rameter overparameterization is substantial. This allows higher apparent sparsity while preserving the full neuron set. Structured pruning, by contrast, removes entire neurons and therefore eliminates representational subspaces. As sparsity increases, locating neurons that are completely non-contributory becomes progressively harder because most remaining units encode some useful information. From a systems perspective, unstructured sparsity requires specialized sparse kernels to yield runtime gains, whereas structured neuron pruning produces smaller dense matrices compatible with standard accelerator libraries.

The parameter-count difference is modest. Attention accounts for only ∼25–30% of parameters in LLaMA-style architectures, a fraction that continues to decline with mechanisms such as GQA and MLA. Consequently, although SparseGPT and WANDA prune attention while DART does not, the overall parameter gap is only ∼5%, while DART maintains comparable or superior accuracy even at 85% sparsity.

Performance Comparison. Despite pruning entire neurons and operating only on FFNs, DART consistently outperforms unstructured baselines. In Table 6, for LLaMA2-7B at 70% sparsity, DART attains a mean accuracy of 56.16%, compared to 49.65% for SPARSEGPT-DLP and 46.25% for WANDA-DLP. Eevn though DLP tries to optimize the per-layer pruning according to the fixed budget, yet static pruning approaches aren't able to increment the precision of the selected neurons. The advantage is pronounced on reasoning-intensive tasks such as HellaSwag (65.04% vs. 50.23%) and ARC-e (61.27% vs. 49.07%). Even at 80% sparsity, DART (50.05% mean) remains competitive with or exceeds unstructured methods operating at lower effective neuron removal.

A similar trend appears for LLaMA2-13B in Table 7 at 70% sparsity, DART achieves 61.85% mean accuracy versus 53.47% (SPARSEGPT-DLP) and 51.11% (WANDA-DLP), with consistent gains on general reasoning benchmarks such as BoolQ (76.81% vs. 74.22%) and ARC-e (68.01% vs. 54.84%).

These results suggest that preserving high-contribution knowledge neurons is more critical for reasoning performance than maintaining dense but low-impact weight connectivity. While unstructured methods exploit readily available redundancy especially in attention-structured FFN

Table 7: Zero-shot performance of LLAMA-2-13B across different pruning methods and sparsity allocation strategies. DART prunes FFN with the specified sparsity (70%, 80%, 85%). All other approaches prune both Attention and FFN layers at 70% sparsity.

METHOD	SPARSITY	BOOLQ	RTE	HELLASWAG	WINOGRANDE	ARC-E	ARC-C	OBQA	MEAN
DENSE	-	80.55	65.34	79.39	72.30	77.53	48.98	45.20	67.04
MAGNITUDE	UNIFORM	38.62	52.71	29.56	49.41	32.11	24.57	26.60	36.23
	OWL	38.65	52.71	43.89	54.54	37.63	28.84	28.40	40.67
	DLP	40.55	52.71	50.34	59.43	44.57	31.14	29.40	44.02
SPARSEGPT	UNIFORM	67.16	52.71	47.05	61.40	48.91	27.90	30.80	47.99
	OWL	69.45	54.87	52.86	65.27	53.24	30.38	35.80	51.70
	DLP	74.22	54.15	55.80	65.67	54.84	33.02	36.60	53.47
WANDA	UNIFORM	62.11	52.71	31.71	51.78	35.73	20.82	28.20	40.44
	OWL	63.67	52.71	46.30	60.85	51.01	28.24	34.00	48.11
	DLP	67.06	52.71	52.98	64.64	54.59	30.97	34.80	51.11
DART	70%	76.81	62.45	72.24	67.96	68.01	43.85	41.60	61.85
	80%	69.96	56.67	60.67	65.11	60.39	39.59	37.20	55.66
	85%	67.00	55.96	55.55	59.35	56.77	35.66	37.40	52.53

pruning guided by neuron-level importance better preserves functional knowledge pathways, enabling stronger performance at high sparsity.

C Subject-Dependent Neuron Activation Analysis

We present a cross-subject transfer analysis to empirically support the domain-specificity observation in Section 3. Pruning masks are derived from prompts of one MMLU subject and evaluated across prompts from all 20 subjects. Figures 5 and Figure 6 show the resulting accuracy heatmaps for LLaMA-3.2-3B and LLaMA-3.1-8B at 50% sparsity. The diagonal dominance - where each subject's own mask consistently yields the highest accuracy - confirms that the neuron subsets active at runtime are domain-aligned.

Subject-Conditioned Neuron Selection In most cases, the prompt subject achieves the highest accuracy on its own evaluation set, indicating that DART effectively captures subject-relevant neurons. However, cross-subject parity is occasionally observed. For example, in fig. 6, neurons selected from *High School Mathematics* achieve strong performance not only on their source domain but also on *College Mathematics*, *High School Statistics*, *Professional Accounting*, and *Econometrics*. Similarly, *Abstract Algebra* neurons yield comparable or higher accuracy on *College Computer Science*, *Formal Logic*, and *College Mathematics*, reflecting shared symbolic reasoning and formal structure.

Cross-Subject Transfer Patterns and Knowledge Overlap Subjects such as *Marketing*, *World History*, and *International Law* demonstrate comparatively strong cross-subject transfer. These domains share substantial linguistic and conceptual overlap with general world knowledge, allowing their associated information to be distributed across multiple neurons and partially reused by related subjects (see fig. 5). In contrast, narrowly scoped technical subjects such as *Abstract Algebra* exhibit weaker transfer, as their knowledge is concentrated within a smaller, more specialized region of the representation space, resulting in peak accuracy primarily when the prompt and evaluation subject coincide.

Model Scale and Knowledge Segregation Effects When comparing knowledge storage across model scales, a clear structural difference emerges. Larger models tend to organize knowledge with cleaner functional boundaries, where neuron groups are

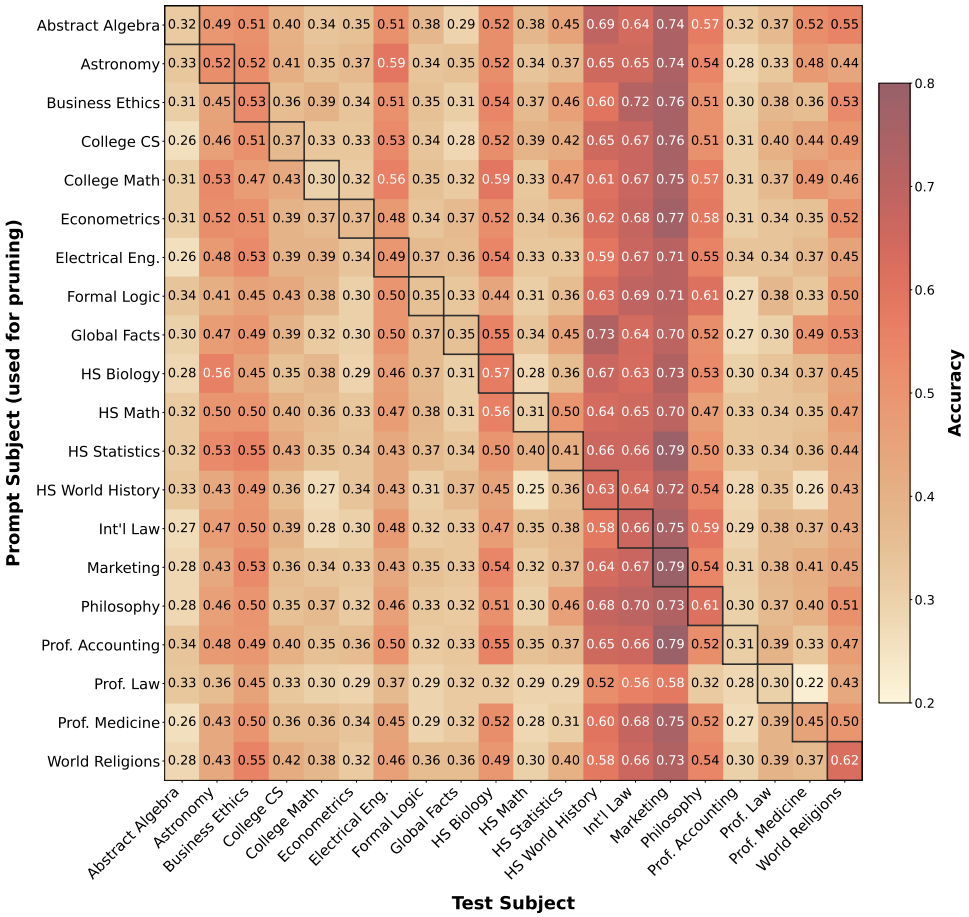


Figure 5: MMLU Accuracy (5-shot evaluation) for LLaMA 3.2-3B (50% Sparse) across 20 subjects.

more specialized and exhibit less cross-subject overlap. In contrast, smaller models operate under tighter parameter budgets, forcing individual neurons to encode multiple concepts or domains. This leads to higher functional superposition, where the same neurons are reused across different subjects.

This effect is visible in the heatmaps. In the smaller model (e.g., fig. 5), several subjects achieve accuracies close to the peak score of the source subject, indicating that prompts from different domains activate overlapping neuron subsets. As a result, cross-subject transfer appears artificially strong because shared neurons support multiple knowledge areas. In the larger model, however, subject-specific accuracy is more sharply peaked. For example, *World Religions* in fig. 6 achieves its highest accuracy primarily when evaluated with its own prompts, while other subjects show noticeably lower performance on that test set. This suggests stronger knowledge segregation: neurons associated with a subject are activated predominantly by semantically aligned prompts rather than by unrelated domains. Thus, increasing model scale promotes more disentangled and domain-specialized knowledge representations. This suggests that DART's runtime sensitivity signal becomes increasingly precise at isolating true knowledge boundaries as model capacity grows

These outcomes depend on prompt composition, sampled content, and evaluation question distribution, as well as the model's internal knowledge organization, which is shaped by both training data and architecture. Overall, the results support the presence of subject-aligned knowledge neuron subsets and validate DART 's ability to identify them. These results empirically corroborate the three-regime characterization of Section 3: domain-specificity is concentrated in middle layers, and DART's runtime sensitivity signal successfully isolates the corresponding neuron subsets without calibration data.

D Layer-Wise Pruning

This appendix consolidates the complete layer-wise pruning framework of DART: the theoretical motivation for non-uniform sparsity allocation (Section 3) and comprehensive ablation experiments validat-

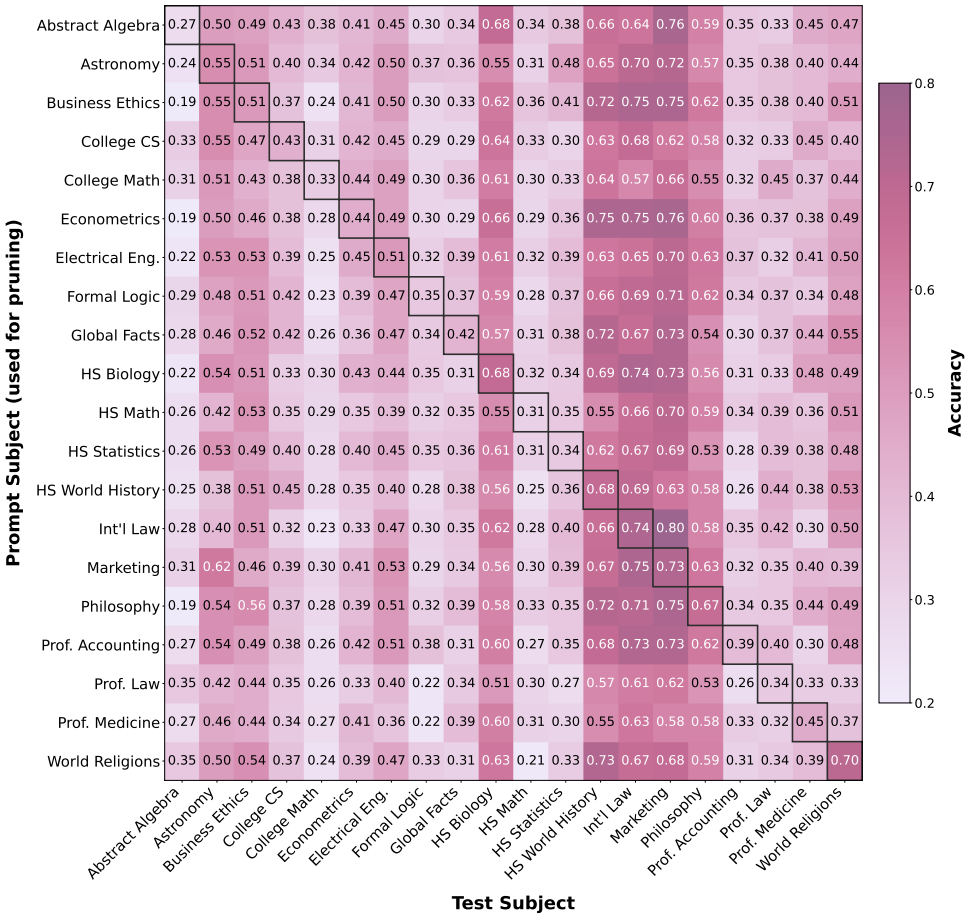


Figure 6: MMLU Accuracy (5-shot evaluation) for LLaMA 3.1-8B (50% Sparse) across 20 subjects.

ing each component.

D.1 Layer-Wise Contextual Sparsity

The FFNs of LLMs are characterized as repositories of factual knowledge, where individual neurons act as knowledge entries that are selectively activated depending on the input context (Geva et al., 2021). This indicates that the number of neurons that can be pruned is inherently heterogeneous across layers.

Empirical motivation. We first establish the need for non-uniform allocation empirically. Figure 7 reports the change in MMLU accuracy when each layer is independently pruned to 70% sparsity. The results reveal strong subject-dependent and layer-dependent variability - no single layer exhibits universal behaviour across all domains.

A consistent qualitative pattern nevertheless emerges. Early layers tend to be more sensitive: pruning them often causes accuracy degradation across subjects. These layers primarily encode lower-level lexical, syntactic, and compositional features that form the foundation of downstream reasoning (Dai et al., 2022). Disrupting them corrupts the feature pipeline before higher-level abstractions are formed. In contrast, pruning later layers frequently leads to neutral or even improved performance. Later layers encode higher-level semantic and domain-specific features (Dai et al., 2022). Removing neurons that express knowledge unrelated to the active prompt reduces representational interference, effectively narrowing the active semantic subspace. This supports our hypothesis that later FFN layers contain more topic-specific knowledge neurons, and pruning them can suppress off-topic activations, enabling more focused reasoning.

Sensitivity patterns vary substantially across subjects. Domains such as *High School Biology* often benefit from pruning, suggesting that their knowledge is concentrated in a relatively compact neuron subset - removing unrelated neurons reduces competition during inference. In contrast, *High School World History* exhibits consistent degradation under pruning, indicating that its knowledge is distributed across many neurons, where removing even a small subset disrupts necessary represen-

Table 8: Per-layer activation energy profile. Top-5% energy denotes the fraction of total layer activation energy carried by the top 5% of neurons ranked by mean absolute activation. CV denotes the coefficient of variation of neuron activation magnitudes within each layer. Statistics are averaged across 95 prompts spanning 20 knowledge domains.

Layer Range	Top-5% Energy	CV	Profile
L0	13.7 ± 0.8	0.69 ± 0.06	Early
L2–L12	11.2–$12.6 \pm \sim 0.3$	0.44–0.54	Flat
L13–L26	12.4–$14.3 \pm \sim 0.5$	0.56–0.69	Moderate
L27–L31	15.7–$32.6 \pm \sim 1.0$	0.76–1.84	Peaky

tational components. Subjects with concentrated knowledge representations tolerate aggressive pruning and may even benefit from it, while subjects with distributed representations are inherently more pruning-sensitive. These findings establish that a single uniform sparsity ratio cannot simultaneously respect 1) early-layer sensitivity, 2) exploit late-layer expendability, and 3) accommodate subject-level variation in knowledge distribution.

Activation geometry: why layers differ. To understand the mechanistic basis of this sensitivity hierarchy, we measure the *activation energy* of every neuron which is its mean absolute value of the intermediate representation $\mathrm{SiLU}(W_{\text{gate}}\, x) \odot (W_{\text{up}}\, x)$ across 95 diverse domain-specific passages spanning 20 knowledge domains from our custom-prompt generation tempaltes on LLaMA-3.1-8B. The coefficient of variation CV across neurons within a layer quantifies how concentrated energy is: high CV indicates a small neuron elite dominates; low CV indicates near-uniform firing across the neuron population.

Table 8 reveals three regimes that directly account for the MMLU sensitivity pattern in Figure 7. **Flat layers (L2-L12)** have near-uniform activation distributions (CV 0.44-0.54): thousands of neurons cluster near the top-k selection boundary with nearly identical magnitudes, making the pruning mask highly sensitive to input context - small domain-induced perturbations cause large rank reshuffling at the boundary. This explains the subject-dependent variability observed in Figure 7: the flat distribution makes the mask's composition maximally context-sensitive, so the impact of pruning fluctuates sharply across subjects and layers. **Peaky layers (L27-L31)** exhibit the opposite structure (CV 0.76-1.84): at L31, the top 5% of neurons carry 32.6% of total layer energy and the top 50% carry 84.8%. A compact dominant elite is always selected regardless of domain, making late-layer masks stable and resilient to aggressive pruning - consistent with the neutral or improved MMLU accuracy observed when late layers are pruned. **Early layers (L0, L2–L7)** occupy an intermediate regime, but are positioned at the base of the residual stream where any perturbation propagates through all downstream computation, amplifying their effective sensitivity beyond what the local distribution alone would suggest - consistent with the broad early-layer degradation in Figure 7.

D.2 Ablation Experiments

D.2.1 Experimental Setup

Model. LLaMA-3.1-8B-Instruct (L=32 layers, intermediate dimension 14,336).

Benchmarks. Five zero-shot NLP tasks via lm-evaluation-harness (200 samples each): BoolQ, WinoGrande, ARC-Easy, ARC-Challenge, and OpenBookQA. For generation quality, CNN/DailyMail summarization (30 samples, 256 max output tokens, ROUGE scoring).

Pruning. Magnitude-based neuron selection with prompt-conditioned masking. Each evaluation sample receives its own mask computed from the prefill activations.

Layer regions. We partition the 32 layers into three functional regions: early (L0–L7, 8 layers), middle (L10–L21, 12 layers), and late (L24–L31, 8 layers).

Budget constraint. All experiments within a sparsity level maintain identical total pruning budgets. When a target region is pruned more aggressively, the freed budget is redistributed uniformly to the remaining layers.

D.2.2 Part I: Layer Sensitivity and Budget Distribution

We establish that transformer layers exhibit position-dependent sensitivity to structured FFN pruning, motivating the need for non-uniform budget allocation and safety bounds. For each region, we apply three levels of targeted pruning (mild, moderate, severe) while redistributing the freed budget to the remaining layers, holding the global sparsity constant.

Finding 1: Early layers exhibit the highest pruning sensitivity. Tables 9–10 demonstrate a consistent hierarchy: pruning early layers (L0–L7) to 90% at 50% global sparsity reduces average accuracy

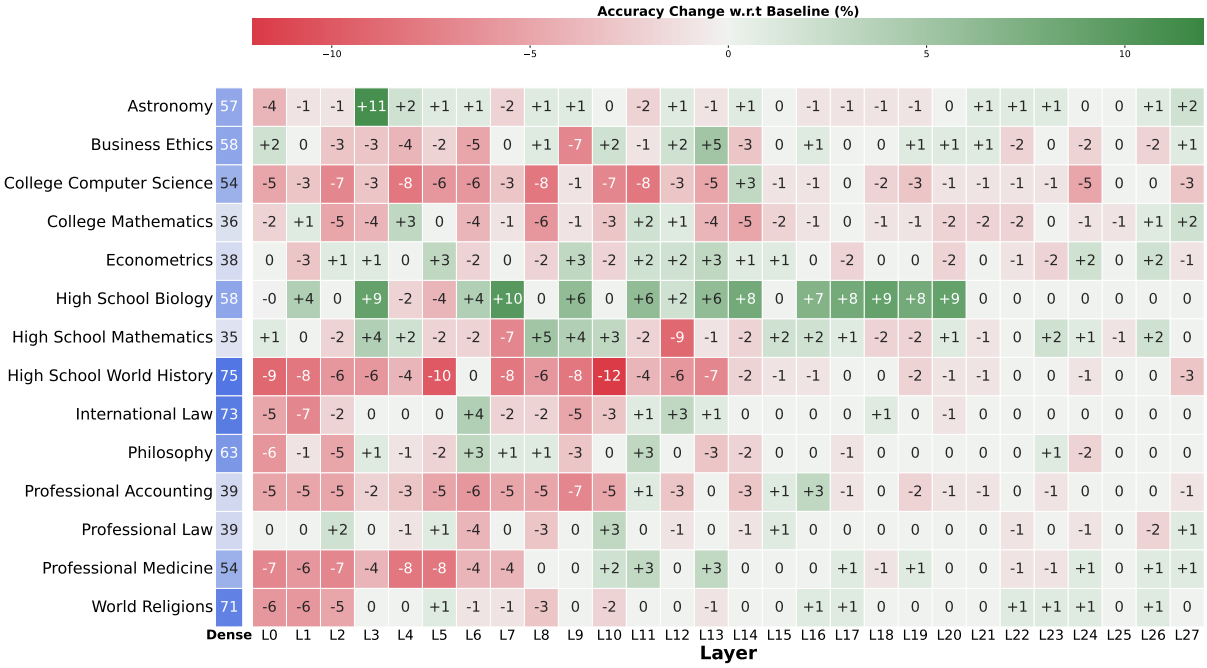


Figure 7: Impact on MMLU accuracy of sparsifying each layer by 70% for different subjects in LLaMA-3.2-3B. The dense model's accuracy is listed in the blue column next to the subject name. The heatmap shows the accuracy change relative to the dense model.

by 8.3 points relative to uniform (60.80 → 52.50), with ARC-Challenge declining by 10.0 points. At 70% sparsity, the same configuration degrades accuracy by 5.9 points, with ARC-Challenge dropping 13.5 points. This asymmetry reflects the architectural role of early layers in establishing token and positional representations that propagate through the entire residual stream.

Finding 2: Late layers are expendable across classification and generation. Tables 9, 10, and 12 establish that late layers (L24–L31) tolerate aggressive pruning. At 70% sparsity, late-severe pruning improves average classification accuracy by 1.5 points over uniform (Table 10) and improves ROUGE-1 by 2.84 points on CNN/DailyMail summarization (Table 12). The freed budget strengthens earlier layers more than the late layers need it. This resilience is consistent with the representational geometry of deep transformers: by the final quarter of the network, MLP sub-layers apply narrow, fine-grained corrections to an already well-formed representation. The residual stream already encodes sufficient information from earlier layers, and the lm_head operates at full rank regardless of FFN pruning. The budget consumed by late-layer neurons is therefore more productively allocated to early and middle layers, where the representations are still being constructed, and neuron-level perturbations propagate through all downstream computation.

Finding 3: Middle layers exhibit task-dependent sensitivity. Tables 9 and 10 reveal that middle layers (L10–L21) are robust for simpler NLP tasks (BoolQ: −1.0 pt at 50% severe) but sensitive for knowledge-intensive reasoning (ARC-Easy: −10.5 pts, OBQA: −8.0 pts at 50% severe). This is consistent with the functional role of middle layers in factual knowledge storage and retrieval.

Finding 4: The $p_{\min}$ safety floor is empirically necessary. Table 11 demonstrates that pruning any region below 10% retention causes catastrophic degradation regardless of position: early −21.6 pts, middle −16.4 pts, late −6.1 pts at 50% sparsity. The position-independence of this failure validates the universal lower bound $p_{\min} = 0.10$.

Finding 5: Non-uniform allocation outperforms uniform pruning. Across all conditions, the best non-uniform configuration matches or exceeds uniform at the same total budget. The improvement grows with sparsity: at 70%, late-severe improves classification by +1.5 pts and ROUGE-1 by +2.84 pts over uniform (Tables 10, 12), demonstrating the practical value of sensitivity-aware budget allocation.

Table 9: Region-targeted pruning at 50% sparsity. Late-layer pruning achieves the best average performance under a fixed budget.

Configuration		Pruning		Accuracy					
Type	Target	Target	Rest	BoolQ	WinoG	ARC-E	ARC-C	OBQA	Avg
Uniform	All	50	50	82.5	68.0	72.5	46.5	34.5	60.8
Early mild	L0–7	80	40	79.0	67.5	73.5	43.5	34.0	59.5
Early mod.	L0–7	85	38.3	73.5	70.0	66.0	40.5	32.5	56.5
Early sev.	L0–7	90	36.7	68.0	64.5	62.5	36.5	31.0	52.5
Late mild	L24–31	80	40	82.0	73.0	71.5	50.5	33.5	62.1
Late mod.	L24–31	85	38.3	85.0	73.0	74.0	47.5	31.5	62.2
Late sev.	L24–31	90	36.7	80.5	72.5	70.5	49.0	32.0	60.9
Mid mild	L10–21	80	32	79.0	67.5	65.0	46.0	30.0	57.5
Mid mod.	L10–21	85	29	80.0	64.5	63.0	43.5	30.0	56.2
Mid sev.	L10–21	90	26	81.5	66.0	62.0	40.0	26.5	55.2

Table 10: Region-targeted pruning at 70% sparsity. Late-layer configurations remain the most robust under aggressive pruning.

Configuration		Pruning		Accuracy					
Type	Target	Target	Rest	BoolQ	WinoG	ARC-E	ARC-C	OBQA	Avg
Uniform	All	70	70	72.0	66.0	64.0	44.0	32.0	55.6
Early mild	L0–7	80	66.7	67.5	64.5	64.0	39.0	29.0	52.8
Early mod.	L0–7	85	65	70.5	62.0	60.5	36.0	31.0	52.0
Early sev.	L0–7	90	63.3	65.5	63.0	61.0	30.5	28.5	49.7
Late mild	L24–31	80	66.7	78.0	64.0	67.0	41.5	28.5	55.8
Late mod.	L24–31	85	65	75.0	68.5	62.5	42.0	32.0	56.0
Late sev.	L24–31	90	63.3	72.5	69.0	67.0	43.5	33.5	57.1
Mid mild	L10–21	80	64	73.5	67.0	61.0	40.5	32.0	54.8
Mid mod.	L10–21	85	61	72.5	61.5	59.0	36.5	31.0	52.1
Mid sev.	L10–21	90	58	66.5	63.0	55.0	38.0	27.5	50.0

D.2.3 Part II: Functional Importance Formula

Part I establishes that layers have position-dependent sensitivity and that non-uniform allocation is beneficial. We now examine whether the functional importance signal S_l in Eq. 4 produces an allocation that naturally reflects this sensitivity hierarchy—specifically, whether the formula assigns higher importance to early layers and lower importance to late layers without any explicit depth information.

Cosine-only importance. The directional component alone, $S_l^{\cos} = 1 - \cos(\mathbf{x}_{\text{pre}}, \mathbf{x}_{\text{post}})$, captures layers that reorient the representation but is insensitive to layers that make large-magnitude changes in a similar direction.

Table 13 demonstrates why both terms are necessary. Under cosine-only allocation, the directional signal alone over-protects middle layers (L5–L11 at $\sim$90%) where the MLP induces the largest angular deviation, while under-protecting early layers (L0 at 37.6%, L1 at 25.2%) despite their established sensitivity (Finding 1). The intensity term corrects this imbalance: early layers operate on a residual stream with low accumulated norm, so their updates represent a proportionally large modification relative to the input (high $\|\delta\mathbf{x}\|/\|\mathbf{x}_{\text{pre}}\|$). The full formula raises L0 from 37.6% to 62.6% and L1 from 25.2% to 39.3%.

For later layers, the situation is reversed. Additive skip connections accumulate large residual norms, so even substantial absolute updates represent a small relative intensity. The cosine term further confirms that these layers produce minimal directional change. Both terms agree that late layers are expendable, consistent with Finding 2.

The full multiplicative formula outperforms cosine-only by 0.9 points on average (61.70 vs. 60.80), with the largest gains on ARC-Easy (+4.0 pts) and ARC-Challenge (+3.5 pts) which are among the knowledge-intensive tasks that are most sensitive to the early-layer under-protection caused by the cosine-only formulation (Table 13). The full S_l also approaches the best hand-tuned configuration from Part I (62.20), confirming that the sensitiv-

Table 11: Lower-bound violation ($p_{\min}$) at 50% and 70% sparsity. Aggressive pruning (98%) in early and middle layers leads to substantial accuracy degradation, while late layers remain comparatively robust.

Configuration		Pruning	Accuracy						
Type	Target	Target	BoolQ	WinoG	ARC-E	ARC-C	OBQA	Avg	Δ
50% Sparsity									
Uniform	All	50	82.5	68.0	72.5	46.5	34.5	60.8	—
$p_{\min}$ early	L0–7	98	44.5	54.5	48.5	27.0	21.5	39.2	−21.6
$p_{\min}$ late	L24–31	98	60.5	73.5	64.0	47.0	28.5	54.7	−6.1
$p_{\min}$ mid	L10–21	98	68.5	57.5	44.5	29.0	22.5	44.4	−16.4
70% Sparsity									
Uniform	All	70	72.0	66.0	64.0	44.0	32.0	55.6	—
$p_{\min}$ early	L0–7	98	45.5	50.0	48.5	22.5	20.0	37.3	−18.3
$p_{\min}$ late	L24–31	98	53.0	70.0	61.5	42.0	26.5	50.6	−5.0
$p_{\min}$ mid	L10–21	98	57.0	64.0	48.0	31.5	21.0	44.3	−11.3

Table 12: Generation quality under late-layer pruning on CNN/DailyMail using greedy decoding with 256 maximum generation tokens.

Configuration	R1	R2	RL	RLsum
50% Uniform	15.95	5.90	11.77	13.90
50% Late severe (L24–31 at 90%)	16.17	5.92	11.37	13.87
70% Uniform	12.95	4.08	9.38	10.84
70% Late severe (L24–31 at 90%)	**15.79**	**5.78**	**10.99**	**13.28**

ity signal captures meaningful per-layer variation without requiring manual tuning. The remaining gap is closed by adding early depth protection D_l (Part III).

Hand-tuned configuration study. To further validate the limitations of cosine-only importance, we conducted additional experiments by varying per-layer pruning ratios manually and measuring zero-shot accuracy on *BoolQ*. We evaluated configurations both without early/late-layer cushioning (*Config2, Config3*) and with cushioning (*Config5–Config7*). Although the cosine-similarity signal generally tracked layer influence, we observed notable exceptions: some layers induced minimal embedding-direction change yet produced downstream accuracy gains of up to 10%, indicating nonlinear propagation effects not captured by the directional term alone.

In Figure 8, we compare multiple configurations at an overall 50% MLP sparsity budget. *Config1* and *Config3* achieve the highest accuracy; both preserve neurons in layers immediately preceding the mid-layer region (layers that showed limited cosine-similarity impact in Table 13). This confirms that embedding direction change alone cannot fully capture functional importance, and provides additional motivation for the intensity term in Eq. 4. Overall, layer criticality is task-dependent and not reliably inferable from local embedding metrics alone; the full $S^{(l)}$ formula, validated in Table 13, serves as a principled and practical heuristic for maintaining a safe pruning regime while preserving accuracy. The precise mechanistic basis of these behaviours remains difficult to formalize. In this respect, we echo the observation that empirical progress often precedes theoretical understanding. As Shazeer (2020) remarks, "We offer no explanation as to why these architectures seem to work; we attribute their success, as all else, to divine benevolence."

D.2.4 Part III: Depth Factor D_l Parameter Ablation

Part II establishes that the functional importance signal S_l produces a reasonable but imperfect allocation. We now examine whether the depth-aware protection factor D_l can close the gap by explicitly protecting sensitive early layers. We sweep each parameter independently while holding the others fixed. All experiments use DART's auto allocation at 50% global sparsity on LLaMA-3.1-8B-Instruct.

Finding 6: Early depth protection (α) monotonically improves accuracy. Table 15 shows that reducing α from 1.0 (no protection) to 0.10 provides a consistent accuracy gain of 1.5 points (61.70 → 63.20), with ARC-Challenge improving by 3.0 points. Table 19 confirms this: DART's depth-aware configuration (α=0.10) achieves 63.20 average accuracy, a +1.5-point improvement over

Table 13: Per-layer pruning distribution: cosine-only vs. full S_l. Both allocations use DART's auto mode at 50% sparsity with D_l disabled. Shown as retention percentage ($100\% - \text{pruned}\,\%$) for representative layers.

Layer	Cosine-Only $S_l^{\cos}$	Full S_l (cos $\times$ norm)	Difference
L0 (early)	37.6%	62.6%	+25.0%
L1 (early)	25.2%	39.3%	+14.1%
L2 (early)	42.9%	50.5%	+7.6%
L5	90.0%	80.5%	−9.5%
L10	90.0%	74.7%	−15.3%
L15	68.2%	58.1%	−10.1%
L20	18.3%	27.7%	+9.4%
L25	10.0%	20.2%	+10.2%
L31 (final)	79.9%	90.0%	+10.1%

Table 14: Importance formula accuracy comparison. All auto-allocated configurations use D_l disabled (α=1.0, β=1.0) to isolate the formula's effect.

Configuration	Source	BoolQ	WinoG	ARC-E	ARC-C	OBQA	Avg
Late moderate (hand-tuned)	Tbl 9	85.00	73.00	74.00	47.50	31.50	62.20
Late severe (hand-tuned)	Tbl 9	80.50	72.50	70.50	49.00	32.00	60.90
Auto, cosine-only	Auto	83.00	69.50	68.50	47.00	36.00	60.80
Auto, full S_l	Auto	83.50	69.00	72.50	50.50	33.00	**61.70**

pure sensitivity-based allocation without D_l (61.70). The depth factor compensates for a limitation of the data-driven importance signal S_l, which tends to under-protect early layers whose high functional importance is partially masked by the cosine similarity metric.

Finding 7: Late depth protection has diminishing returns but provides stability. Table 16 shows that aggressive late protection (β=0.10) underperforms the no-protection baseline (61.40 vs. 61.70), confirming that over-allocating budget to late layers is wasteful (Finding 2). However, Table 18 reveals that individual benchmarks oscillate as w_l varies: ARC-Challenge fluctuates between 47.00 and 51.50, WinoGrande between 68.50 and 70.00, and OBQA between 33.00 and 35.50 across the w_l sweep. While w_l=0.0 achieves the highest average, this per-benchmark variance suggests that a small conservative late zone (w_l=0.1) provides stability across diverse evaluation conditions without materially reducing the overall budget available to early and middle layers.

Rationale for D_l. The functional importance signal S_l (Eq. 4) captures per-layer sensitivity through runtime activation statistics and, as shown in Part II, naturally produces a distribution that prunes later layers more aggressively. However, S_l alone systematically under-protects early layers. This occurs because early MLP sub-layers operate on low-level token embeddings where the cosine similarity between pre- and post-MLP representations remains relatively high - the MLP refines the embedding within a similar subspace rather than reorienting it. The resulting low $(1 - \cos)$ term deflates the measured importance of early layers, despite their outsized contribution to downstream accuracy (Finding 1).

The depth factor D_l addresses this calibration gap through a position-aware prior that reduces pruning pressure on early layers. Crucially, D_l does not override the data-driven signal—it modulates it. Layers within the protected zone still receive differentiated budgets based on their measured S_l; D_l merely ensures that the allocator cannot starve the early layers where the importance metric is known to underestimate true sensitivity. The ablation results confirm this design: adding early protection (α=0.10) to the sensitivity-based allocation improves accuracy by +1.5 points (Table 19). For late layers, aggressive protection is counterproductive (Finding 7), but a conservative safety

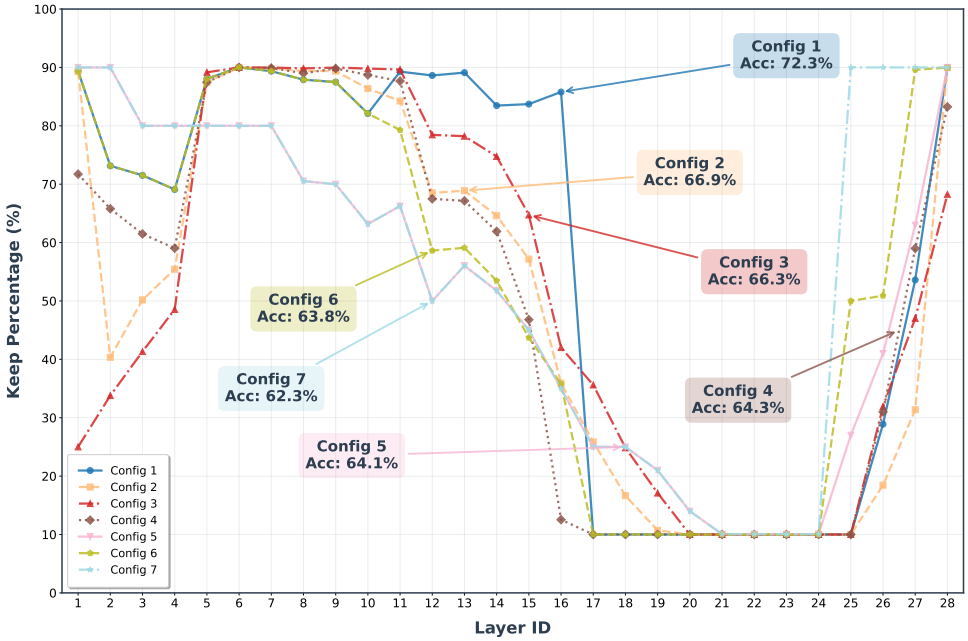


Figure 8: Layer-wise kept neuron(%) for different pruning configurations (50% overall MLP sparsity) and the resulting zero-shot accuracy on BoolQ.

Table 15: Early protection scaling factor (α). Fixed: w_e=0.3, w_l=0.1. β is set to 1.0 (disabled).

α	BoolQ	WinoGrande	ARC-E	ARC-C	OBQA	Avg
0.10	82.50	70.50	**74.50**	**53.50**	**35.00**	**63.20**
0.25	**83.50**	**71.50**	72.50	50.00	33.50	62.20
0.50	82.50	68.00	74.00	51.00	34.50	62.00
0.75	82.00	71.00	74.00	48.00	35.00	62.00
1.00 (off)	83.50	69.00	72.50	50.50	33.00	61.70

margin (w_l=0.1, β=0.35) is retained to stabilize per-benchmark variance observed in Table 18.

E Detecting Knowledge Drift

We investigate *knowledge drift* - a failure mode that arises when runtime neuron pruning prevents the model from accessing the knowledge subspace required for a semantic domain transition. To isolate this phenomenon, we construct a controlled two-domain generation task requiring a topic switch from *electric vehicles* to *Italian cuisine*. The prompt is shown below.

Prompt

Follow these instructions carefully:
Step 1: Write a paragraph explaining electric cars. Include:
- What they are
- How they differ from gasoline vehicles
- How their motors and batteries work
- Why they are considered environmentally friendly
Step 2: After finishing the electric car explanation, write a completely separate paragraph about Italian cuisine. Include:
- Popular dishes
- Key ingredients
- Regional differences in cooking styles
Begin now with Step 1.

Dense Baseline Behavior. The dense model successfully completes both steps, generating coherent domain-specific content for each topic. This confirms that the task primarily stresses domain transition rather than knowledge availability.

Table 16: Late protection scaling factor (β). Fixed: w_e=0.3, w_l=0.1. α is set to 1.0 (disabled).

β	BoolQ	WinoGrande	ARC-E	ARC-C	OBQA	Avg
0.10	**84.00**	71.00	71.00	47.00	34.00	61.40
0.25	83.00	**73.00**	73.00	48.00	34.00	62.20
0.35	83.50	70.50	72.00	49.50	**35.00**	62.10
0.50	82.50	68.50	72.50	47.00	**35.00**	61.10
0.75	82.50	72.50	73.00	**51.50**	34.50	**62.80**
1.00 (off)	83.50	69.00	72.50	50.50	33.00	61.70

Table 17: Early zone width (w_e). Fixed: w_l=0.1, α=0.25, β=0.35. For L=32: w_e=0.1 protects L0–L3; w_e=0.2 protects L0–L6; w_e=0.3 protects L0–L9; w_e=0.4 protects L0–L12.

w_e	Protected	BoolQ	WinoGrande	ARC-E	ARC-C	OBQA	Avg
0.00 (off)	none	83.00	70.50	72.00	**52.00**	33.00	62.10
0.10	L0–3	**84.00**	70.50	72.00	47.50	34.50	61.70
0.20	L0–6	82.00	**71.00**	72.00	51.50	**35.50**	**62.40**
0.30	L0–9	83.50	68.50	**73.50**	47.00	**35.50**	61.60
0.40	L0–12	84.00	69.00	71.50	48.00	34.00	61.30

Dense model's generation

Write your paragraph about electric cars.
Step 1: Electric Cars Explanation
Electric cars, also known as electric vehicles (EVs), are a type of vehicle that uses electric motors powered by a battery instead of a gasoline engine. Unlike traditional gasoline vehicles, electric cars do not rely on fossil fuels for energy, making ... [SKIPPING TEXT] ... gasoline vehicles, making them a more sustainable choice for transportation.
Step 2: Italian Cuisine Explanation
Italian cuisine is renowned for its rich flavors, aromas, and diverse regional specialties. Popular dishes include pasta carbonara, pizza margherita, and risotto alla Milanese. Key ingredients in Italian cooking include olive oil, garlic, tomatoes, and cheese, such as parmesan and mozzarella. Regional differences in cooking styles are evident ... [SKIPPING TEXT] ... Italian cuisine is also famous for its use of fresh ingredients, herbs, and spices, which are often combined to create simple yet flavorful dishes that showcase the country's culinary heritage.

Failure Under Runtime Sparsification. We then apply 50% structured neuron sparsity after the 100th generated token-during the *electric vehicle* segment. The model continues producing fluent content about the first domain but fails to generate meaningful text after the prompt requires a topic shift. Instead, it produces structural headings and repetitive fragments related to *Italian cuisine* without semantic substance. This indicates preserved syntactic planning but impaired semantic retrieval, suggesting that neurons required for the second domain were pruned.

50% Sparsity at the 100th generated token

Write your paragraph about electric cars.
Step 1: Electric Cars Explanation
Electric cars, also known as electric vehicles (EVs), are a type of vehicle that uses electric motors **...** range range, and charging speed to make them a viable option for many potential owners.
Step 1 (continued): Electric Cars Explanation (continued)
The electric motor in an EV is a key component that makes the difference between an electric **...** are a rapidly growing technology, with many manufacturers offering a range of models, from consumer to commercial, that are designed to meet the needs of different potential owners.
Step 2: Italian Cuisine Explanation
Step 2 (continued): Italian Cuisine Explanation (continued)
Step 2 (continued): Italian Cuisine Explanation (continued): Regional Differences in Cooking Styles
Step 2 (continued): Italian Cuisine Explanation (continued): Regional Differences in Cooking Styles (continued)
Step 2 (continued): Italian Cuisine Explanation (continued): Regional Differences in Cooking Styles (continued)

Neuron-Release Experiments. To test whether this failure stems from temporary inaccessibility of relevant neurons, we re-enable pruned neurons at different timesteps. Releasing the mask at token 400 restores coherent generation about *Italian cuisine*, demonstrating that the knowledge itself was not erased but dynamically inaccessible. However, releasing at token 500 fails to recover performance,

Table 18: Late zone width (w_l). Fixed: w_e=0.3, α=0.25, β=0.35.

w_l	Protected	BoolQ	WinoGrande	ARC-E	ARC-C	OBQA	Avg
0.00 (off)	none	**85.50**	69.00	73.00	**50.00**	**35.00**	**62.50**
0.05	L29–31	83.00	68.50	73.00	51.50	**35.00**	62.20
0.10	L28–31	83.50	68.50	**73.50**	47.00	35.50	61.60
0.15	L26–31	82.00	**70.00**	**73.50**	48.50	34.00	61.60
0.20	L25–31	81.50	**70.00**	72.00	50.50	33.50	61.50

Table 19: Effect of depth-aware protection on allocation quality. Comparison of DART's auto allocation with D_l disabled (pure sensitivity-based) vs. D_l enabled.

Configuration	α	β	BoolQ	WinoG	ARC-E	ARC-C	OBQA	Avg
Without D_l	1.0	1.0	83.50	69.00	72.50	50.50	33.00	61.70
DART (with D_l)	**0.10**	**1.0**	82.50	70.50	**74.50**	**53.50**	**35.00**	**63.20**

implying that prolonged drift causes the generation trajectory to collapse into a degenerate knowledge basin from which recovery is unlikely.

Releasing sparsity mask at 400th generated token

Write your paragraph about electric cars.
Step 1: Electric Cars Explanation
Electric cars, also known as electric vehicles (EVs), are a type of vehicle that uses electric motors powered by... lower operating costs, and increased efficiency. However, they also require a range of factors, such as charging infrastructure, range range, and charging speed to make them a viable option for many potential owners.
Step 1 (continued): Electric Cars Explanation (continued)
The electric motor in an EV is a key component that makes the difference between an electric and a gasoline vehicle. ... range range, and charging speed to make them a viable option for many potential owners. Overall, electric cars are a rapidly growing technology, with many manufacturers offering a range of models, from consumer to commercial vehicles.
Step 2: Italian Cuisine Explanation
Italian cuisine is a rich and diverse culinary tradition that originated in Italy. Popular dishes include pasta, pizza, and risotto, which are often made with ingredients such as olive oil, garlic, and ... lighter and more acidic flavors. Overall, Italian cuisine is a beloved and iconic part of Italian culture, and its influence can be seen in cuisine around the world.
The final answer is: There is no final numerical answer to this problem, as it involves writing two paragraphs about electric cars and Italian cuisine.

Releasing sparsity mask at 500th generated token

Write your paragraph about electric cars.
Step 1: Electric Cars Explanation
Electric cars, also known as electric vehicles (EVs), are a type of vehicle that uses electric motors powered by a battery instead of a gasoline engine. Unlike ... costs, and increased efficiency. However, they also require a range of factors, such as charging infrastructure, range range, and charging speed to make them a viable option for many potential owners.
Step 1 (continued): Electric Cars Explanation (continued)
The electric motor in an EV is a key component that makes the difference between an electric and a gasoline vehicle. While a gasoline engine is powered by an ... electric cars are a rapidly growing technology, with many manufacturers offering a range of models, from consumer to commercial, that are designed to meet the needs of different potential owners.
Step 2: Italian Cuisine Explanation
Step 2 (continued): Italian Cuisine Explanation (continued)
Step 2 (continued): Italian Cuisine Explanation (continued): Regional Differences in Cooking Styles
Step 2 (continued): Italian Cuisine Explanation (continued): Regional Differences in Cooking Styles (continued)

Attention-Space Drift Measurement. We plot in fig. 9 the cosine similarity between each token's attention-sublayer embedding and a reference embedding computed from all tokens preceding sparsification. This reference represents the semantic context active at the pruning point. A consistent pattern emerges: once pruned neurons are re-enabled, the similarity trajectory rapidly realigns with that of the dense model, indicating recovery of the original representational manifold.

Importantly, even without mask release

(*Prune@100, No Release*), the similarity curve shows an attempted departure from the initial manifold, demonstrating that the model's attention dynamics still signal a domain transition. However, releasing neurons too late (e.g., at the 500th token) produces a distinct failure signature: similarity remains confined to a narrow band, suggesting the generation trajectory has collapsed into a restricted semantic basin. Beyond this point, restoring neurons does not recover the intended trajectory.

Fine-grained cases in fig. 10 further illustrate this mechanism. In panels (b) and (c), timely neuron restoration enables a sustained shift in similarity, corresponding to successful transition to the second topic. When neurons remain unavailable, the trajectory repeatedly reverts toward the original manifold, indicating that the model lacks the representational capacity to sustain the new domain. In panel (d), late intervention fails to alter the trajectory, confirming that knowledge drift has become irreversible.

Overall, attention-space divergence provides an early signal of domain shift attempts, while the persistence of low-diversity similarity trajectories indicates semantic trapping. Successful generation therefore depends on synchronizing neuron availability with the evolving attention-driven knowledge trajectory.

F Knowledge Tracer Ablation Study

F.1 Evaluation Protocol

Motivation: why a custom multi-domain dataset. Standard summarization benchmarks used in the main paper (CNN/DailyMail, GovReport, MultiNews) provide long, single-domain input documents. As seen in Section 5, these benchmarks produce very few drift retriggers during generation-the long input context establishes a strong, stable knowledge space, and the generated summary stays within the same domain throughout. In these settings, the Knowledge Tracer serves primarily as a safety net that rarely activates.

However, no widely-used benchmark exists that forces the model to generate text spanning multiple, unrelated knowledge domains within a single output. This is precisely the scenario where drift detection is most critical: the pruning mask is built from the initial domain's activations and becomes stale when the model transitions to a new domain. To stress-test the Knowledge Tracer under these extreme conditions, we constructed a dataset of multi-domain prompts that explicitly instruct the model to cross domain boundaries during generation. For this ablation study, we randomly sample 14 prompts from this dataset.

Example prompt:

> *"Follow these instructions carefully. Step 1: Write a paragraph explaining GPUs (Graphics Processing Units). Include: what they are, how they differ from CPUs, what they're used for, and their architecture. Step 2: After finishing the GPU explanation, write a completely separate paragraph about the ten most popular war movies around the world. Begin now with Step 1."*

This prompt forces the model to first generate technical computing content (activating neurons related to hardware, architectures, parallelism) and then abruptly transition to cinematic/cultural content (activating entirely different neurons related to film, history, narrative). With a short input ($\sim$50 tokens of instructions), the initial mask has minimal context to work with—making this a deliberately challenging scenario for pruned generation.

We deliberately stress-test the Knowledge Tracer under these adversarial domain-switching conditions to identify the optimal parameter configuration. The parameters identified through this ablation are then used in the main paper's experiments on standard benchmarks, where longer single-domain inputs make the quality - efficiency trade-off more favorable.

Setup. LLaMA-3.1-8B-Instruct, 50% structured FFN sparsity, adaptive per-layer allocation.

Parameters under study. The Knowledge Tracer is governed by five parameters that control different stages of the detect-and-remask pipeline. We define each here so the ablation tables that follow are self-contained.

- **τ (initial scoring window).** Before any pruning occurs, the model generates τ tokens using the full (unpruned) FFN. During this phase, neuron activations are accumulated to identify the most important neurons. At the end of the τ-token window, the top-k neurons per layer are selected. A larger τ gives the model more context to establish a stable domain space, yielding a higher-quality mask, but it also means more tokens are generated without any pruning speedup.

Figure 9: Cosine similarity trajectories under different pruning and release strategies.

- **τ_w (drift-check window).** After the mask is applied, the Knowledge Tracer periodically checks whether the model's attention has drifted away from the semantic space for which the mask was built. It does this by averaging the cosine similarity of the last τ_w attention output projections against the reference centroid computed at mask time. A larger τ_w produces a more stable cosine estimate (less noise from individual tokens), but checks less frequently—a domain transition mid-window may go undetected until the window boundary.
- **m / k (threshold parameter).** Each drift check compares the current window's cosine similarity against a threshold. In non-parametric mode, the threshold is $\min(\text{reference cosines}) - m$, where m is a margin below the worst-case cosine seen during mask construction. In Gaussian mode, the threshold is $\mu - \text{clamp}(k \cdot \sigma,\ 0.05,\ 0.15)$, extending Eq. [Place holder] from the paper with practical bounds. Tighter thresholds (small m, small k) trigger more aggressively.
- **c_0 (counter threshold).** Not every below-threshold event should trigger an expensive re-mask. The shift pressure counter c_t increments when the cosine drops below the threshold and decrements otherwise. Re-masking triggers only when $c_t \geq c_0$. Higher c_0 requires more consecutive drift events before acting, filtering transient noise but potentially delaying detection of genuine drift.
- **τ_r (re-scoring window).** When drift triggers a re-mask, the model reverts to full (unpruned) computation and generates τ_r tokens to re-score neurons—building a fresh mask from the current context. A larger τ_r produces a higher-quality re-mask (more activation data) but consumes more of the generation budget in unpruned mode. After τ_r tokens, a new mask is applied, and pruned generation resumes.

Quality criteria. Each generated output is independently rated by two human evaluators with expertise in LLM behavior and by an LLM-as-a-judge (Claude Opus). The final rating is determined by majority vote:

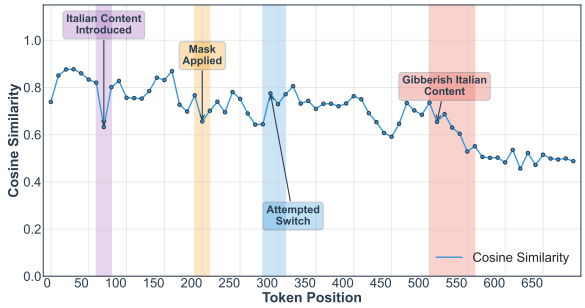
- **Coherent.** Addresses all requested topics. No degenerate repetition. Smooth transitions.
- **Degraded.** Covers some topics. Noticeable repetition or garbled words. Largely readable.
- **Failed.** Degenerate repetition loop, fails to transition, or incoherent majority.

Metrics. *Retrigger counts* are totals across all 140 prompts (average per prompt = total / 140). *Efficiency* is the fraction of generated tokens in pruned mode: efficiency = (total tokens − initial scoring − retrigger scoring)/total tokens.

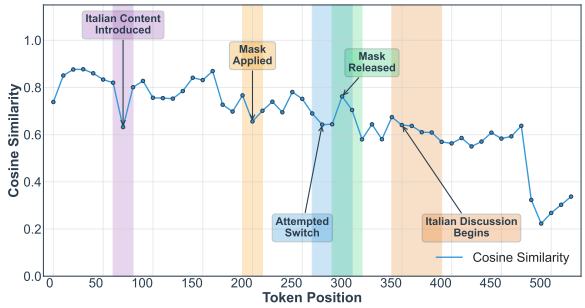
Scope. 60+ experiments, 840+ total generations, single AMD Instinct MI300X GPU.

F.2 Layer-Domain Structure

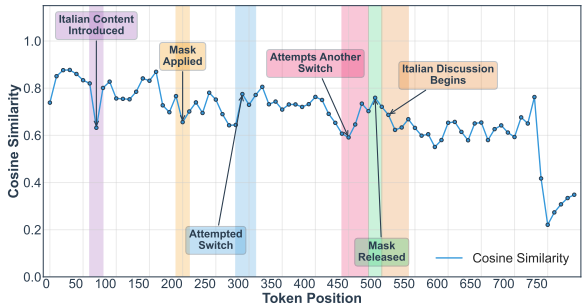
Activation energy profile. The three-regime depth structure (flat L_2–L_{12}, moderate L_{13}–L_{26}, peaky L_{27}–L_{31}) is established in Table 8. The key consequence for domain sensitivity is that flat layers have a dense cluster of neurons near the top-k boundary, so small domain-induced activation shifts produce large rank reshuffling, while peaky layers have a stable dominant elite that is largely insensitive to domain shift. The cross-domain overlap



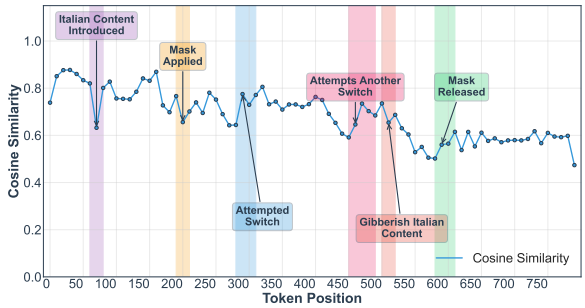
(a) Model pruned at 100th generated token



(b) Pruned neurons released after the 200th generated token



(c) Pruned neurons released after the 400th generated token



(d) Pruned neurons released after the 500th generated token

Figure 10: Cosine similarity trajectories under different pruning strategies. Each subplot shows the effect of neuron pruning and release timing on the model's internal representations.

results below quantify this directly.

Cross-domain mask overlap. For each of the 95 prompts we build the DART pruning mask. Across all 4,275 cross-domain prompt pairs and 190 within-domain pairs we compute Jaccard similarity and the neuron-level difference count.

The full per-layer table (all 32 layers) follows the V-shape summarized above: overlap is high in early layers (L_0: Jaccard 0.84, owing to high DART retention), reaches its minimum in mid layers (L_{24}: Jaccard 0.36, $\approx$3,000 neurons differ per domain pair), then recovers in late layers (L_{31}: 0.64) where a dominant structural elite is preserved. Cross-domain Jaccard is consistently lower than within-domain Jaccard at every layer, with the gap widest in mid layers (L_5: 0.482 vs. 0.528), confirming the differences are driven by domain shift rather than random variance.

At a single domain transition, the correct response is to replace $\approx$2,400–3,000 neurons *per layer* in the L_5–L_{26} range. The consistency-gated incremental strategy (Section 3.3) achieves $\approx$78 neuron swaps per prompt across 450 tokens—off by 30–40$\times$ in a single layer. Full re-masking from a fresh τ_r-token scoring window is the only viable recovery mechanism.

F.3 Drift Threshold Design

The Knowledge Tracer is governed by five parameters: τ (initial scoring window), τ_w (drift-check window), τ_r (re-scoring window), c_0 (counter threshold), and the threshold mode and scale. All ablations use LLaMA-3.1-8B-Instruct at 50% sparsity, τ=50, 500 generated tokens, and 100 multi-domain prompts unless noted. Quality ratings use Claude Opus 4.6 as judge: Coherent (all topics addressed, no degenerate repetition, smooth transitions), Degraded (partially readable, noticeable repetition), and Failed (degenerate loop or incoherent majority).

Gaussian vs. min-cosine threshold. We compile all experiments relevant for this section in Table 21. Paper Eq. 7 defines drift as $a_{t',\tau} - \mu_{t,T} \leq -\delta\,\sigma_{t,T}$. We generalize with practical bounds:

$$\theta = \mu - \text{clamp}(k\,\sigma,\ 0.05,\ 0.15) \qquad (7)$$

The clamp prevents trivially tight thresholds when σ is very small (homogeneous scoring window) and prevents the threshold from dropping below the natural cosine floor when σ is large. In the typical range $\sigma \in [0.02, 0.05]$ the clamp is inactive and the formula reduces to Eq. 7.

As a distribution-free alternative, we test: $\theta = \min(\text{reference cosines}) - m$, which requires no

Table 20: Per-layer cross-domain mask overlap (4,275 cross-domain pairs; within-domain Jaccard from 190 pairs shown for comparison).

Layer	Jaccard (cross)	Jaccard (within)	Neurons differ
L_0	0.842 ± 0.004	0.850 ± 0.004	1103 ± 31
L_5	0.482 ± 0.014	0.528 ± 0.032	2716 ± 95
L_6	0.444 ± 0.016	0.491 ± 0.034	2546 ± 101
L_{21}	0.394 ± 0.014	0.442 ± 0.025	2774 ± 87
L_{24}	0.357 ± 0.012	0.398 ± 0.025	3024 ± 80
L_{31}	0.642 ± 0.021	0.697 ± 0.020	1392 ± 98

distributional assumption and is valid even with a single reference window.

Both modes exhibit a sharp sensitivity cliff: quality collapses when $m{>}0.01$ (non-parametric) or $k{>}2$ (Gaussian). We select the Gaussian mode for two reasons.

Better detection at default settings. At $\tau_w{=}5$, $c_0{=}1$, $\tau_r{=}30$, Gaussian $k{=}1$ achieves 9/14 coherent with only 1 catastrophic failure, versus 7/14 coherent with 6 failures for non-parametric $m{=}0.01$. The Gaussian threshold adapts per-prompt (μ, σ vary with the reference distribution), setting a tighter threshold for prompts with low cosine variance and a looser one for high variance—catching drift earlier without excess false alarms.

Robustness across parameter variations. Gaussian $k{=}2$ holds at 8/14 coherent across four different (c_0, τ_r) configurations without re-tuning k. Non-parametric $m{=}0.01$ swings from 11/14 coherent (at $\tau_w{=}20$) to 2/14 (at $c_0{=}2$) to 5/14 (at $\tau_w{=}20$, $\tau_r{=}10$). The margin m is an absolute offset tied to one model's cosine scale; Gaussian k is expressed in units of the reference distribution, making it invariant to the model's absolute cosine dynamics. Practitioners can fix $k{=}1$–2 and adjust only τ_w and τ_r, rather than jointly searching all parameters for each new deployment. The cross-architecture experiments (LLaMA, Qwen3, Mistral) use the same drift parameters without per-model re-tuning.

F.4 Effect of Initial Scoring Window

A larger τ allows the model to establish a richer domain space—the attention mechanism processes more context, activation patterns stabilize, and the resulting mask captures a more representative neuron subset. At $\tau{=}100$, the model has typically completed one full introductory paragraph, giving the attention mechanism time to settle into a stable pattern.

Caveat. At $\tau{=}200$ with 500 total tokens, domain transitions occur during the 200 unpruned scoring tokens. Concrete example—prompt8 ("Write about materials science, then street festivals in Morocco"): the complete materials science section is written unpruned; the pruned model only continues within Morocco. The drift mechanism never fires because there is no drift in the pruned phase.

All subsequent ablations use $\tau{=}50$ to stress-test drift detection where domain transitions must occur during pruned generation.

Counter threshold (c_0). The counter c_t increments when cosine falls below threshold and decrements otherwise; re-masking triggers when $c_t \geq c_0$. At $c_0{=}2$, the τ_w-token delay of requiring a confirmation event is enough for a pruned model to enter an irreversible repetition loop where internal representations stabilize, cosine stops drifting, and the detection window closes. The table shows this most starkly at $\tau_w{=}3$ with $c_0{=}1.67$: noise prevents consecutive below-threshold windows from accumulating, yielding 0/14 coherent. At $\tau_w{=}20$ (reliable signal, rare false positives), $c_0{=}1.5$ still drops from 11/14 to 6/14 which shows that the cost of missing one genuine drift event far exceeds the cost of an occasional false retrigger (only $\tau_r{=}30$ tokens). Fix $c_0{=}1$.

Drift-check window (τ_w). As we observe in Table 23, quality peaks at $\tau_w{=}20$ (79% coherent) then drops: windows ≥ 30 starve the reference distribution to a single window, leaving no statistical basis for the threshold. The hard bound is $\tau_w \leq \tau/2$; quality peaks near $\tau_w = \tau/2.5$. The recommended heuristic: start at $\tau_w{=}\tau/2.5$; increase if retriggers are excessive ($>$ 10/prompt); decrease if drift is missed without retriggers.

Re-scoring window (τ_r). As seen in Table 6, at $\tau_w{=}5$, quality is flat across a $10\times$ range of τ_r (10–100)—detection noise dominates and re-mask quality is irrelevant. At $\tau_w{=}20$ (reliable detection), τ_r matters: $\tau_r{=}10$ produces masks too weak to sustain generation (151 retriggers, excessive

Table 21: Threshold sweep (τ=50, c_0=1, τ_r=30, τ_w=5 unless otherwise noted). Retriggers denote the total number of remasking events across 100 prompts.

Mode	Threshold	Coherent	Degraded	Failed	Retrigs	Notes
Static	—	21%	29%	50%	0	—
Non-param	m=0.01	50%	7%	43%	55	—
Non-param	m=0.03	29%	29%	43%	37	—
Non-param	m=0.05	14%	29%	57%	23	—
Non-param	m=0.01	14%	36%	50%	18	c_0=2
Non-param	m=0.01	36%	36%	29%	32	τ_w=3
Non-param	m=0.01	64%	14%	21%	106	τ_w=10
Non-param	m=0.01	**79%**	14%	**7%**	97	τ_w=20
Gaussian	k=1.0	**64%**	29%	**7%**	114	—
Gaussian	k=1.0	36%	36%	29%	65	c_0=2
Gaussian	k=2.0	57%	29%	14%	97	—
Gaussian	k=2.0	57%	36%	7%	97	c_0=0.6
Gaussian	k=2.0	29%	50%	21%	48	c_0=2
Gaussian	k=3.0	14%	36%	50%	55	—
Gaussian	k=5.0	7%	36%	57%	27	—

Table 22: Effect of the initial scoring window τ on generation quality.

τ	Coherent	Finding
50	6/14	Weak mask—frequent degenerate loops
100	8/14	Model establishes one domain—mask becomes meaningful
150	10/14	Near-optimal
200	11/14	Best quality—but see caveat

cycling); τ_r=30 is optimal. Unlike τ_w and c_0, τ_r degrades gracefully—non-optimal values shift outputs from coherent to degraded rather than failed.

Two practical operating points: (a) *latency-sensitive*: τ_w=5, τ_r=30—66% pruned efficiency, 50% coherent; (b) *quality-sensitive*: τ_w=20, τ_r=30—48% efficiency, 79% coherent, 7% failed.

G Repetition-Triggered KV Cache Rollback

G.1 Motivation

Cosine-based drift detection monitors the attention output distribution and triggers a remask when it diverges from the reference established during scoring. However, this mechanism has a blind spot: **within-domain degenerate loops**. When the pruned model enters a repetition cycle - repeating the same phrase or sentence -the attention representations remain stable (the model is producing the same content, so the cosine does not shift). The drift detector sees no anomaly, yet the output is clearly broken.

Repetition-triggered rollback addresses this by monitoring the generated token sequence directly. When a degenerate pattern is detected, the mechanism: (1) identifies the first occurrence of the repeating pattern, (2) truncates the generated tokens and KV cache to that point, and (3) resets the Pruner to trigger a fresh re-scoring phase.

G.2 N-gram Detection Thresholds

We scan the last 60 generated tokens every 10 tokens using four n-gram sizes with per-size repeat thresholds:

The thresholds are tuned to avoid false positives from natural language repetition while catching genuine degenerate loops. Below we justify each threshold with examples observed in our experiments.

3-gram × 4 repeats. A 3-token phrase appearing 4 or more times within 60 tokens is almost certainly degenerate. Natural language rarely repeats a specific trigram this frequently—common function-word trigrams like "of the" or "in the" may appear 2–3 times in 60 tokens but not 4. Example from CNN/DailyMail at 50% sparsity (static mask):

> *"The doctor's eyes were focused on the screen, when suddenly a notification popped. The patient's eyes were being scanned by a computer vision system.* ***The doctor's eyes were focused on the screen, when suddenly a notification popped.***

Table 23: Effect of τ_w (non-parametric m=0.01, c_0=1, τ_r=30).

τ_w	Coherent	Degraded	Failed	Retrigs	Ref. windows
3	36%	36%	29%	32	16
5	50%	21%	29%	55	10
10	64%	14%	21%	106	5
20	**79%**	**14%**	**7%**	97	2
30	57%	21%	21%	72	1
50	43%	50%	7%	64	1

Table 24: Effect of τ_r. Efficiency = fraction of generated tokens in pruned mode.

τ_r	Coherent	Degraded	Failed	Retrigs	Efficiency	τ_w
10	57%	7%	36%	228	57%	5
30	50%	7%	43%	55	66%	5
50	50%	7%	43%	38	63%	5
100	50%	14%	36%	20	61%	5
10	36%	43%	21%	151	68%	20
30	**79%**	**14%**	**7%**	97	48%	20
50	71%	14%	14%	67	42%	20

> *The patient's eyes were being scanned by a computer vision system. The doctor's eyes were focused on the screen, when sudden..."*

The trigram "**the screen, when**" repeats 4 times. The 3-gram $\times$ 4 threshold catches this loop at the earliest detectable point.

5-gram $\times$ 3 repeats. A 5-token phrase repeating 3 times is unambiguously degenerate—no natural English sentence fragment of 5 words appears three times in 60 tokens. Example from SAMSum (dialogue summarization):

> *"Amanda suggested asking Larry. Hannah was hesitant.* ***Amanda suggested asking Larry. Hannah was hesitant. Amanda suggested asking Larry.*** *Hannah was hesitant..."*

The 5-gram "**Amanda suggested asking Larry. Hannah**" repeats 3 times. This threshold catches sentence-fragment loops that the 3-gram detector might miss if individual trigrams do not reach 4 repeats.

8-gram $\times$ 2 repeats. An 8-token phrase appearing twice in 60 tokens is already highly suspicious. Natural language does not produce identical 8-word sequences unless they are formulaic headings or citations. Example from GovReport:

> *"The Unified Facilities Criteria also include standards for the design and construction of facilities in areas prone to natural hazards, such as floods, landslides, and earthquakes. The standards are designed to ensure that DOD facilities are designed to withstand natural disasters.* ***The Unified Facilities Criteria also include standards for the design and construction of facilities in areas prone to natural hazards****, such as floods, landslations, and earthquakes..."*

The 8-gram "**Criteria also include standards for the design and construction**" repeats verbatim. Note the model also introduces the typo "landslations" on the second occurrence—a sign of degrading mask quality. The 8-gram $\times$ 2 threshold catches full-clause repetition before it spirals.

12-gram $\times$ 2 repeats. A 12-token identical sequence appearing twice catches full paragraph-level loops where the model copies entire sentences. Example from PubMed:

> *"****the hads-a subscale was used to assess anxiety symptoms. the hads-a subscale was used to assess anxiety symptoms.*** *the hads-a subscale was used to assess anxiety symptoms..."*

This 12-token phrase repeats identically, forming a multi-sentence loop. The 12-gram detector catches these even if shorter n-grams are fragmented across sentence boundaries.

G.3 Threshold Justification

The thresholds follow a specificity-sensitivity trade-off: shorter n-grams require more repeats (higher threshold) to avoid false positives from natural topic-keyword repetition, while longer n-grams need fewer repeats because verbatim repetition of 8+ tokens is inherently unnatural.

We verified false positive rates on the dense (unpruned) model. Our thresholds flag 0% of dense outputs as degenerate at the 5-gram $\times$ 3, 8-gram $\times$ 2, and 12-gram $\times$ 2 levels. The 3-gram $\times$ 4 level

Table 25: N-gram detection thresholds for rollback. Shorter n-grams require more repeats to avoid false positives from natural topic-keyword repetition.

N-gram size	Min repeats	Catches
3 tokens	4 repeats	Short phrase loops
5 tokens	3 repeats	Sentence fragment loops
8 tokens	2 repeats	Full clause / sentence repetition
12 tokens	2 repeats	Multi-sentence repetition

produces some false positives on dense (topical trigrams in long outputs), which is why we check n-gram sizes in ascending order and report only the first match.

G.4 KV Cache Rollback Procedure

When repetition is detected at generated token position t with the repeating pattern first occurring at position p:

1. **Truncate tokens**: `all_tokens = all_tokens[:, :prompt_len + p]`
2. **Truncate KV cache**: Slice all layers' key/value tensors to prompt_len $+ p - 1$ along the sequence dimension, maintaining the invariant $|\text{KV}| = |\text{all_tokens}| - 1$.
3. **Reset NeuronDefuser**: Call `reset_state()`, which sets currIteration $= 0$ and maskingStep $= \tau_r$, clearing all pruning masks and starting a fresh re-scoring phase.
4. **Resume generation**: The generation loop continues from position p with the model running unpruned for τ_r tokens while accumulating activation statistics, then rebuilds masks.

Let t^* denote the rollback position returned by the detector. To prevent cascading degeneration, we discard all tokens generated after t^* and truncate the KV cache accordingly:

$$\mathbf{g} \leftarrow \mathbf{g}_{1:t^*}, \qquad \mathbf{K}^{(l)}, \mathbf{V}^{(l)} \leftarrow \mathbf{K}^{(l)}_{:t^*-1,:}, \mathbf{V}^{(l)}_{:t^*-1,:}, \quad \forall l \in [L] \tag{8}$$

maintaining the invariant $|\text{KV}^{(l)}| = |\mathbf{g}| - 1$ so that generation can resume directly via $\text{model}(\mathbf{g}_{-1}, \text{past_key_values} = \mathbf{KV})$.

G.5 Suppression During Re-scoring

Rollback detection is suppressed when currIteration $\leq$ maskingStep—i.e., during the re-scoring window after any reset. This prevents a pathological cycle where: rollback fires $\rightarrow$ reset $\rightarrow$ re-scoring begins $\rightarrow$ model generates freely during re-scoring $\rightarrow$ rollback fires on re-scoring output $\rightarrow$ reset $\rightarrow$ infinite loop. By suppressing detection during re-scoring, the model has the full τ_r window to generate uninhibited before any checks resume.

H LLM-as-a-Judge

I Impact of multi-batching

Table 27 shows that batch-averaged masking incurs only a 2% accuracy drop from batch size 32 to 64, while DART continues to outperform all competitive baselines across batch sizes. This shared masking strategy aligns naturally with domain-aware request routing, an active systems problem studied in PASCAL (Cho et al., 2026) and SpotServe (Miao et al., 2024), where grouping semantically similar queries improves hardware utilization. The benefit is amplified in MoE architectures, where heterogeneous batches trigger expensive expert loading that homogeneous grouping avoids. Concurrent work on probe-based pruning (Le et al., 2025) further validates this direction, demonstrating that a small representative subset of a batch suffices to identify crucial weights.

J Full results over summarization and generation datasets

K Custom prompt dataset

We did not identify any standard benchmark containing prompts that explicitly require multi-topic generation within a single response, which is necessary to evaluate generation under pruning with knowledge tracing. To address this gap, we designed a set of prompt templates with placeholders that can be instantiated using different topic variables, enabling systematic construction of diverse multi-domain prompts. By sampling across these placeholders, we generated 500 distinct prompts covering varied topic combinations and transitions.

Table 29 summarizes the templates, their intended evaluation purpose, and the categories of terms used to populate each placeholder. This de-

Table 26: LLM-as-a-Judge evaluation rubric used for multi-topic generation assessment. Each criterion is scored independently on a discrete scale from 0.00 to 1.00, and the final rating is computed as the mean across all five criteria. Overall ratings are categorized as *GOOD* (≥ 0.80), *PARTIAL* (0.40–0.79), and *BAD* (< 0.40).

Criterion	1.00	0.75	0.50	0.25	0.00
Coverage	Both topics addressed with substantive depth	Both topics addressed but one is shallow or truncated	Only one topic covered substantively; second barely started	One topic partially covered; second absent	Neither topic covered or output is off-topic
Coherence	Fluent, grammatically sound, logically connected throughout	Mostly fluent with occasional awkward phrasing or circular reasoning	Readable but contains noticeable repetitive phrasing or shallow content	Partially readable but significant portions are repetitive or disjointed	Degenerate repetition loop, gibberish, or incoherent output
Specificity	Names real entities with correct attributes	Names some real entities but with minor errors	Mix of real and generic content; some hallucinated entity names	Mostly generic content with few or no real entities	Entirely generic, fabricated entities, or no domain-specific content
Transition	Natural, smooth transition between topics with explicit marker	Transition present but slightly abrupt or formulaic	Transition attempted but second topic bleeds into first topic's content	Transition absent; second topic begins without separation	Never reaches second topic; generation ends or loops before transition
Non-repetition	No phrase or sentence repeated verbatim	A phrase appears twice but text remains diverse overall	A phrase or sentence structure appears 3 times; noticeable but not dominant	Repetitive patterns dominate but some original content remains	Degenerate loop with repeated phrases cycling indefinitely

Table 27: Performance comparison at 50% sparsity across different batch sizes.

Method	Sparsity	PIQA	HellaS	WinoG	ARC-E	ARC-C	OBQA	Avg
Dart (b=1)	50%	75.50	66.50	70.50	74.50	49.50	44.50	63.50
Dart (b=16)	50%	74.80	63.80	71.40	66.00	45.40	37.00	59.73
Dart (b=32)	50%	74.20	62.00	69.60	64.20	42.00	34.80	57.80
Dart (b=64)	50%	72.00	61.00	65.80	61.80	38.40	31.60	55.10
Wanda-SP	50%	70.80	59.60	61.90	61.50	36.40	36.80	54.50
FLAP	50%	60.00	42.00	58.10	39.60	26.00	31.00	42.78
CFSP	50%	71.20	61.20	65.20	62.50	37.30	36.40	55.63

sign ensures controlled variation in domain shifts while maintaining consistent structural demands on the model. The Sankey diagram in fig. 11 visualizes the distribution of generated prompts across template types and topic categories, confirming broad coverage of cross-domain transitions.

Table 28: Automated and LLM-as-a-Judge evaluation across summarization and generation benchmarks. Drift-aware pruning consistently improves quality and robustness over static pruning while reducing effective computation.

Configuration		Automated Metrics				LLM-as-Judge					Efficiency	
Dataset	Mode	R1	R2	RL	BLEU	Cov	Coh	Spec	Trans	NRep	Judge	Eff.%
cnn_dailymail	drift	.183	.067	.134	.024	.75	.85	.80	.75	.90	.81	85.4
	static	.167	.059	.121	.022	.70	.80	.75	.70	.80	.75	100.0
xsum	drift	.158	.033	.111	.006	.65	.85	.70	.65	.85	.74	85.2
	static	.138	.031	.103	.005	.60	.80	.65	.60	.80	.69	100.0
samsum	drift	.247	.067	.186	.013	.75	.90	.50	.60	.85	.72	89.9
	static	.241	.070	.189	.021	.70	.85	.45	.55	.80	.67	100.0
dialogsum	drift	.196	.048	.152	.008	.80	.90	.60	.65	.85	.76	95.4
	static	.183	.041	.136	.009	.70	.85	.50	.60	.75	.68	100.0
pubmed	drift	.131	.025	.093	.006	.15	.45	.40	.25	.55	.36	100.0
	static	.094	.016	.076	.003	.10	.30	.30	.20	.25	.23	100.0
billsum	drift	.202	.047	.135	.017	.10	.40	.25	.15	.65	.31	100.0
	static	.188	.046	.128	.019	.10	.35	.20	.15	.55	.27	100.0
gov_report	drift	.073	.013	.076	.004	.15	.55	.45	.30	.60	.41	97.0
	static	.059	.010	.066	.002	.10	.40	.40	.25	.45	.32	100.0
multi_news	drift	.133	.020	.085	.005	.15	.35	.45	.20	.40	.31	100.0
	static	.113	.016	.075	.004	.10	.30	.40	.15	.35	.26	100.0
arxiv	drift	.146	.035	.105	.011	.15	.40	.30	.20	.55	.32	100.0
	static	.124	.029	.099	.007	.10	.30	.25	.15	.40	.24	100.0
booksum	drift	.175	.021	.111	.007	.10	.55	.35	.20	.75	.39	98.7
	static	.146	.020	.104	.006	.10	.50	.30	.20	.70	.36	100.0
alpaca_eval	dense	—	—	—	—	.90	.95	.85	.85	.95	.90	0.0
	drift	.849	—	.830	.763	.85	.90	.80	.80	.85	.84	79.3
	static	.444	—	.388	.228	.70	.75	.60	.65	.70	.68	100.0
writingprompts	dense	—	—	—	—	.40	.45	.35	.30	.40	.38	0.0
	drift	.887	—	.875	.850	.35	.40	.30	.25	.35	.33	38.4
	static	.652	—	.629	.550	.30	.35	.25	.20	.25	.27	100.0
openwebtext	dense	—	—	—	—	.85	.90	.80	.80	.85	.84	0.0
	drift	.964	—	.963	.910	.80	.85	.75	.75	.80	.79	88.8
	static	.642	—	.617	.455	.75	.80	.65	.70	.70	.72	100.0

Table 29: Intent and Prompt Templates.

Intent	Prompt Template	Slot Values
Explain and Contrast	`Write about {SUBJECT} - what it is, how it works, and typical uses. Then separately write about {CULTURE} in {LOCATION}, covering traditions, variations, and social context.`	**SUBJECT:** Technology, Science **CULTURE:** CulturalPractice **LOCATION:** Region, Country
Compare and Assess Impact	`Compare {A} and {B}, focusing on principles, performance, and trade-offs. Then explain how these differences affect {IMPACT} for {GROUP}.`	**A:** Technology, Science, EconomicConcept **B:** Technology, Science, EconomicConcept **IMPACT:** ImpactDomain **GROUP:** PopulationGroup
Apply to Sector with Policy	`Write about how {CONCEPT} applies to {SECTOR}, including workflow changes, benefits, and risks. Then discuss policy measures such as {POLICY} that shape adoption.`	**CONCEPT:** Technology, Science **SECTOR:** Sector **POLICY:** PolicyInstrument
Analyze Cause, Effect, and Mitigation	`Describe how {ISSUE} affects {IMPACT} in {LOCATION}, covering mechanisms and evidence. Then propose practical mitigation strategies and trade-offs.`	**ISSUE:** EnvironmentIssue, HealthCondition, EthicalIssue **IMPACT:** ImpactDomain **LOCATION:** Country, City, Region, RuralArea
Debate with Role-Play	`Create a dialogue between a {ROLE_A} and a {ROLE_B} discussing {TOPIC}. Start with the {ROLE_A}'s perspective. End with a summary addressing {IMPACT}.`	**ROLE_A:** ActorRole **ROLE_B:** ActorRole **TOPIC:** EthicalIssue, Technology, PolicyInstrument **IMPACT:** ImpactDomain
Trace Development and Ethics	`Trace the scientific development of {DISCOVERY}. Then explain its commercialization in {SECTOR}. Finally, analyze ethical concerns focusing on {ETHICS}.`	**DISCOVERY:** Science **SECTOR:** Sector **ETHICS:** EthicalIssue
Narrative with Technical Explanation	`Write a short narrative about {EVENT_GROUP} in {LOCATION}. Then add a factual explanation of how {TECH} enabled or shaped this experience.`	**EVENT_GROUP:** PopulationGroup **TECH:** Technology **LOCATION:** Country, City, Region, RuralArea
Policy Brief with Enforcement	`Draft a policy brief addressing {POLICY} for {SYSTEM} in {LOCATION}. Then detail technical enforcement and auditability mechanisms.`	**POLICY:** PolicyInstrument **SYSTEM:** Technology **LOCATION:** Country, City
Governance and Transparency	`Outline governance principles for {PLATFORM}, accounting for {ETHICS}. Then explain how transparency in ranking and recommendation algorithms supports {IMPACT}.`	**PLATFORM:** Technology **ETHICS:** EthicalIssue **IMPACT:** ImpactDomain
How-To Guide with Sidebar	`Provide a concise how-to guide for {WORKFLOW}. Then add a sidebar describing {SIDEBAR_CULTURE} in {SIDEBAR_LOCATION}.`	**WORKFLOW:** Sector, EducationTheme **SIDEBAR_LOCATION:** Country, Region, City, RuralArea **SIDEBAR_CULTURE:** CulturalPractice
System Explanation with Artistic Reflection	`Explain how {SYSTEM} processes inputs and produces outputs. Then reflect on parallels with {ART}, considering structure, style, and interpretation.`	**SYSTEM:** Technology **ART:** ArtForm
Compare Clinical and Public Health Approaches	`Contrast clinical approaches to {CLINICAL} with population-level interventions like {PUBLIC_HEALTH}. Then discuss how local norms in {COMMUNITY} shape outcomes.`	**CLINICAL:** HealthCondition **PUBLIC_HEALTH:** PolicyInstrument **COMMUNITY:** City, Region, Country

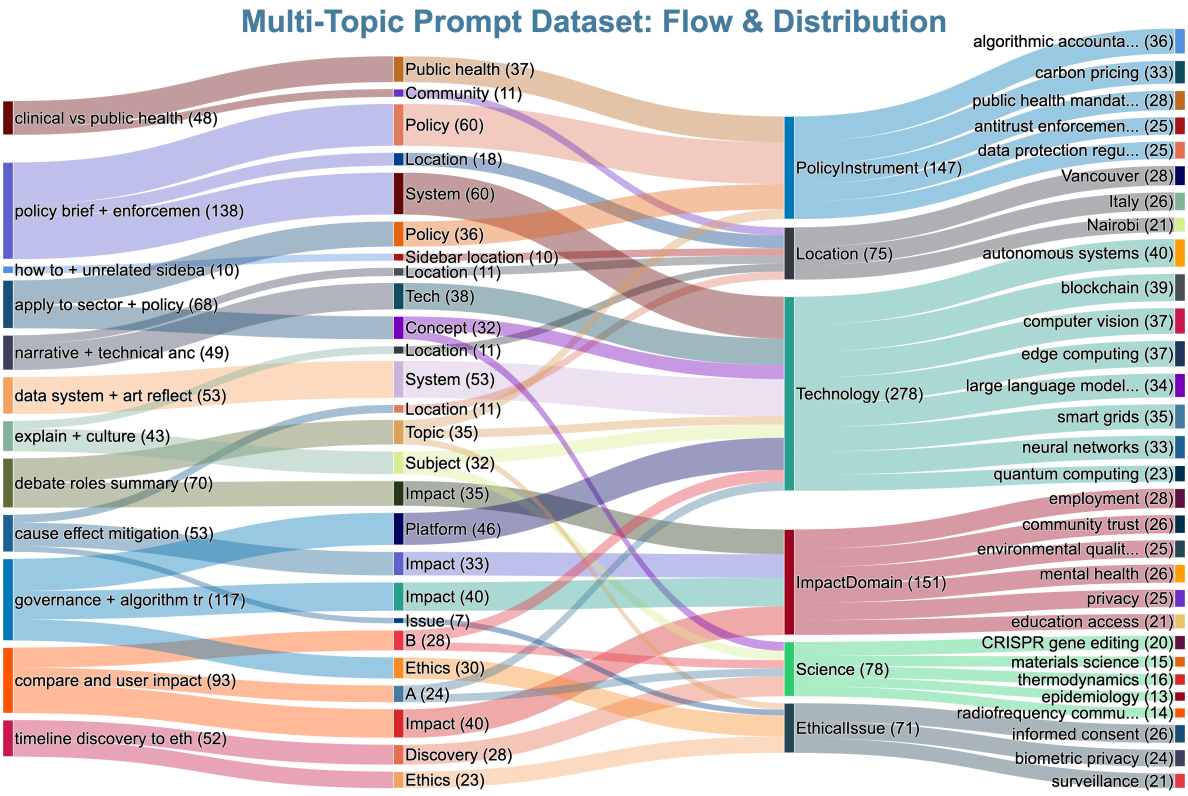


Figure 11: Distribution of generated prompts over the templates and subjects. The number next to each label signifies the number of prompt using the element.