TADP: Structured Pruning of Large Language Models via Temporal Activation Dynamics

Adriano Puglisi, Flavia Monti, Francesco Leotta, Massimo Mecella
Department of Computer, Control, and Management Engineering
Antonio Ruberti, Sapienza University of Rome
Via Ariosto 25, 00185 Rome, Italy
{puglisi,monti,leotta,mecella}@diag.uniroma1.it

Abstract

Structured pruning of large language models provides a practical route to inference acceleration on standard hardware, but most existing methods rely on neuron importance criteria that compress activation behavior into static scalar scores, discarding the sequential structure of activations. We propose TADP (Temporal Activation Dynamics Pruning), a structured pruning method that models neuron importance along the temporal dimension of activations by jointly capturing mean activation, peak activation, positional variance, transition dynamics, and position-weighted response within a unified scoring framework. TADP requires no gradient computation or weight updates and performs pruning in a single forward pass over a calibration dataset, making it zero-shot and computationally lightweight. We evaluate TADP on eight models from the LLaMA, Mistral, Qwen, and Minerva families at 3B and 7B/8B scale, under pruning levels from 10% to 50%, measuring perplexity on WikiText-2 and zero-shot accuracy on ARC, HellaSwag, MMLU, and PIQA.

1 Introduction

The widespread use of large language models (LLMs) (Brown et al., 2020; Touvron et al., 2023a,b) in real-world applications has made inference efficiency a central concern, especially when deployment targets resource-constrained computational platforms such as edge devices. This is particularly relevant in scenarios where low-latency, privacy-preserving, or intermittently connected operation is required, motivating on-device inference. Representative edge applications include smart assistants running on mobile phones for offline query answering and text generation, industrial IoT systems enabling natural-language interaction with machinery diagnostics, autonomous or semi-autonomous vehicles supporting in-vehicle conversational agents for navigation and system status queries, and wearable devices providing context-aware assistance such as real-time transcription or health-related guidance. In all these cases, reliance on cloud-based inference is either impractical or undesirable due to bandwidth limitations, latency constraints, or privacy considerations.

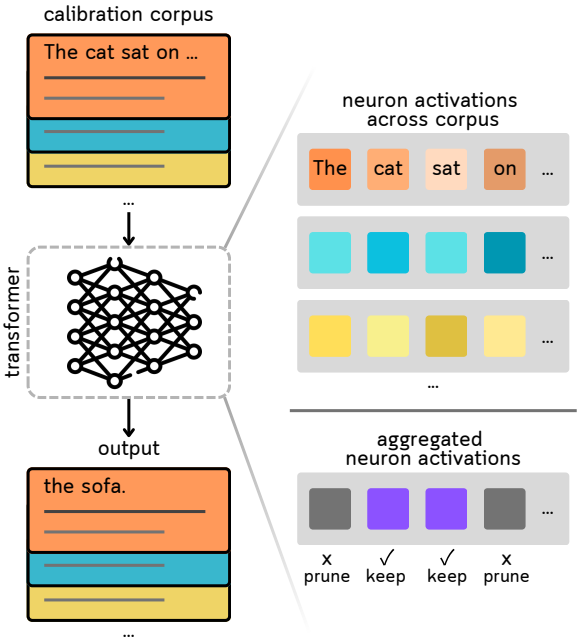


Figure 1: TADP Overview. A calibration corpus is processed through the frozen transformer in a single forward pass, one batch at a time (orange, blue, and yellow rows denote different sequences). For each FFN layer, TADP records the per-position input activations, where color intensity indicates neuron activation strength at each token position. The method preserves the temporal structure across positions and batches to extract aggregated neuron activations. Neurons with the lowest scores are pruned (grey).

Several compression strategies have been proposed to address the computational demands of LLM inference, including quantization, knowledge distillation, and structured pruning. *Quantization* (Frantar et al., 2023; Dettmers et al., 2022; Xiao

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 39222–39239
October 24-29, 2026 ©2026 Association for Computational Linguistics

et al., 2023) reduces numerical precision while preserving the original parameter count, typically relying on low-precision hardware support to translate compression into substantial inference acceleration. Similarly, *knowledge distillation* (Hinton et al., 2015; Gu et al., 2024) relies on training a separate, smaller model, which typically requires substantial data and computational resources. In contrast, *structured pruning* has emerged as a practical solution for reducing computational cost, since it produces dense models with smaller hidden or intermediate dimensions that run faster on standard hardware without requiring specialized sparse kernels. However, structured pruning inevitably alters the internal representations learned during pre-training, making the design of effective neuron importance criteria crucial for preserving downstream performance.

Most existing pruning methods estimate neuron importance through static scalar metrics derived from weight magnitudes, activation statistics, or sensitivity estimates aggregated over a calibration corpus (Sun et al., 2023; Ma et al., 2023; Ashkboos et al., 2024; Frantar and Alistarh, 2023; Puglisi et al., 2026). While effective, these approaches collapse the sequential activation behaviour of neurons into a single aggregated value, ignoring how neuron responses evolve across token positions within a sequence. Transformer activations are highly context-dependent, and neurons often exhibit selective behaviour that emerges only at specific positions or linguistic contexts. As a result, neurons with moderate average activation but strong position-specific responses may be systematically underestimated by static criteria. These observations suggest that neuron importance should account not only for activation magnitude, but also for the temporal dynamics of activations across token positions in autoregressive generation.

In addition, removing structural components from a trained model generally degrades its internal representations, as the correlations learned during pre-training among attention heads, MLP neurons, and surrounding weight matrices are disrupted, potentially leading to a significant drop in predictive accuracy. To mitigate this issue, several recent approaches incorporate a parameter-efficient adaptation stage, most commonly based on LoRA (Hu et al., 2022), after structural reduction. This stage involves gradient-based optimization over a calibration or training corpus, introducing an additional fine-tuning step beyond pruning.

In this paper, we propose Temporal Activation Dynamics Pruning (TADP)[1], a structured pruning method that operates through a single backpropagation-free calibration pass over a small corpus, without gradient computation, weight updates or any kind of post-pruning adaptation. Figure 1 illustrates a high-level overview of the method. Instead of representing neuron importance with a single static score, TADP models the temporal activation behaviour observed across sequence positions through five complementary signals capturing average response, selectivity, positional variability, transition behaviour, and positional contribution. These signals are aggregated into a unified neuron importance score used to identify redundant neurons for removal.

We evaluate TADP on eight decoder-only LLMs from the LLaMA, Mistral, Qwen, and Minerva families at 3B and 7B/8B scales. We focus on these parameter ranges because they represent practical model sizes deployable in real-world applications, where inference efficiency is, as above mentioned, particularly important. Rather than starting from extremely large models and aggressively compressing them to smaller sizes, we investigate whether structured pruning can effectively preserve performance in models that are already relatively compact and widely adopted.

Experiments cover pruning ratios from 10% to 50% across several benchmark datasets, including WikiText-2 and MMLU, measuring perplexity and zero-shot accuracy. The results show that TADP consistently outperforms recent structured pruning baselines across models and pruning regimes.

The remainder of the paper is organized as follows. Section 2 reviews related work on structured pruning for LLMs. Section 3 presents the proposed methodology. Section 4 describes the experimental setup, while Section 5 discusses the results. Finally, Section 6 concludes the paper.

2 Related work

Parameter reduction techniques for pre-trained large language models fall into two broad categories: unstructured and structured pruning. Unstructured pruning identifies individual weights to remove regardless of their position in the network, relying on criteria such as weight magnitude (Han et al., 2015), Hessian-based sensitivity (Frantar and

[1]Code available at https://github.com/Adrianoski/TADP

Alistarh, 2023), or activation statistics. While effective at achieving high pruning ratios, unstructured methods produce irregular weight matrices that require specialized sparse kernels to translate pruning into actual inference speedup, limiting their practical deployability on standard hardware.

A representative of activation-informed unstructured pruning is Wanda (Sun et al., 2023), which scores each weight by the product of its magnitude and the ℓ_2 norm of the corresponding input activations, estimated over a small calibration set requiring neither retraining nor weight update and executed in a single forward pass. Its key insight is that input activation norms, rather than weight magnitudes alone, are a more informative proxy for the contribution of a weight to the layer output.

In contrast, structured pruning removes groups of parameters, such as neurons, channels, or entire Transformer sub-modules, instead of individual weights. By eliminating complete rows, columns, or blocks, structured methods preserve dense computation patterns and are therefore more readily compatible with standard inference hardware and optimized libraries. Methods in this category differ primarily in the granularity removal is performed.

Width pruning operates on subsets of weight matrix rows and columns, reducing the hidden or intermediate dimensions of the network. SliceGPT (Ashkboos et al., 2024) applies orthogonal transformations to slice entire weight matrix rows while preserving model expressivity. LLM-Pruner (Ma et al., 2023) adopts a dependency-aware approach, identifying coupled structures across the network via a graph-based analysis and estimating their importance through first-order gradient information.

Depth pruning instead removes entire Transformer blocks or specific sub-modules. ShortGPT (Men et al., 2025) and sliding window cosine similarity methods assess layer redundancy by measuring the similarity between input and output representations, discarding layers whose contribution is negligible. Gromov et al. (Gromov et al., 2025) showed that many deeper Transformer layers can be removed with limited degradation by pruning blocks with highly similar intermediate representations. BlockPruner (Zhong et al., 2025) and EvoPress (Sieberling et al., 2025) extend this idea to finer granularity, enabling the independent removal of attention and feed-forward sub-modules within each block.

More recent approaches combine width and depth pruning within unified frameworks. 2SSP (Sandri et al., 2025) integrates FFN neuron pruning with iterative attention module removal in a two-stage pipeline, achieving competitive performance with substantially lower pruning runtime than existing state-of-the-art methods.

Orthogonal to the pruning granularity is the design of the importance criterion used to rank parameters. CFSP (Wang et al., 2025) proposes a coarse-to-fine structured pruning strategy based on activation information, combining block-level sparsity allocation with intra-block neuron selection through a single forward calibration pass. Other methods instead rely on learned masks and lightweight adaptation strategies to recover performance after compression. For example, Compresso (Guo et al., 2024) combines structured mask learning with parameter-efficient adaptation, often through low-rank updates, to mitigate pruning-induced degradation while maintaining computational efficiency.

Both Wanda and LLM-Pruner rely on static or aggregated activation statistics that compress neuron behavior into scalar importance scores, overlooking the sequential structure of the input. Closer in terms of underlying intuition is SiRE (Puglisi et al., 2026), which also uses token-level saliency scores derived from input gradients to weight the importance of structural components, anticipating the idea that not all tokens contribute equally to the pruning signal. However, SiRE ultimately compresses this information into a single component-level score, losing the temporal structure of how activations evolve across token positions.

TADP departs from this paradigm by decomposing neuron importance along the temporal dimension of the activation signal, capturing mean response, peak selectivity, positional variability, transition dynamics, and position-weighted contribution within a unified scoring framework. Moreover, TADP requires neither gradient computation nor weight updates, operating entirely within a single backpropagation-free calibration pass.

3 Methodology

Existing structured pruning methods typically estimate neuron importance using static proxies such as weight magnitude or aggregated activation norms. While computationally efficient, these criteria reduce each neuron to a single scalar value, ignoring how its behaviour evolves across sequence positions during inference. This constraint be-

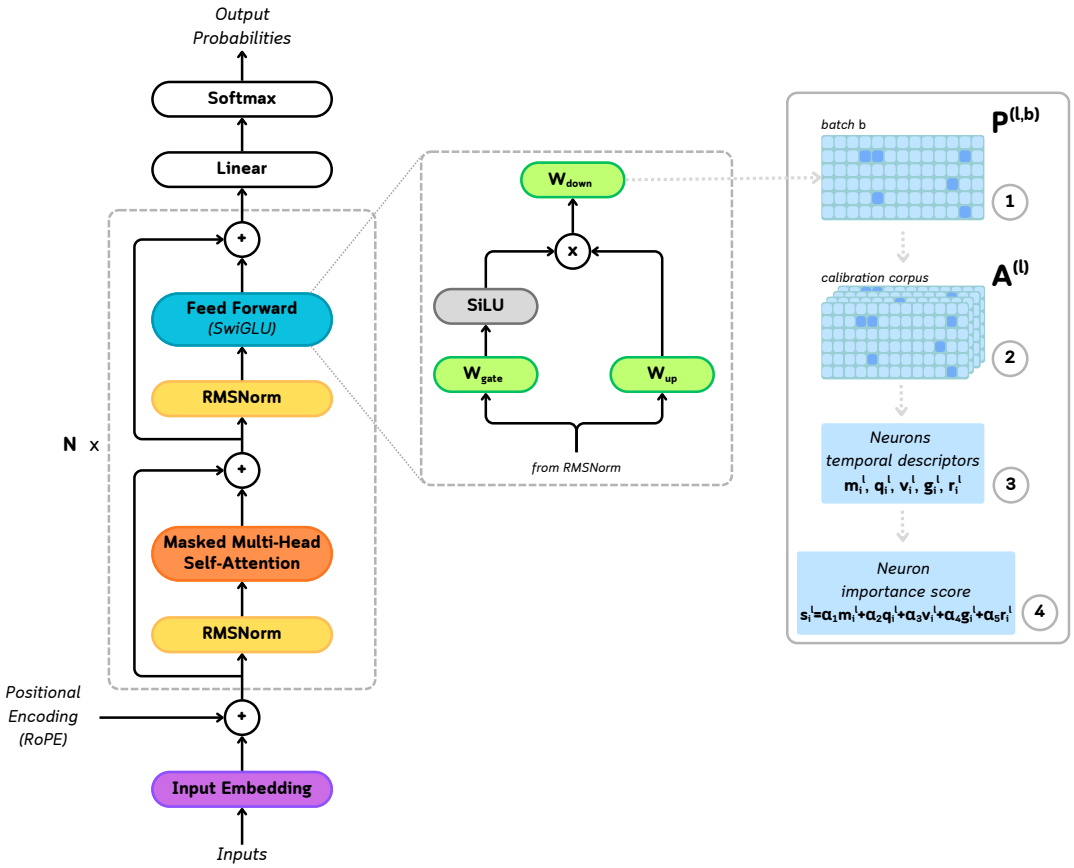


Figure 2: Detailed view of the TADP. The left panel shows the decoder-only transformer architecture. A forward hook registered on the input of W_{down} captures, for each batch b, the per-position ℓ_2 norms of the pre-projection activations, producing the matrix $\mathbf{P}^{(\ell,b)}$ ①. After all batches have been processed, the per-batch matrices are stacked into the three-dimensional tensor $\mathbf{A}^{(\ell)}$, encoding the complete temporal activation profile of every neuron across all calibration sequences and token positions ②. Five temporal descriptors are then extracted from $\mathbf{A}^{(\ell)}$ for each neuron i: mean activation m_i^ℓ, peak activation q_i^ℓ, positional variance v_i^ℓ, transition dynamics g_i^ℓ, and position-weighted response r_i^ℓ ③. These are combined into a final importance score ④, which drives the pruning decision.

comes extremely relevant in transformer-based language models, where neuron activations exhibit strong temporal structure; their magnitude and functional role vary across token positions depending on syntactic boundaries, contextual dependencies, and long-range interactions. Compressing this behaviour into position-agnostic statistics removes information that may be relevant to generation dynamics.

Temporal Activation Dynamics Pruning (TADP) addresses this shortcoming by defining neuron importance as a function of the full temporal activation profile observed during a calibration phase on a dataset. Instead of collapsing activations into a single summary statistic, TADP preserves their structure across sequence positions and derives an importance score from multiple complementary temporal signals.

TADP focuses specifically on decoder-only transformer language models with gated feed-forward networks (Touvron et al., 2023a). In each layer ℓ, the FFN is parameterized by three projection matrices $\mathbf{W}_{\text{gate}}^{(\ell)}, \mathbf{W}_{\text{up}}^{(\ell)} \in \mathbb{R}^{d_{\text{int}} \times d_{\text{model}}}$ and $\mathbf{W}_{\text{down}}^{(\ell)} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{int}}}$. The full pipeline is illustrated in Figure 2.

For each index $i \in \{1, \ldots, d_{\text{int}}\}$, where d_{int} is the intermediate dimension of the FFN, we define a *neuron triplet*, i.e., the vectors comprising the i-th row of $\mathbf{W}_{\text{gate}}^{(\ell)}$, the i-th row of $\mathbf{W}_{\text{up}}^{(\ell)}$, and the i-th column of $\mathbf{W}_{\text{down}}^{(\ell)}$. We refer to this as the i-th neuron triplet of layer ℓ, where the index i identifies a single computational unit along the intermediate dimension shared by all three projection matrices. Removing a neuron triplet is a structurally consistent operation that reduces the intermediate dimension d_{int} uniformly across all three matrices.

Given a pruning ratio $p \in (0, 1)$, the goal is to re-

move, for each layer, a fraction p of neuron triplets while minimizing degradation in language modeling performance. TADP achieves this through a four-stage pipeline: *(i)* a single forward-only calibration pass over a small corpus with frozen parameters and no gradient computation, *(ii)* extraction of layer-wise temporal activation tensors, *(iii)* decomposition of neuron behaviour into temporal response components, and *(iv)* layer-wise ranking and structured pruning based on the resulting scores. The stages are described in detail in the following sections.

3.1 Calibration and temporal signal extraction

TADP collects activation statistics using a single forward pass over a calibration dataset divided into N batches, with all model parameters kept frozen. The calibration data is used only to observe how neurons respond to natural inputs.

We focus on the feed-forward blocks of the transformer, and in particular on the input of the down-projection layer $\mathbf{W}_{\text{down}}^{(\ell)}$, which represents the pre-output activations of the FFN at layer ℓ. During inference, we record these intermediate activations for each layer and each input batch.

For each batch b, this yields a tensor of hidden activations $\mathbf{X}^{(\ell,b)} \in \mathbb{R}^{B\times T\times d_{\text{int}}}$ where B is the batch size and T is the sequence length.

To obtain a compact representation that preserves token-level structure, we aggregate across the batch dimension using the ℓ_2 norm, obtaining $\mathbf{P}^{(\ell,b)} \in \mathbb{R}^{T\times d_{\text{int}}}$.

Intuitively, this step removes dependence on individual samples while preserving how each neuron behaves across positions within a sequence.

After processing all N calibration batches, we stack these observations along the batch dimension, obtaining the final activation tensor $\mathbf{A}^{(\ell)}$. This tensor encodes the empirical temporal behaviour of each neuron across inputs and token positions, and serves as the only input to the pruning stage.

3.2 Neuron selection

The core of TADP lies in how it interprets the activation tensor $\mathbf{A}^{(\ell)}$ collected during calibration. Rather than collapsing it into a single scalar per neuron, TADP treats it as a structured representation of temporal behaviour and derives multiple complementary projections from it.

The stacked tensor $\mathbf{A}^{(\ell)} \in \mathbb{R}^{N\times T\times d_{int}}$ fully encodes the empirical temporal behavior of each neuron. Starting from $\mathbf{A}^{(\ell)}$, we define two summary views over the batch dimension by operating on its constituent batch matrices $\mathbf{P}^{(\ell,b)}$:

$$\bar{\mathbf{A}}^{(\ell)} = \frac{1}{N}\sum_{b=1}^{N}\mathbf{P}^{(\ell,b)}, \quad \hat{\mathbf{A}}^{(\ell)} = \max_{b\in\{1,\dots,N\}}\mathbf{P}^{(\ell,b)}.$$

These two quantities separate typical activation behavior from rare but strong responses, providing complementary views of neuron activity across inputs.

For each neuron i in layer ℓ, TADP extracts the importance score $s_i^{(\ell)}$, which is defined as a weighted sum of five temporal descriptors from $\bar{\mathbf{A}}^{(\ell)}$ and $\hat{\mathbf{A}}^{(\ell)}$:

$$s_i^{(\ell)} = \alpha_1 m_i^{(\ell)} + \alpha_2 q_i^{(\ell)} + \alpha_3 v_i^{(\ell)} + \alpha_4 g_i^{(\ell)} + \alpha_5 r_i^{(\ell)},$$

$$\sum_j \alpha_j = 1.$$

Mean activation:

$$m_i^{(\ell)} = \left\|\bar{\mathbf{A}}_{:,i}^{(\ell)}\right\|,$$

which captures the average magnitude of the neuron's contribution across positions.

Peak activation:

$$q_i^{(\ell)} = \left\|\hat{\mathbf{A}}_{:,i}^{(\ell)}\right\|,$$

which preserves neurons that respond strongly but sparsely across inputs.

Positional variance:

$$v_i^{(\ell)} = \mathrm{Var}_{t\in[T]}\left(\bar{\mathbf{A}}_{t,i}^{(\ell)}\right),$$

which measures how strongly the activation depends on token position t in the sequence T.

Transition dynamics:

$$g_i^{(\ell)} = \frac{1}{T-1}\sum_{t=2}^{T}\left|\bar{\mathbf{A}}_{t,i}^{(\ell)} - \bar{\mathbf{A}}_{t-1,i}^{(\ell)}\right|,$$

which captures local changes in activation across adjacent tokens t in the sequence T.

Position-weighted response:

$$r_i^{(\ell)} = \frac{\sum_{t=1}^{T} w_t \bar{\mathbf{A}}_{t,i}^{(\ell)}}{\sum_{t=1}^{T} w_t}, \quad w_t = 0.5 + \frac{t-1}{T-1},$$

where t is the token and T is entire sequence. This term reflects the asymmetric information flow characteristic of autoregressive decoding.

Neurons with similar average activation can exhibit substantially different behaviours across token positions. A neuron that responds moderately throughout a sequence and one that activates sharply for a small number of context-dependent tokens may receive similar static importance scores, although removing the latter can disrupt specific lexical, syntactic, or semantic transitions. Mean activation captures broad contribution, whereas positional variance and transition dynamics distinguish persistent responses from position-dependent and rapidly changing activation patterns. The position-weighted response accounts for the asymmetric information flow of autoregressive decoding, while peak activation protects highly selective neurons whose contribution would otherwise be diluted by averaging. We therefore assign a larger weight to the peak descriptor, aiming to preserve neurons that respond infrequently but strongly to specific contextual patterns.

This interpretation is supported by the exhaustive ablation in Appendix A, which evaluates 47 score configurations: all 31 non-empty descriptor subsets under uniform weighting and the 16 subsets containing peak activation under peak-heavy weighting. The complete five-component peak-heavy formulation obtains the lowest perplexity, and removing any individual component consistently degrades performance.

3.3 Layer-wise pruning

Neurons are ranked independently within each layer according to $s_i^{(\ell)}$. Given a pruning ratio p, the retained set is:

$$\mathcal{K}^{(\ell)} = \mathrm{TopK}\left(s_i^{(\ell)}, k\right), \quad k = \lfloor (1-p)\, d_{\mathrm{int}} \rfloor .$$

The corresponding rows and columns are removed consistently across the gate, up, and down projections, producing a structurally valid feed-forward block with reduced intermediate dimension and direct compatibility with standard inference frameworks.

4 Experimental setup

Models. We evaluate TADP on eight decoder-only transformer language models spanning four families at two scales. For the LLaMA family (Touvron et al., 2023a) we include LLaMA-3.2-3B and LLaMA-3-8B, for the Qwen family (Yang et al., 2024) we used Qwen2.5-3B and 7B, for Mistral family Ministral-3B (Liu et al., 2026) and Mistral-7B (Jiang et al., 2023), and for the Minerva family (Orlando et al., 2024) Minerva-3B and 7B.

Pruning ratios. We consider five method-specific target levels, denoted as 10%, 20%, 30%, 40%, and 50%. Because the nominal ratio has different meanings across pruning methods, it should not be interpreted directly as the fraction of total model parameters removed. For the budget-matched evaluation, we select model-specific TADP/CFSP FFN pruning ratios that approximate the total-parameter reductions produced by the corresponding pruning configurations.

Calibration set. Importance scores are estimated from a single forward pass over 512 samples drawn from the English split of the C4 dataset (Raffel et al., 2023), specifically from the first shard of the training split, with each sequence truncated or padded to a maximum length of 512 tokens. C4 is a large-scale, cleaned web crawl corpus providing broad lexical and syntactic diversity without domain bias, making it well-suited as a general-purpose calibration corpus for importance estimation. Importantly, any sufficiently large and diverse calibration corpus could be used in principle. We adopt C4 as a standard and widely used choice to ensure comparability and generality of our results.

Evaluation metrics and datasets. Perplexity is evaluated on the WikiText-2 test set (Merity et al., 2016), using sequences of 512 tokens with padding positions excluded from the loss computation. This context length fits within the available hardware budget while remaining sufficiently long to capture meaningful sequential dependencies and consistent with the calibration setting. Zero-shot downstream accuracy is measured on ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021b,a), and PIQA (Bisk et al., 2020).

Related approaches. We compare TADP against several structured pruning methods proposed for large language models, introduced in Section 2, including CFSP (Wang et al., 2025), 2SSP (Sandri et al., 2025), BlockPruner (Zhong et al., 2025), EvoPress (Sieberling et al., 2025), Window-Based (Gromov et al., 2025), ShortGPT (Men et al., 2025), SliceGPT (Ashkboos et al., 2024), and SiRE (Puglisi et al., 2026).

Hardware. All experiments are conducted on a workstation equipped with an NVIDIA GeForce RTX 3090 GPU (24 GB VRAM) and 64 GB of system RAM. All models are loaded in half-precision (`float16`). This setup is representative of a mid-range research workstation and demonstrates that TADP is fully executable without access to multi-GPU infrastructure or data-center hardware.

5 Results and discussion

Tables 1 and 2 report budget-matched results under moderate and high compression. For each model, TADP and CFSP use model-specific FFN pruning ratios that produce the same total-parameter reduction as the corresponding block and depth-pruning configurations. Across the evaluated architectures, the moderate setting removes 18.36 to 21.08% of the total model parameters, while the high-compression setting removes 35.31 to 39.47% depending on the specific model. Because 2SSP defines its compression budget over the main Transformer layers and jointly allocates pruning between FFN channels and complete attention submodules, it removes slightly fewer total parameters in most configurations. SliceGPT instead reduces the shared embedding dimension and is reported only for supported architectures, as detailed in Appendix C.

Overall, TADP achieves the joint-best results in many evaluation settings, with improvements that remain consistent across LLaMA, Qwen, Minerva, and Mistral-based architectures. The advantage becomes more pronounced at 40% pruning, where competing approaches exhibit severe degradation in both perplexity and downstream accuracy, while TADP preserves comparatively stable behavior.

The gains are particularly evident on fluency-oriented metrics, namely WikiText-2 perplexity and HellaSwag accuracy. At 20% pruning, TADP achieves the lowest perplexity in 5 out of 8 evaluated models and consistently ranks among the strongest methods on HellaSwag. At 40% pruning, TADP obtains the best perplexity in 5 out of 8 evaluated configuration. This behavior is consistent with the design motivation of TADP, which prioritizes components associated with temporally dynamic activations and therefore appears better aligned with preserving contextual continuity and local token dependencies during generation.

At moderate compression rates, several baselines remain competitive on individual tasks. In particular, CFSP occasionally achieves slightly stronger results, while some methods remain comparable on ARC under 20% pruning. However, the gap widens under aggressive compression. Depth-oriented pruning strategies such as BlockPruner, Window-Based, and ShortGPT frequently produce perplexity values in the hundreds or thousands at 40% pruning, indicating substantial deterioration of language modeling capability. Width-based approaches degrade more gracefully, but still exhibit large performance losses.

SiRE remains moderately competitive at 20% pruning on several models, but its performance deteriorates sharply at higher pruning levels, with perplexity exceeding 10^2 in multiple settings. This comparison suggests that token-level saliency alone is insufficient to preserve stable language modeling behavior under aggressive compression, and that incorporating temporal activation dynamics provides additional information for identifying components that remain important across sequential contexts.

TADP does not consistently achieve the best performance on MMLU. Unlike perplexity and HellaSwag, which are closely tied to contextual continuity and fluency, MMLU primarily evaluates factual and domain-specific parametric knowledge. The weaker relative gains on this benchmark may indicate that temporally dynamic activations are more strongly associated with contextual reasoning patterns than with the distributed storage of factual associations across MLP parameters.

Ministral-3B-Instruct exhibits substantially higher baseline perplexity on WikiText-2 compared to the other 3B models (92.04 versus 15.97 for LLaMA 3B and 14.58 for Qwen 3B), suggesting a markedly different intrinsic calibration on this benchmark independently of pruning.

Finally, SliceGPT reports results only for LLaMA 8B and Mistral 7B, as the method requires specific structural properties in the weight matrices that are not supported by all evaluated decoder-only architectures.

Pruning and inference efficiency. We complement the performance evaluation with a wall-clock analysis of the cost required to construct the pruned model and the resulting inference latency. Table 3 reports results on LLaMA-3.2-3B for the high-compression configurations used in the budget-matched comparison and Table 4 reports results on LLaMA-3-8B. All measurements were collected on the same hardware and in the same numerical

	LLaMA 3B					Ministral-3B-Instruct					Qwen 3B					Minerva 3B				
	Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy ↑			
Method	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA
Baseline	15.97	45.99	73.61	54.40	77.48	92.04	26.19	37.75	24.60	59.96	14.58	47.27	73.52	65.20	78.94	18.70	29.86	60.81	24.70	75.41
budget-matched 20%																				
TADP (ours)	28.31	33.19	62.97	31.00	74.05	105.93	25.94	36.76	23.57	59.09	22.18	36.60	64.10	36.73	74.28	30.95	27.22	51.41	23.28	69.75
CFSP	30.68	31.74	57.18	32.42	71.33	109.41	24.91	35.59	23.02	57.56	24.57	33.53	60.55	48.69	73.18	35.51	24.15	48.28	23.02	68.66
2SSP	28.46	32.17	56.89	31.36	71.65	107.61	25.43	35.99	23.87	58.65	22.46	34.73	63.56	45.67	74.21	31.41	25.85	48.65	25.47	68.99
BlockPruner	35.54	30.55	54.18	25.87	69.48	180.93	24.49	32.10	23.42	55.93	201.71	35.32	49.15	35.04	73.45	58.81	24.23	41.01	22.92	66.00
EvoPress	33.27	31.74	52.79	26.76	69.91	135.91	23.63	32.00	23.38	55.82	23.62	35.24	61.15	35.07	74.97	45.20	23.72	42.79	23.00	66.16
Window-Based	1.18×10^3	37.03	51.61	52.50	66.97	169.15	24.06	32.86	23.42	56.69	37.75	29.95	51.76	25.63	70.40	2.11×10^4	24.4	36.42	22.98	55.28
ShortGPT	124.40	22.44	33.48	23.00	59.30	534.06	25.43	32.45	22.93	55.01	78.47	23.38	35.00	23.49	59.68	5.37×10^4	29.18	25.76	25.64	48.69
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	72.27	26.96	42.02	25.95	64.64	115.31	25.19	35.25	23.14	57.94	51.50	38.05	59.91	36.71	72.47	74.73	25.58	45.89	24.00	66.49
budget-matched 40%																				
TADP (ours)	70.97	24.91	39.25	22.95	63.55	181.08	24.57	32.32	23.15	55.55	43.62	26.88	47.34	22.94	68.32	172.67	24.06	35.64	23.24	55.01
CFSP	109.29	24.32	35.78	23.19	59.41	202.10	22.78	30.81	23.02	55.33	46.88	22.87	44.59	27.95	64.74	201.38	23.98	31.23	22.90	55.06
2SSP	72.28	22.70	41.15	22.93	62.68	173.47	23.29	31.78	23.43	55.60	44.33	26.28	48.58	23.32	67.68	128.98	22.95	32.31	23.41	59.85
BlockPruner	128.05	25.77	34.67	23.21	59.14	441.41	24.23	28.68	23.54	54.08	319.36	25.85	37.57	24.01	65.07	307.78	22.35	29.97	22.95	57.34
EvoPress	135.02	25.00	34.68	23.27	60.55	456.94	23.12	27.38	25.35	52.39	59.05	25.68	42.40	23.11	67.08	199.30	22.95	30.68	22.95	57.56
Window-Based	$1.40x10^4$	32.08	38.10	53.88	59.36	$4.80x10^6$	27.82	26.19	22.95	50.49	80.16	25.77	38.69	25.18	63.76	$2.49x10^6$	25.85	29.53	23.10	52.77
ShortGPT	$1.74x10^3$	25.51	26.53	24.81	52.67	$2.98x10^3$	24.66	26.47	23.73	51.69	178.96	21.50	32.44	24.26	58.43	$5.7x10^4$	28.92	25.48	25.89	48.75
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	396.38	23.89	29.08	24.15	55.77	186.41	24.15	32.10	23.74	57.07	298.31	25.68	39.59	24.96	61.48	$6.54x10^3$	26.45	27.08	24.13	54.03

Table 1: 3B models: structured pruning at budget-matched 20% and 40% of reduction budget with all methods. PPL = perplexity on WikiText-2 (↓); ARC, HellaSwag (HS), MMLU, PIQA = zero-shot accuracy (↑).

	LLaMA 8B					Mistral 7B					Qwen 7B					Minerva 7B				
	Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑			
Method	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA
Baseline	13.33	53.41	79.19	62.17	80.79	11.28	54.01	81.05	59.61	82.15	12.07	51.02	78.92	71.95	79.82	12.54	48.29	73.15	40.61	78.35
budget-matched 20%																				
TADP (ours)	21.35	37.63	71.26	41.61	77.58	19.25	38.74	72.20	43.18	78.89	19.58	43.49	72.62	54.71	77.69	17.33	41.47	66.43	29.40	76.66
CFSP	23.10	37.54	66.61	42.81	74.37	19.24	37.46	69.03	37.86	77.69	19.42	38.74	69.18	58.13	75.30	17.23	39.42	65.44	32.49	75.19
2SSP	21.56	38.05	69.72	46.65	77.04	18.73	38.91	70.20	42.34	78.18	17.54	39.59	72.44	59.81	77.80	17.03	43.34	66.95	32.67	76.61
BlockPruner	29.53	35.58	61.29	33.07	71.49	22.72	39.51	69.85	36.51	75.79	37.96	39.42	64.39	27.83	76.33	19.02	39.76	66.43	35.05	74.70
EvoPress	23.19	35.67	62.50	24.53	74.21	18.43	37.29	69.84	29.02	76.17	19.78	42.83	68.09	38.69	76.61	19.68	39.33	66.84	40.85	72.91
Window-Based	1.68×10^3	29.35	33.21	35.08	59.03	40.74	31.74	55.87	23.26	69.21	29.19	36.26	60.58	28.17	73.12	19.65	41.55	66.27	39.37	73.07
ShortGPT	103.13	42.32	64.86	59.20	70.95	26.29	28.41	55.86	23.12	69.70	30.01	40.27	57.97	38.35	72.14	19.65	41.55	66.27	39.37	73.07
SliceGPT	67.17	36.60	67.55	28.44	74.92	35.89	37.63	68.20	37.44	76.61	—	—	—	—	—	—	—	—	—	—
SiRE	38.84	35.49	62.51	30.22	73.72	21.70	40.96	66.28	42.86	76.50	123.44	41.55	63.52	52.55	68.34	28.26	38.57	63.44	30.71	73.29
budget-matched 40%																				
TADP (ours)	44.42	28.24	52.72	22.92	70.95	37.10	28.92	52.80	22.92	70.35	36.17	27.73	54.32	23.23	69.86	33.45	31.23	54.30	24.28	70.46
CFSP	49.94	26.71	44.78	23.89	65.29	42.66	24.06	46.16	22.95	66.54	38.06	24.32	48.93	39.79	67.30	35.23	28.92	51.47	24.21	67.19
2SSP	45.99	27.99	49.71	27.11	69.26	37.66	25.77	50.02	22.95	68.55	34.17	25.60	56.64	30.86	69.59	68.84	30.29	51.98	25.82	68.72
BlockPruner	121.13	27.82	44.17	25.98	60.94	81.27	27.22	44.55	25.49	63.98	176.21	26.45	37.02	26.00	59.47	38.57	29.95	53.29	27.65	69.31
EvoPress	74.48	28.16	44.36	23.22	62.89	56.61	29.95	51.06	26.73	65.40	55.98	28.16	43.60	25.22	64.85	39.29	27.22	47.87	24.11	66.81
Window-Based	9842.24	29.78	37.86	34.47	59.74	1150.68	28.92	25.73	24.43	50.82	94.92	25.68	40.36	26.18	63.38	189.35	29.35	45.88	19.95	62.84
ShortGPT	1462.41	28.67	34.53	25.73	57.62	1643.57	25.68	25.70	23.67	49.73	502.57	23.63	30.23	24.70	54.68	189.35	29.35	45.88	19.95	62.84
SliceGPT	185.30	23.38	42.78	22.87	63.44	89.12	25.34	42.46	23.14	62.68	—	—	—	—	—	—	—	—	—	—
SiRE	157.23	28.92	43.25	23.98	65.02	100.30	28.50	47.86	23.55	63.71	309.74	32.18	48.12	28.59	62.30	58.66	32.25	53.00	27.00	67.63

Table 2: 7B/8B models: structured pruning at budget-matched 20% and 40% of reduction budget with all methods. PPL = perplexity on WikiText-2 (↓); ARC, HellaSwag (HS), MMLU, PIQA = zero-shot accuracy (↑).

precision. Pruning time covers calibration, importance estimation, component selection, and application of the reduction, while excluding model and tokenizer loading, dataset preparation, and downstream evaluation. Inference latency is measured using identical input and generation settings for all methods.

TADP exhibits the lowest pruning cost among the evaluated methods, requiring 30 seconds to produce the compressed model. This is lower than CFSP, 2SSP, BlockPruner, and the substantially more expensive search procedures used by EvoPress and SiRE, which require 3,240 and 3,720 seconds, respectively.

After pruning, TADP reduces inference latency from 72.02 to 49.91 ms, corresponding to a $1.44\times$ speedup. Methods that remove complete Transformer layers or residual submodules obtain larger speedups of up to $1.61\times$, despite removing a comparable number of parameters. This happens because depth and block removal preserve the original matrix shapes while reducing the number of sequential operations, whereas FFN width pruning produces narrower matrices whose practical speedup also depends on hardware utilization and kernel efficiency. It is important to note that TADP and CFSP prune only FFN channels, matching methods that also remove attention components requires

Method	Removed (%)	Time (s)	Latency (ms)	Speedup ×
Dense	0.00	–	72.02	1.00
TADP	37.60	30	49.91	1.44
CFSP	37.60	33	50.23	1.43
2SSP	35.09	76	56.30	1.28
BlockPruner	37.60	34	46.45	1.55
EvoPress	37.60	3240	44.74	1.61
Window-Based	37.60	39	44.87	1.60
ShortGPT	37.60	39	44.87	1.60
SiRE	37.60	3720	48.82	1.48

Table 3: Pruning cost and post-pruning inference efficiency on LLaMA-3.2-3B at budget-matched 40% pruning. Pruning time excludes model loading, tokenization, and downstream evaluation.

Method	Removed (%)	Time (s)	Latency (ms)	Speedup ×
Dense	0.00	–	137.75	1.00
TADP	35.31	64	93.9	1.47
CFSP	35.31	69	94.4	1.46
2SSP	34.77	128	111.6	1.23
BlockPruner	35.31	84	91.4	1.51
EvoPress	35.31	6617	90.35	1.52
Window-Based	35.31	77	89.3	1.54
ShortGPT	35.31	76	88.5	1.56
SiRE	35.31	7590	91.7	1.50
SliceGPT	32.27	1206	92.54	1.49

Table 4: Pruning cost and post-pruning inference efficiency on LLaMA-3-8B at budget-matched 40% pruning .

higher FFN pruning ratios. In contrast, 2SSP and SliceGPT follow their original allocation; consequently, their closest configurations remove slightly fewer total parameters.

Table 5 reports the exact fraction of total model parameters removed in the two compression regimes used in the comparison, for a complete and more accurate comparison refer to Appendix C. Model-specific FFN ratios are used for TADP to match the parameter reductions of the corresponding pruning methods. 2SSP follows its original hybrid FFN/attention allocation and therefore retains slightly more parameters in most settings.

6 Conclusion

We presented TADP, a structured pruning method for decoder-only large language models that estimates neuron importance through the temporal dynamics of activations rather than static scalar summaries. By decomposing neuron activations into five complementary signals mean activation, peak activation, positional variance, transition dynamics, and position-weighted response, TADP explicitly prioritizes neurons exhibiting strong peak activations on rare contextual patterns. Our results suggest that these highly selective neurons correspond to structurally important components for preserving sequential coherence and context-sensitive generation.

	20%		40%	
Model	All	2SSP	All	2SSP
LLaMA-3.2-3B	18.80	17.55	37.60	35.09
Ministral-3B-Instruct	19.74	18.42	39.47	36.84
Qwen2.5-3B	19.98	17.98	37.46	35.97
Minerva-3B	20.61	18.84	38.27	37.68
LLaMA-3-8B	19.01	17.38	35.31	34.77
Mistral-7B	21.08	19.28	39.15	38.55
Qwen2.5-7B	18.36	17.14	36.72	34.27
Minerva-7B	20.64	18.87	38.32	37.73

Table 5: Effective fraction of total model parameters removed (%).

Evaluated on eight models across the LLaMA, Mistral, Qwen, and Minerva families at both 3B and 7B/8B scale, TADP achieves the joint-best performance in the majority of evaluated settings across different pruning regimes and all model families. The largest gains are observed on WikiText-2 perplexity and HellaSwag accuracy, two benchmarks that strongly depend on coherent contextual continuation and autoregressive language modeling quality. At 40% pruning, where competing pruning methods exhibit severe degradation, TADP preserves substantially lower perplexity and stronger downstream performance. Overall, the results suggest that modeling activation dynamics along the temporal dimension provides a promising direction for robust and training-free compression of large language models under high pruning regimes.

Two limitations constrain the current method. First, pruning is applied exclusively to the FFN sublayers, while self-attention modules remain structurally intact; extending the temporal activation criterion to attention heads represents a natural direction for future work. Second, the reliance on a general calibration corpus may favor the preservation of contextual and sequential processing over factual knowledge retrieval, which could partially explain the comparatively weaker results observed on MMLU. Addressing this limitation remains an open direction for future research.

Limitations

TADP currently prunes only MLP sublayers, leaving self-attention modules unchanged. While this targets the dominant parameter component in decoder-only LLMs, extending the temporal activation criterion to attention heads could further improve compression efficiency.

In addition, TADP relies on temporal activation dynamics measured on a general calibration corpus. This criterion appears effective for preserving contextual and syntactic processing, but less suited to factual knowledge retrieval, which may explain the comparatively weaker results on MMLU. Future work could investigate task-specific calibration data or importance metrics that better capture parametric knowledge storage.

Ethical Considerations

TADP is a structured pruning method for publicly available large language models and does not involve collecting or annotating personal data. All calibration and evaluation datasets are standard public NLP benchmarks.

Compressed models produced by TADP inherit the biases and failure modes of the original models, and pruning may alter these behaviours unpredictably. Therefore, compressed variants should be evaluated independently before deployment in sensitive applications.

By reducing inference cost and memory usage, TADP can improve accessibility and energy efficiency of LLM deployment. However, lower hardware requirements may also facilitate deployment in settings with limited oversight, motivating responsible use of compression techniques.

Acknowledgment

The work of Adriano Puglisi is supported by Thales Alenia Space and Regione Lazio through the fellowship "AI-based Digital Industrial Processes for Space Systems Production Workflow" (CUP B83C23001060009; SIGEM 22066DP000000035). This work has also been partly supported by the EU projects GenDAI (ID 101182801), GAINAfrica (ID 101298559), and CoAgent (ID 101297191).

References

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. SliceGPT: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*.

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*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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. *Preprint*, arXiv:1803.05457.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm.int8(): 8-bit matrix multiplication for transformers at scale. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Elias Frantar and Dan Alistarh. 2023. SparseGPT: Massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Dan Roberts. 2025. The unreasonable ineffectiveness of the deeper layers. In *The Thirteenth International Conference on Learning Representations*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Song Guo, Jiahang Xu, Li Lyna Zhang, and Mao Yang. 2024. Compresso: Structured pruning with collaborative prompting learns compact large language models.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021a. Aligning AI with shared human values. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021b. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, and 101 others. 2026. Ministral 3. *Preprint*, arXiv:2601.08584.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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, Vienna, Austria. Association for Computational Linguistics.

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

Riccardo Orlando, Luca Moroni, Pere-Lluís Huguet Cabot, Simone Conia, Edoardo Barba, Sergio Orlandini, Giuseppe Fiameni, and Roberto Navigli. 2024. Minerva LLMs: The first family of large language models trained from scratch on Italian data. In *Proceedings of the Tenth Italian Conference on Computational Linguistics (CLiC-it 2024)*, pages 707–719, Pisa, Italy. CEUR Workshop Proceedings.

Adriano Puglisi, Flavia Monti, Francesco Leotta, Christian Napoli, and Massimo Mecella. 2026. Slimai4edge: A cloud-edge framework for downsizing ai models as-a-service. In *Advanced Information Systems Engineering: 38th International Conference, CAiSE 2026, Verona, Italy, June 8–12, 2026, Proceedings, Part I*, page 341–357, Berlin, Heidelberg. Springer-Verlag.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Fabrizio Sandri, Elia Cunegatti, and Giovanni Iacca. 2025. 2SSP: A two-stage framework for structured pruning of LLMs. *Transactions on Machine Learning Research*.

Oliver Sieberling, Denis Kuznedelev, Eldar Kurtic, and Dan Alistarh. 2025. Evopress: Accurate dynamic model compression via evolutionary search. In *Forty-second International Conference on Machine Learning*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. In *Workshop on Efficient Systems for Foundation Models @ ICML2023*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Yuxin Wang, MingHua Ma, Zekun Wang, Jingchang Chen, Shan Liping, Qing Yang, Dongliang Xu, Ming Liu, and Bing Qin. 2025. CFSP: An efficient structured pruning framework for LLMs with coarse-to-fine activation information. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9311–9328, Abu Dhabi, UAE. Association for Computational Linguistics.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

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*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. 2025. BlockPruner: Fine-grained pruning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5065–5080, Vienna, Austria. Association for Computational Linguistics.

A Importance score ablation study

To validate the proposed scoring formulation, we conduct an ablation study on LLaMA 3B using WikiText-2 perplexity at 50% pruning. We selected LLaMA 3B as a representative model since the effectiveness of TADP is subsequently demonstrated across eight different architectures and model families in the full experimental evaluation. The analysis focuses on perplexity because it directly reflects language modeling quality and therefore provides a suitable setting for evaluating whether emphasizing peak activations in the importance score better preserves temporally selective neurons and contextual generation behavior. We additionally perform the study at 50% pruning because this represents the most aggressive compression regime considered in our experiments, where differences between scoring components become more pronounced. The analysis considers all combinations of the five temporal descriptors characterizing the importance score $s_i^{(\ell)}$, with coefficients α_j: mean activation (M), peak activation (P), positional variance (V), transition dynamics (G), and position-weighted response (W). In total, we evaluate 47 score configurations. Specifically, the analysis covers all $2^5 - 1 = 31$ non-empty descriptor subsets under uniform weighting and all $2^4 = 16$ subsets containing peak activation under peak-heavy weighting. In the uniform setting (U), all active descriptors receive the same weight, normalized to sum to one. In the peak-heavy setting (H), the peak descriptor is assigned a weight of 0.35, while the remaining active descriptors equally share the residual weight.

Table 6 reports the results. The complete five-component formulation with peak-heavy weighting achieves the best performance, reaching the lowest perplexity of 60.78. This indicates that each component contributes meaningful information and that no term is entirely redundant. Removing any individual component consistently degrades performance. Among them, the transition dynamics term appears to be the least influential when considered independently, as excluding it only slightly increases perplexity from 60.78 to 65.49. In contrast, the peak activation term is clearly the most influential component. Configurations without P exhibit a substantial deterioration in perplexity, with the mean-only configuration reaching 109.68.

Across equivalent configurations, the peak-heavy strategy systematically outperforms uniform weighting, supporting the decision to emphasize peak selectivity within the scoring formulation. Moreover, peak activation alone achieves a perplexity of 73.07, significantly outperforming isolated variants based on positional variance (PPL of 97.14), mean activation (PPL of 109.68), or position-weighted response (PPL of 110.95).

The position-weighted response component r_i^ℓ contributes only marginally to the final score, as rank correlations between r_i^ℓ and m_i^ℓ exceed 0.999 across many layers, indicating that the two signals capture largely overlapping information. Despite this redundancy, its inclusion still improves perplexity compared to omitting it entirely, leading to its retention in the final configuration.

B Complete results

Table 7 and Table 8 report the complete structured pruning results across all evaluated pruning levels for the 3B and 7B/8B model families, respectively. These tables extend the main-paper analysis presented at 20% and 40% pruning ratios by including intermediate pruning settings (i.e., 10%, 30%, and 50%) together with the full set of evaluation metrics. Results are reported in terms of WikiText-2 perplexity and zero-shot accuracy on ARC, HellaSwag, MMLU, and PIQA. Overall, TADP consistently achieves the best trade-off between compression and performance degradation, maintaining lower perplexity and stronger downstream accuracy across most architectures and pruning regimes, particularly at higher pruning levels where competing methods exhibit a more pronounced collapse in language modeling quality.

Config	Scheme	PPL	Config	Scheme	PPL
Baseline	–	15.97	U[M+P+V+W]	uniform	64.95
H[M+P+V+G+W]	peak-heavy	**60.78**	H[M+P+V]	peak-heavy	65.07
H[M+P+V+G]	peak-heavy	61.48	H[M+P+W]	peak-heavy	65.13
H[M+P+G+W]	peak-heavy	61.75	U[M+P+W]	uniform	65.20
U[M+P+V+G+W]	uniform	61.89	U[M+P+V]	uniform	65.47
U[M+P+G+W]	uniform	61.89	H[M+P+V+W]	peak-heavy	65.49
H[P+V+G+W]	peak-heavy	61.91	H[P+W]	peak-heavy	65.55
U[M+P+V+G]	uniform	62.32	U[M+P]	uniform	65.65
H[M+P+G]	peak-heavy	62.40	U[P+V+W]	uniform	65.84
U[M+P+G]	uniform	62.51	U[V+G]	uniform	65.89
U[P+V+G+W]	uniform	62.71	U[G]	uniform	65.99
U[P+G+W]	uniform	62.79	H[P+V+W]	peak-heavy	66.18
U[M+G+W]	uniform	63.09	U[P+W]	uniform	66.31
U[M+V+G]	uniform	63.14	U[P+V]	uniform	70.03
U[M+G]	uniform	63.15	H[P+V]	peak-heavy	70.32
H[P+G+W]	peak-heavy	63.19	U[P]	uniform	73.07
U[M+V+G+W]	uniform	63.20	H[P]	peak-heavy	73.07
U[V+G+W]	uniform	63.72	U[V]	uniform	97.14
U[G+W]	uniform	63.75	U[M+V+W]	uniform	107.42
H[P+V+G]	peak-heavy	63.91	U[M+V]	uniform	108.57
U[P+V+G]	uniform	64.07	U[M]	uniform	109.68
H[P+G]	peak-heavy	64.23	U[M+W]	uniform	109.81
U[P+G]	uniform	64.57	U[V+W]	uniform	110.80
H[M+P]	peak-heavy	64.82	U[W]	uniform	110.95

Table 6: Ablation study of the proposed pruning score at 50% pruning ratio. Lower perplexity (PPL) indicates better language modeling performance. Configuration abbreviations: M = mean activation, P = peak activation, V = positional variance, G = transition dynamics, W = position-weighted response. U denotes uniform weighting, while H denotes peak-heavy weighting.

To complement the numerical comparisons reported in Table 7 and Table 8, we provide heatmap visualizations of pruning performance across all evaluated models and reduction levels in Figure 3 and Figure 4. Each row corresponds to a pruning method, while columns represent different model architectures. Separate heatmaps are reported for downstream accuracy retention (cf. Figure 3) and perplexity degradation (cf. Figure 4). For perplexity, values are normalized with respect to the corresponding unpruned baseline using $\log_2(\mathrm{PPL}_{pruned}/\mathrm{PPL}_{baseline})$, where lower values indicate better preservation of language modeling quality. For downstream evaluation, values represent the average retained zero-shot accuracy across ARC, HellaSwag, MMLU, and PIQA relative to the baseline model.

C Effective parameter reductions

The nominal pruning ratio does not represent the same operation across method families. TADP and CFSP remove intermediate FFN channels, ShortGPT and Window-Based remove complete Transformer layers, BlockPruner and EvoPress operate on attention and MLP components, and 2SSP distributes its compression budget between FFN width reduction and the removal of attention submodules. SliceGPT instead reduces the shared embedding dimension by slicing rows and columns across the model's projection matrices. Consequently, applying the same nominal target can produce substantially different final model sizes. To make these differences explicit, we compute the effective reduction over all model parameters by comparing the parameter count of each structurally pruned model with that of its original dense counterpart. Table 9 reports this effective reduction for every available model, method, and target configuration. For TADP and CFSP, model-specific FFN pruning ratios are selected to approach the effective parameter budgets produced by the corresponding block and depth pruning configurations. Small residual differences arise from architectural granularity, including the need to remove an integer number of layers, submodules, channels, or embedding dimensions. Under its original allocation rule, 2SSP removes slightly fewer total parameters in several configurations and therefore retains a modest parameter-budget advantage. SliceGPT results are reported only for architectures supported by its implementation.

	LLaMA 3B					Ministral-3B-Instruct					Qwen 3B					Minerva 3B				
	Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑			
Method	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA
Baseline	15.97	45.99	73.61	54.40	77.48	92.04	26.19	37.75	24.60	59.96	14.58	47.27	73.52	65.20	78.94	18.70	29.86	60.81	24.70	75.41
pruning 10%																				
TADP (ours)	20.61	38.31	70.37	46.48	76.44	100.15	26.11	37.41	24.08	59.90	17.79	42.66	71.13	57.38	76.82	22.80	27.99	56.14	24.96	72.47
CFSP	20.90	38.91	67.23	45.29	75.63	100.61	24.91	36.68	23.47	58.22	17.82	41.64	68.31	58.76	76.82	24.61	26.88	55.70	25.20	71.76
2SSP	20.09	38.91	68.13	46.15	75.52	97.90	26.19	37.30	24.12	59.52	17.21	42.06	70.20	56.95	77.15	22.49	27.30	55.75	25.61	72.36
BlockPruner	23.01	37.97	64.65	33.16	74.48	127.23	23.72	33.55	23.29	57.18	142.93	43.86	58.35	49.17	77.20	30.48	26.54	49.95	23.22	70.62
EvoPress	21.56	35.92	63.84	34.28	74.81	113.80	25.09	34.08	24.11	56.64	16.98	40.70	65.81	48.43	75.19	31.41	25.94	51.92	23.79	69.42
Window-Based	164.91	42.06	62.29	52.69	72.20	110.36	25.51	34.08	24.20	56.96	25.69	35.32	62.51	25.82	73.07	$1.91x10^4$	27.39	40.0	23.64	59.41
ShortGPT	68.04	23.98	38.35	23.28	59.41	127.57	24.57	34.48	24.80	57.78	27.17	43.43	60.91	44.62	70.46	$5.98x10^4$	27.73	25.76	25.98	48.69
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	37.87	28.58	53.30	27.47	71.00	99.15	26.02	36.53	23.09	59.36	72.42	46.93	69.87	60.28	72.31	52.94	28.24	51.92	25.62	70.89
pruning 20%																				
TADP (ours)	28.31	33.19	62.97	31.00	74.05	105.93	25.94	36.76	23.57	59.09	22.18	36.60	64.10	36.73	74.28	30.95	27.22	51.41	23.28	69.75
CFSP	30.68	31.74	57.18	32.42	71.33	109.41	24.91	35.59	23.02	57.56	24.57	33.53	60.55	48.69	73.18	35.51	24.15	48.28	23.02	68.66
2SSP	28.46	32.17	56.89	31.36	71.65	107.61	25.43	35.99	23.87	58.65	22.46	34.73	63.56	45.67	74.21	31.41	25.85	48.65	25.47	68.99
BlockPruner	35.54	30.55	54.18	25.87	69.48	180.93	24.49	32.10	23.42	55.93	201.71	35.32	49.15	35.04	73.45	58.81	24.23	41.01	22.92	66.00
EvoPress	33.27	31.74	52.79	26.76	69.91	135.91	23.63	32.00	23.38	55.82	23.62	35.24	61.15	35.07	74.97	45.20	23.72	42.79	23.00	66.16
Window-Based	$1.18x10^3$	37.03	51.61	52.50	66.97	169.15	24.06	32.86	23.42	56.69	37.75	29.95	51.76	25.63	70.40	$2.11x10^4$	24.40	36.42	22.98	55.28
ShortGPT	124.40	22.44	33.48	23.00	59.30	534.06	25.43	32.45	22.93	55.01	78.47	23.38	35.00	23.49	59.68	$5.37x10^4$	29.18	25.76	25.64	48.69
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	72.27	26.96	42.02	25.95	64.64	115.31	25.19	35.25	23.14	57.94	51.50	38.05	59.91	36.71	72.47	74.73	25.58	45.89	24.00	66.49
pruning 30%																				
TADP (ours)	46.92	30.46	52.59	23.35	68.99	147.60	24.74	34.07	23.02	57.73	30.94	32.08	57.17	24.06	70.57	61.66	23.89	42.92	23.29	63.17
CFSP	52.73	26.45	44.87	26.41	65.40	156.98	22.87	32.56	23.24	56.09	32.16	29.86	53.91	32.32	69.80	66.08	22.70	40.17	22.83	61.97
2SSP	42.96	27.13	48.91	29.80	68.55	130.52	25.43	33.94	22.96	57.45	31.86	30.80	56.30	33.83	70.08	58.02	23.63	39.07	23.75	63.71
BlockPruner	67.95	26.79	43.64	23.30	62.57	316.47	24.06	29.41	24.06	55.06	222.66	31.57	44.05	26.45	71.27	138.58	23.98	34.58	22.95	60.94
EvoPress	67.76	28.84	43.56	23.05	63.22	274.21	23.98	29.61	22.89	54.68	34.84	32.25	51.87	24.33	70.35	83.53	23.55	37.31	22.99	61.86
Window-Based	$3.80x10^3$	33.28	44.42	50.85	63.71	823.30	25.34	26.56	25.47	51.20	48.90	28.16	46.16	23.71	69.48	$5.93x10^5$	26.02	33.03	26.37	54.68
ShortGPT	362.92	21.16	29.04	22.89	56.58	$2.12x10^3$	24.32	27.97	25.66	52.83	129.83	21.84	32.37	23.69	58.16	$6.28x10^3$	29.35	25.79	25.76	48.62
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	145.79	24.40	35.00	23.52	59.14	142.16	24.91	33.54	23.44	57.34	100.42	30.38	50.75	28.97	68.77	192.46	23.98	35.29	23.07	60.07
pruning 40%																				
TADP (ours)	70.97	24.91	39.25	22.95	63.55	181.08	24.57	32.32	23.15	55.55	43.62	26.88	47.34	22.94	68.32	172.67	24.06	35.64	23.24	55.01
CFSP	109.29	24.32	35.78	23.19	59.41	202.10	22.78	30.81	23.02	55.33	46.88	22.87	44.59	27.95	64.74	201.38	23.98	31.23	22.90	55.06
2SSP	72.28	22.70	41.15	22.93	62.68	173.47	23.29	31.78	23.43	55.60	44.33	26.28	48.58	23.32	67.68	128.98	22.95	32.31	23.41	59.85
BlockPruner	128.05	25.77	34.67	23.21	59.14	441.41	24.23	28.68	23.54	54.08	319.36	25.85	37.57	24.01	65.07	307.78	22.35	29.97	22.95	57.34
EvoPress	135.02	25.00	34.68	23.27	60.55	456.94	23.12	27.38	25.35	52.39	59.05	25.68	42.40	23.11	67.08	199.30	22.95	30.68	22.95	57.56
Window-Based	$1.40x10^4$	32.08	38.10	53.88	59.36	$4.80x10^6$	27.82	26.19	22.95	50.49	80.16	25.77	38.69	25.18	63.76	$2.49x10^6$	25.85	29.53	23.10	52.77
ShortGPT	$1.74x10^3$	25.51	26.53	24.81	52.67	$2.98x10^3$	24.66	26.47	23.73	51.69	178.96	21.50	32.44	24.26	58.43	$5.7x10^4$	28.92	25.48	25.89	48.75
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	396.38	23.89	29.08	24.15	55.77	186.41	24.15	32.10	23.74	57.07	298.31	25.68	39.59	24.96	61.48	$6.54x10^3$	26.45	27.08	24.13	54.03
pruning 50%																				
TADP (ours)	181.25	23.98	32.17	22.86	56.91	277.48	24.23	30.49	22.97	54.84	64.05	23.89	40.87	22.96	63.98	$3.14x10^3$	27.82	26.48	23.71	52.01
CFSP	182.74	22.35	31.09	22.92	55.88	294.45	24.66	29.27	22.90	53.86	64.13	23.21	38.99	23.10	61.59	$6.32x10^3$	27.05	25.59	24.48	50.60
2SSP	120.63	22.44	34.23	22.95	60.07	308.47	23.63	29.50	23.14	54.19	64.58	24.83	41.53	22.94	62.40	307.68	22.95	27.84	23.34	54.35
BlockPruner	240.95	24.49	31.54	23.12	55.17	721.99	23.63	27.58	23.82	53.75	642.77	25.34	32.83	23.06	58.54	580.56	22.78	27.83	22.95	54.95
EvoPress	287.09	25.00	31.92	24.22	56.58	620.34	25.34	27.14	25.87	54.24	136.38	23.12	34.80	23.89	60.12	$5.14x10^3$	26.54	26.42	24.54	50.27
Window-Based	$4.88x10^4$	23.92	33.78	23.89	54.57	$3.31x10^3$	26.19	25.92	22.97	49.51	178.79	23.04	32.77	25.34	59.47	$2.85x10^6$	25.26	28.32	22.89	51.90
ShortGPT	$6.23x10^3$	24.83	26.07	25.85	51.31	$6.40x10^3$	26.11	25.79	22.88	51.03	225.67	21.93	31.25	24.12	57.73	$6.86x10^4$	28.16	25.48	25.40	48.80
SliceGPT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SiRE	$1.60x10^3$	24.74	27.50	25.69	51.74	294.96	24.57	30.80	23.89	55.98	575.39	24.40	33.37	23.87	57.18	$5.89x10^4$	23.33	26.14	22.22	51.58

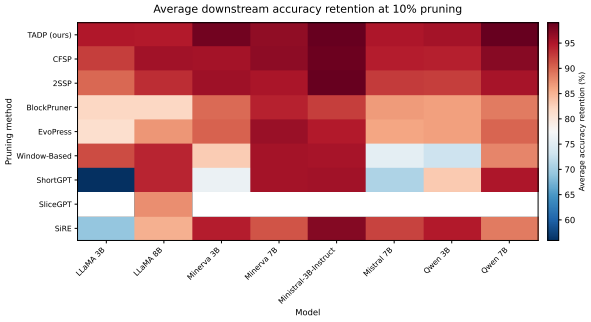
Table 7: 3B models: structured pruning at 10%, 20%, 30%, 40%, and 50% of reduction. PPL = perplexity on WikiText-2 (↓); ARC, HellaSwag (HS), MMLU, PIQA = zero-shot accuracy (↑).

	LLaMA 8B					Mistral 7B					Qwen 7B					Minerva 7B				
	Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑				Perplexity↓	Accuracy↑			
Method	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA	WikiText-2	ARC	HS	MMLU	PIQA
Baseline	13.33	53.41	79.19	62.17	80.79	11.28	54.01	81.05	59.61	82.15	12.07	51.02	78.92	71.95	79.82	12.54	48.29	73.15	40.61	78.35
pruning 10%																				
TADP (ours)	16.24	45.39	76.30	51.84	79.49	14.41	46.59	78.27	48.28	81.34	14.44	49.57	77.85	65.32	79.82	14.10	44.88	70.26	34.60	78.24
CFSP	17.39	46.50	74.03	51.37	78.40	14.30	44.97	76.67	50.23	80.41	14.31	50.17	75.78	65.45	79.22	14.09	44.37	69.45	37.01	77.91
2SSP	16.50	48.04	75.85	55.49	79.54	14.50	47.18	77.69	53.20	80.41	13.89	46.25	77.36	66.84	79.49	13.59	46.50	70.71	36.48	76.66
BlockPruner	19.84	42.58	70.39	39.88	76.50	15.41	44.80	75.21	44.50	79.05	15.53	45.31	71.44	55.55	78.40	15.01	43.34	69.62	37.82	76.77
EvoPress	17.87	42.92	71.97	50.40	77.53	13.92	45.90	76.41	40.19	80.09	13.73	44.88	73.67	57.03	79.33	14.49	43.94	70.30	40.74	75.68
Window-Based	30.50	48.04	73.59	60.96	76.33	14.73	38.31	71.86	27.47	78.73	21.41	51.11	68.09	51.74	74.70	14.69	43.26	69.88	40.19	76.28
ShortGPT	30.50	48.04	73.59	60.96	76.33	16.42	38.99	63.32	23.50	76.93	17.69	52.82	73.10	60.51	79.22	14.69	43.26	69.88	40.19	76.28
SliceGPT	430.94	47.18	75.01	43.08	78.62	21.81	45.31	76.41	51.36	79.92	—	—	—	—	—	—	—	—	—	—
SiRE	25.41	44.88	71.96	44.27	76.33	16.45	50.00	73.25	53.10	79.11	72.42	46.93	69.87	60.28	72.31	22.03	42.32	67.05	35.94	75.35
pruning 20%																				
TADP (ours)	21.35	37.63	71.26	41.61	77.58	19.25	38.74	72.20	43.18	78.89	19.58	43.49	72.62	54.71	77.69	17.33	41.47	66.43	29.40	76.66
CFSP	23.10	37.54	66.61	42.81	74.37	19.24	37.46	69.03	37.86	77.69	19.42	38.74	69.18	58.13	75.30	17.23	39.42	65.44	32.49	75.19
2SSP	21.56	38.05	69.72	46.65	77.04	18.73	38.91	70.20	42.34	78.18	17.54	39.59	72.44	59.81	77.80	17.03	43.34	66.95	32.67	76.61
BlockPruner	29.53	35.58	61.29	33.07	71.49	22.72	39.51	69.85	36.51	75.79	37.96	39.42	64.39	27.83	76.33	19.02	39.76	66.43	35.05	74.70
EvoPress	23.19	35.67	62.50	24.53	74.21	18.43	37.29	69.84	29.02	76.17	19.78	42.83	68.09	38.69	76.61	19.68	39.33	66.84	40.85	72.91
Window-Based	1680.26	29.35	33.21	35.08	59.03	40.74	31.74	55.87	23.26	69.21	29.19	36.26	60.58	28.17	73.12	19.65	41.55	66.27	39.37	73.07
ShortGPT	103.13	42.32	64.86	59.20	70.95	26.29	28.41	55.86	23.12	69.70	30.01	40.27	57.97	38.35	72.14	19.65	41.55	66.27	39.37	73.07
SliceGPT	67.17	36.60	67.55	28.44	74.92	35.89	37.63	68.20	37.44	76.61	—	—	—	—	—	—	—	—	—	—
SiRE	38.84	35.49	62.51	30.22	73.72	21.17	40.96	66.28	42.86	76.50	123.44	41.55	63.52	52.55	68.34	28.26	38.57	63.44	30.71	73.29
pruning 30%																				
TADP (ours)	30.87	35.58	63.70	25.67	76.06	26.65	32.00	64.04	23.10	75.19	26.27	34.13	64.56	32.09	73.78	22.88	36.69	61.20	26.66	73.88
CFSP	32.38	30.97	56.26	30.47	71.60	27.63	29.86	58.37	24.42	72.91	27.13	29.78	59.39	48.91	71.16	23.06	35.84	59.49	29.46	72.36
2SSP	31.16	31.66	58.73	29.23	73.07	25.36	30.29	60.56	32.30	72.74	24.18	32.94	65.55	47.76	73.18	27.98	38.23	61.84	29.61	74.16
BlockPruner	47.71	31.31	53.83	27.30	67.19	39.19	32.51	57.68	26.88	70.73	96.23	32.08	51.53	26.14	69.86	26.78	34.64	59.58	30.15	71.27
EvoPress	34.95	31.57	54.53	25.79	69.48	27.18	31.06	59.10	23.33	71.22	24.22	32.42	55.11	26.46	71.22	24.45	33.11	57.24	24.89	71.22
Window-Based	3521.38	30.46	33.33	36.71	60.88	269.99	28.07	34.66	23.14	56.58	40.78	29.61	52.39	23.47	69.10	29.47	34.73	61.70	37.59	67.85
ShortGPT	3521.38	30.46	33.33	36.71	60.88	95.26	24.91	34.00	23.20	60.01	216.63	25.85	32.78	23.62	57.67	30.73	35.49	60.61	36.98	70.40
SliceGPT	88.37	29.01	55.95	23.42	69.86	52.76	29.01	55.74	26.07	70.57	—	—	—	—	—	—	—	—	—	—
SiRE	64.66	33.53	53.17	27.31	69.53	32.45	33.70	57.00	26.88	71.33	208.92	37.29	55.78	39.00	66.54	39.82	36.52	58.89	31.37	71.00
pruning 40%																				
TADP (ours)	44.42	28.24	52.72	22.92	70.95	37.10	28.92	52.80	22.92	70.35	36.17	27.73	54.32	23.23	69.86	33.45	31.23	54.30	24.28	70.46
CFSP	49.94	26.71	44.78	23.89	65.29	42.66	24.06	46.16	22.95	66.54	38.06	24.32	48.93	39.79	67.30	35.23	28.92	51.47	24.21	67.19
2SSP	45.99	27.99	49.71	27.11	69.26	37.66	25.77	50.02	22.95	68.55	34.17	25.60	56.64	30.86	69.59	68.84	30.29	51.98	25.82	68.72
BlockPruner	121.13	27.82	44.17	25.98	60.94	81.27	27.22	44.55	25.49	63.98	176.21	26.45	37.02	26.00	59.47	38.57	29.95	53.29	27.65	69.31
EvoPress	74.48	28.16	44.36	23.22	62.89	56.61	29.95	51.06	26.73	65.40	55.98	28.16	43.60	25.22	64.85	39.29	27.22	47.87	24.11	66.81
Window-Based	9842.24	29.78	37.86	34.47	59.74	1150.68	28.92	25.73	24.43	50.82	94.92	25.68	40.36	26.18	63.38	189.35	29.35	45.88	19.95	62.84
ShortGPT	1462.41	28.67	34.53	25.73	57.62	1643.57	25.68	25.70	23.67	49.73	502.57	23.63	30.23	24.70	54.68	189.35	29.35	45.88	19.95	62.84
SliceGPT	185.30	23.38	42.78	22.87	63.44	89.12	25.34	42.46	23.14	62.68	—	—	—	—	—	—	—	—	—	—
SiRE	157.23	28.92	43.25	23.98	65.02	100.30	28.50	47.86	23.55	63.71	309.74	32.18	48.12	28.59	62.30	58.66	32.25	53.00	27.00	67.63
pruning 50%																				
TADP (ours)	86.29	25.09	39.64	22.93	63.60	55.40	24.83	42.63	22.96	63.76	48.07	24.57	47.58	22.95	66.81	61.50	26.79	45.05	24.13	64.80
CFSP	90.81	21.33	34.49	22.72	59.14	74.82	22.78	36.81	22.96	60.17	51.97	23.46	42.20	30.84	64.36	63.33	27.73	41.67	23.69	63.55
2SSP	76.40	24.23	39.66	26.02	64.64	56.85	22.35	40.76	22.94	63.44	48.84	23.55	46.06	22.99	66.43	182.08	26.79	44.60	26.19	66.21
BlockPruner	386.76	24.83	33.75	24.21	57.13	262.70	25.68	36.11	25.01	57.24	406.72	24.23	32.98	25.27	55.71	62.53	25.26	41.18	24.55	62.02
EvoPress	179.11	23.21	33.63	23.05	59.96	134.99	27.05	38.06	22.62	58.00	107.65	23.72	36.27	23.10	59.41	70.83	25.94	40.59	24.55	60.94
Window-Based	1051.16	21.84	28.59	23.05	53.65	3576.05	29.01	25.75	24.83	50.65	42873.95	26.71	26.71	25.31	51.85	863.41	30.80	47.57	22.83	62.57
ShortGPT	3740.73	27.82	28.81	23.05	53.97	6656.26	28.58	25.60	24.65	50.98	774.18	24.40	28.85	23.44	54.84	603.12	31.74	47.34	35.66	63.66
SliceGPT	327.57	21.67	34.91	22.89	59.03	139.87	22.27	34.68	22.97	56.91	—	—	—	—	—	—	—	—	—	—
SiRE	479.17	25.34	36.72	23.91	61.04	556.02	27.39	35.99	23.10	56.26	533.30	26.88	39.22	23.82	55.71	96.56	28.75	46.98	23.32	64.42

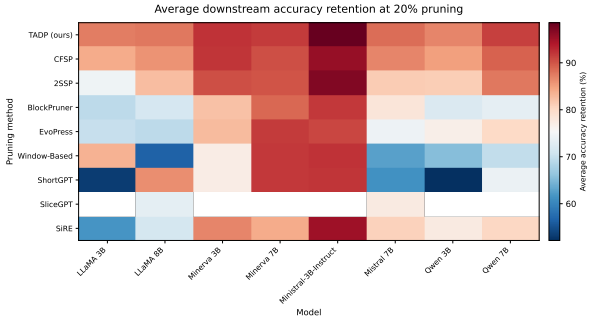
Table 8: 7B/8B models: Structured pruning at 10%, 20%, 30%, 40%, and 50% of reduction. PPL = perplexity on WikiText-2 (↓); ARC, HellaSwag (HS), MMLU, PIQA = zero-shot accuracy (↑).

Model	Method(s)	Nominal target				
		10%	**20%**	**30%**	**40%**	**50%**
LLaMA-3.2-3B	TADP/CFSP	9.40	18.80	28.20	37.60	43.87
	ShortGPT / Window-Based	9.40	18.80	28.20	37.60	43.87
	BlockPruner / EvoPress	9.40	18.80	28.20	37.60	43.87
	2SSP	8.77	17.55	26.32	35.09	43.87
Ministral-3B-Instruct	TADP/CFSP	13.16	19.74	32.89	39.47	46.05
	ShortGPT / Window-Based	13.16	19.73	32.89	39.47	46.05
	BlockPruner / EvoPress	13.16	19.73	32.89	39.47	46.05
	2SSP	9.21	18.42	27.63	36.84	46.04
Qwen2.5-3B	TADP/CFSP	9.99	19.98	27.47	37.46	44.96
	ShortGPT / Window-Based	9.99	19.98	27.47	37.47	44.96
	BlockPruner / EvoPress	9.99	19.98	27.47	37.46	44.96
	2SSP	9.00	17.98	26.98	35.97	44.96
Minerva-3B	TADP/CFSP	11.78	20.61	29.44	38.27	47.10
	ShortGPT / Window-Based	11.78	20.61	29.44	38.27	47.10
	BlockPruner / EvoPress	11.77	20.61	29.44	38.27	47.10
	2SSP	9.42	18.84	28.26	37.68	47.10
LLaMA-3-8B	TADP/CFSP	10.87	19.01	27.16	35.31	43.46
	ShortGPT / Window-Based	10.86	19.01	27.16	35.31	43.46
	BlockPruner / EvoPress	10.86	19.01	27.16	35.31	43.46
	2SSP	8.69	17.38	26.07	34.77	43.46
	SliceGPT	-1.39	10.09	21.31	32.27	42.97
Mistral-7B	TADP/CFSP	12.05	21.08	30.12	39.15	48.19
	ShortGPT / Window-Based	12.05	21.08	30.12	39.15	48.19
	BlockPruner / EvoPress	12.05	21.08	30.12	39.15	48.19
	2SSP	9.64	19.28	28.91	38.55	48.19
	SliceGPT	-2.10	10.09	21.99	33.61	44.93
Qwen2.5-7B	TADP/CFSP	9.18	18.36	27.54	36.72	42.84
	ShortGPT / Window-Based	9.18	18.36	27.54	36.72	42.84
	BlockPruner / EvoPress	9.18	18.36	27.54	36.72	42.84
	2SSP	8.57	17.14	25.71	34.27	42.84
Minerva-7B	TADP/CFSP	11.79	20.64	29.48	38.32	47.17
	ShortGPT / Window-Based	11.79	20.63	29.48	38.32	47.17
	BlockPruner / EvoPress	11.79	20.63	29.48	38.32	47.16
	2SSP	9.43	18.87	28.30	37.73	47.17

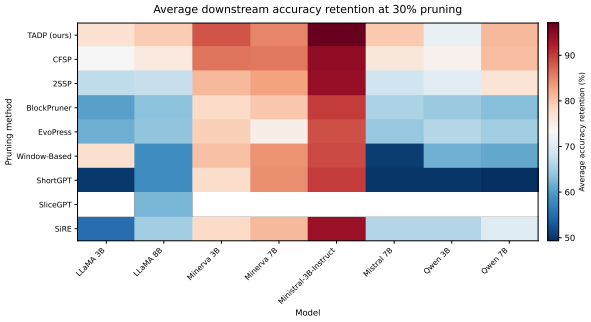
Table 9: Effective fraction of total model parameters removed (%) at each nominal target. Entries are computed from the parameter counts of the dense and structurally pruned models. Small differences between methods reflect their distinct pruning spaces and architectural granularities. SliceGPT is reported only for supported architectures. At the lowest slicing level, negative reductions indicate that the resulting SliceGPT representation contains slightly more registered parameters than the original dense model.



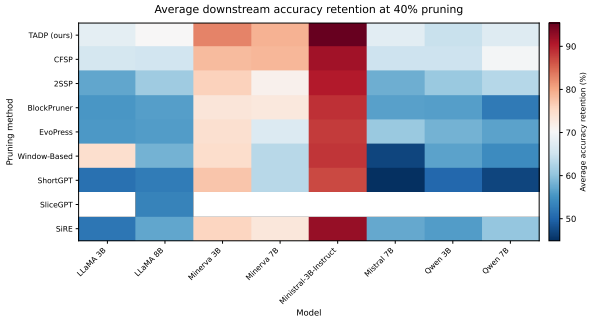
(a) 10% of reduction



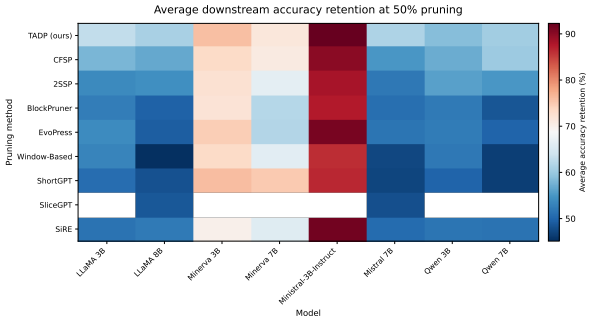
(b) 20% of reduction



(c) 30% of reduction

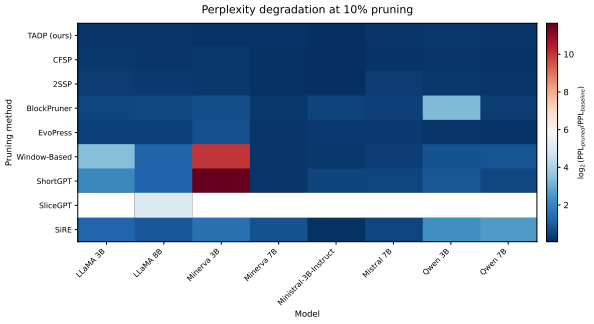


(d) 40% of reduction

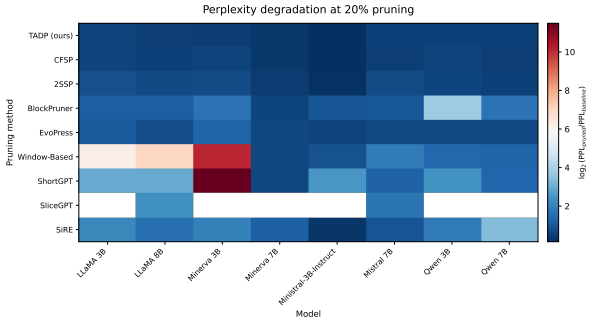


(e) 50% of reduction

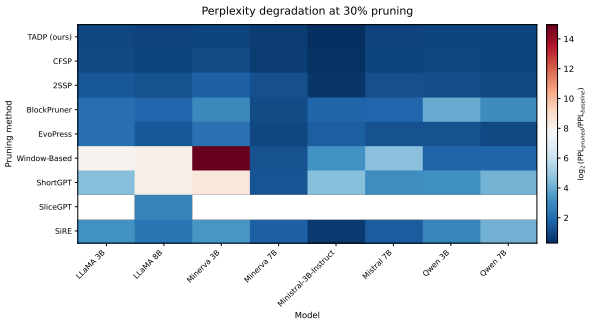
Figure 3: Accuracy retention heatmap across pruning methods and model architectures. Values represent the average retained zero-shot accuracy relative to the corresponding unpruned baseline. Dark red indicates performance close to the original baseline, while dark blue corresponds to larger accuracy degradation.



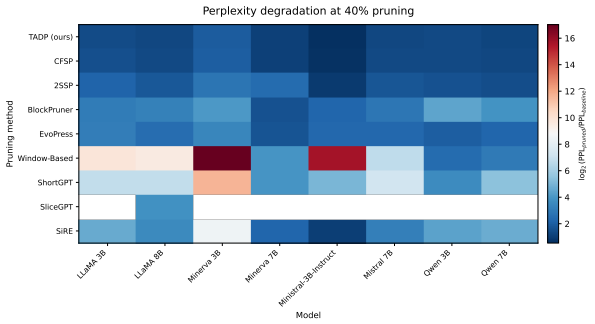
(a) 10% of reduction



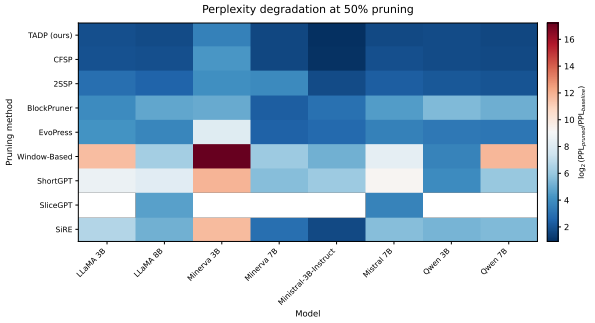
(b) 20% of reduction



(c) 30% of reduction



(d) 40% of reduction



(e) 50% of reduction

Figure 4: Perplexity degradation heatmaps across pruning methods and model architectures at different pruning levels. Values are computed as $\log_2(\text{PPL}_{pruned}/\text{PPL}_{baseline})$, where lower values indicate smaller degradation relative to the original model. Dark blue corresponds to perplexity values closer to the baseline, while dark red indicates stronger degradation after pruning.