GLIALLM: BEYOND NEURONAL SPOTLIGHT — GLIA-INSPIRED STRUCTURAL LEARNING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) typically rely on parameter scaling while maintaining fixed architectures, which leads to considerable redundancy in both model parameters and computation. Inspired by astrocyte-mediated bidirectional synaptic plasticity and microglia-mediated selective synaptic pruning, we propose the artificial Neuron–Astrocyte–Microglia (NAM), a new computational model for neuron-level structure learning in neural networks. Building on NAM, we develop the GliaLLM framework, in which Astrocyte Net (AS-Net) and Microglia Net (MG-Net) jointly learn layer- and projection-specific continuous scaling factors and binary gates from the LLM's linear-layer weights during training. These modulation signals are then computed once offline and folded into the LLM to produce a deployable pruned subnetwork. We introduce a three-stage alternating optimization strategy to jointly learn model parameters and network structure. Relative to the pretrained Llama-style-1.1B baseline, GliaLLM achieves a 23.1% parameter reduction ratio (PRR), while incurring only a 0.006 decrease in aggregate $\mathrm{acc_norm}$ across six benchmarks. These results demonstrate that glia-inspired neuron-level structure learning can yield deployable LLM subnetworks that lower peak GPU memory usage across workloads and reduce TTFT for long-input inference.

1 INTRODUCTION

Transformer-based large language models (LLMs) have advanced primarily through scaling dense architectures (Kaplan et al., 2020). However, the linear projections that dominate their computational cost typically remain fixed throughout training, limiting structural adaptivity and leaving substantial redundancy in both parameters and computation (Ma et al., 2023; Ashkboos et al., 2024). This raises a central challenge: how to enable train-time structural adaptation in the linear layers that account for the majority of LLM parameters while preserving predictive performance and deployability.

Recent efforts to improve LLM efficiency have explored structured pruning, sparsification, low-rank compression, and conditional computation. While effective in their respective settings, these approaches remain largely centered on post-training compression or static priors, and their dynamic behavior is typically limited to sample-level routing rather than train-time structural adaptation (Ma et al., 2023; Ashkboos et al., 2024; Frantar & Alistarh, 2023; Sun et al., 2024; Wang et al., 2025; Shazeer et al., 2017; Fedus et al., 2022). Recent biologically inspired methods move closer to this goal by introducing glial modulation into structure learning, yet they still rely on heuristic thresholding or early hard selection, which can lead to brittle pruning decisions and leave limited opportunity for later correction (Han et al., 2023; 2025). These limitations motivate a train-time structure-learning approach that avoids premature hard decisions throughout optimization.

In the central nervous system, neuronal communication is mediated by synapses whose efficacy is continuously regulated over time. In artificial neural networks, neurons are abstracted as computation nodes and synapses as connection weights (McCulloch & Pitts, 1943). However, neuron-only models with fixed weights offer only a static view of synaptic efficacy and therefore fail to capture the long-term plasticity and structural adaptivity observed in biological circuits (Citri & Malenka, 2008), motivating a closer look at glial mechanisms that modulate and remodel synaptic connections. As shown in Fig. 1(a), astrocytes integrate local synaptic activity, are coupled across neighboring synaptic

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

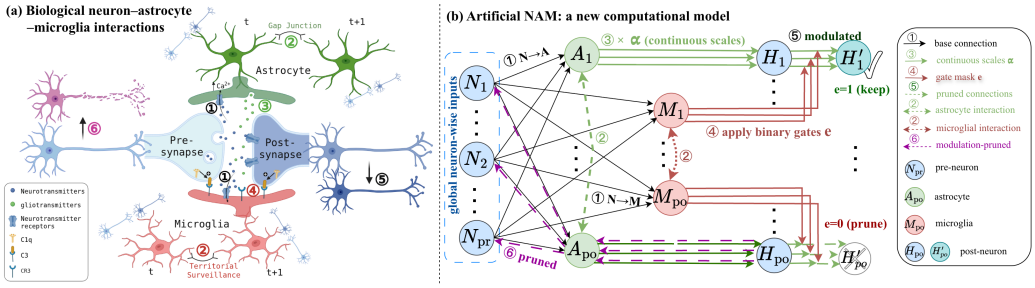


Figure 1: Mapping biological neuron–astrocyte–microglia interactions (a) to artificial NAM (b).

domains through gap junctions, and support temporally extended modulation of synaptic transmission and bidirectional plasticity (Giaume et al., 2010). Microglia monitor synaptic state, coordinate responses during remodeling, and mediate the selective elimination of weaker connections, thereby supporting activity-dependent pruning and circuit refinement (Schafer et al., 2012).

Motivated by this biological correspondence, we introduce NAM as a new computational model for neuron-level structure learning in neural networks, as illustrated in Fig. 1(b). First, the full set of connection weights is separately processed by astrocyte and microglia branches, where interactions among the internal units within each branch capture global dependencies across neuron-wise weights and support history-dependent temporal evolution (steps ①–②). Second, the two branches operate in coordination: the astrocyte branch outputs a row-wise continuous scaling factor α for bidirectional modulation, while the microglia branch outputs a binary structural gate e for selective retention or removal (steps ③–④). Finally, these modulatory signals are assigned to the corresponding neurons and applied jointly to define the effective computation path, where deactivated output neurons and their associated weight rows are excluded from the computation graph (steps ⑤–⑥).

Our contributions are summarized as follows:

- We propose the artificial Neuron–Astrocyte–Microglia (NAM), a new computational model for neuron-level structure learning that coordinates continuous scaling and binary gating to define effective neuron-level computation graphs.
- Based on NAM, we construct the GliaLLM framework.
- We introduce a three-stage alternating optimization strategy for jointly learning model parameters and modulation-induced structure.
- Across six benchmarks on Llama-style-1.1B, GliaLLM achieves a 23.10% PRR with only a 0.006 drop in aggregate acc_norm. Consistent results on Llama-style-120M and a Qwen3-0.6B further demonstrate its applicability across model scales and architectures.

2 Related Work

Compression and sparsification. Recent work on efficient LLMs spans structured pruning, weight sparsification, and low-rank compression. Structured methods such as LLM-Pruner, SliceGPT, Slim-LLM, SoBP, Sheared LLaMA, and Minitron derive deployable subnetworks or compact architectures through pruning, sometimes with lightweight recovery, while SVD-LLM compresses pretrained models via low-rank factorization (Ma et al., 2023; Ashkboos et al., 2024; Guo et al., 2025; Wei et al., 2024; Xia et al., 2024; Muralidharan et al., 2024; Wang et al., 2025). These methods are largely post-training, relatively coarse-grained, and rely on fixed pruning or rank-allocation decisions. GRASPrune (Wang et al., 2026) learns globally budgeted gates over FFN channels and KV heads on a frozen pretrained model, followed by scaling calibration. Weight-sparsification methods such as SparseGPT and Wanda, together with OWL, further show effective pruning of pretrained LLMs under one-shot or layerwise non-uniform sparsity schemes (Frantar & Alistarh, 2023; Sun et al., 2024; Yin et al., 2024). However, they mainly perform post hoc sparsification under predefined criteria, and their practical acceleration at LLM scale still depends heavily on sparse kernels and hardware support. Overall, these studies reveal substantial redundancy in LLMs and motivate our investigation of joint weight–structure learning during training.

Conditional computation and biologically inspired learning. Mixture-of-experts-style models improve parameter utilization through input-dependent routing, but they optimize sample-to-expert assignment rather than interpretable neuron-level structure within linear layers (Shazeer et al., 2017; Fedus et al., 2022). AstroNet and GliaNet introduce glial modulation into structure learning, but early discrete selection can make pruning decisions brittle and hard to correct later (Han et al., 2023; 2025).

3 METHODOLOGY

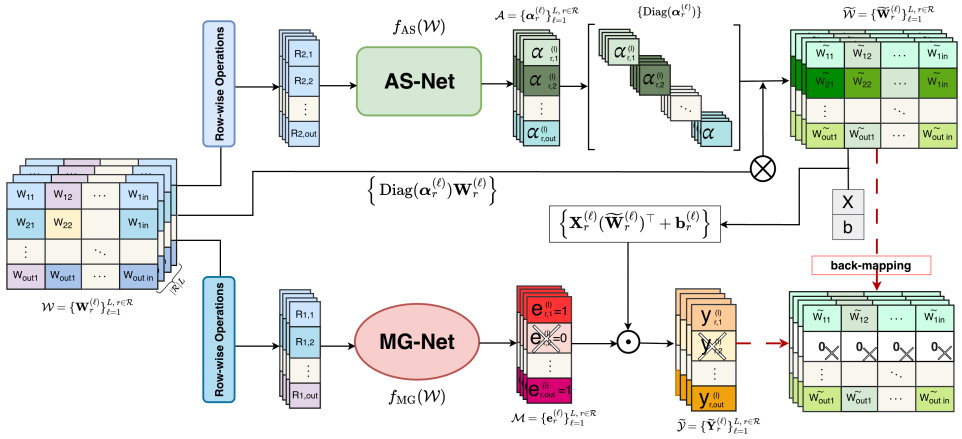


Figure 2: Overall framework of GliaLLM for neuron-level structure learning in LLMs.

3.1 GLIALLM FRAMEWORK

As illustrated in Fig. 2, GliaLLM is a framework for neuron-level structure learning in large language models, built on the NAM computational model, that reconstructs the effective computation graph by learning modulation signals from the full collection of linear-layer weights. For each layer $\ell \in \{1, \ldots, L\}$ and projection type $r \in \mathcal{R} = \{\mathrm{q}, \mathrm{k}, \mathrm{v}, \mathrm{o}, \mathrm{up}, \mathrm{gate}, \mathrm{down}\}$ in the LLM, let $\mathbf{W}_r^{(\ell)} \in \mathbb{R}^{d_{\mathrm{out}}^{(r)} \times d_{\mathrm{in}}^{(r)}}$ denote the corresponding weight matrix, and let $\mathcal{W} = \{\mathbf{W}_r^{(\ell)}\}_{\ell=1}^{L,\, r\in\mathcal{R}}$ collect all such matrices. Since each output neuron in a linear projection is parameterized by a weight row, GliaLLM implements neuron-level modulation via row-wise matrix operations.

Astrocyte Net (AS-Net) and Microglia Net (MG-Net) implement their respective branch-specific mappings over the full collection of LLM linear-layer weights $\mathcal{W}$: $\mathcal{A} = \{\boldsymbol{\alpha}_r^{(\ell)}\}_{\ell=1}^{L,\, r\in\mathcal{R}} = f_{\mathrm{AS}}(\mathcal{W})$, $\mathcal{M} = \{\mathbf{e}_r^{(\ell)}\}_{\ell=1}^{L,\, r\in\mathcal{R}} = f_{\mathrm{MG}}(\mathcal{W})$, where AS-Net emulates astrocytic gliotransmitter-mediated modulation of synaptic activity via continuous scaling signals $\mathcal{A}$, whereas MG-Net emulates microglial recognition and phagocytic clearance of low-activity or weak synapses via binary gates $\mathcal{M}$.

The generated glial signals $\mathcal{A}$ and $\mathcal{M}$ are assigned to each layer–projection and applied sequentially to the corresponding weight matrices via continuous scaling $\boldsymbol{\alpha}_r^{(\ell)}$ and binary gating $\mathbf{e}_r^{(\ell)}$. Given input $\mathbf{X}_r^{(\ell)} \in \mathbb{R}^{B\times T\times d_{\mathrm{in}}^{(r)}}$ and bias $\mathbf{b}_r^{(\ell)} \in \mathbb{R}^{d_{\mathrm{out}}^{(r)}}$, the neuron-level modulation for each (ℓ, r) is defined in Eq. (1). Following the row-wise scheme, $\alpha_{r,i}^{(\ell)}$ continuously scales the contribution strength of neuron i, whereas $e_{r,i}^{(\ell)}$ controls its retention in the effective computation graph.

$$\widetilde{\mathbf{W}}_r^{(\ell)} = \mathrm{Diag}(\boldsymbol{\alpha}_r^{(\ell)})\mathbf{W}_r^{(\ell)}, \qquad \widetilde{\mathbf{Y}}_r^{(\ell)} = \big(\mathbf{X}_r^{(\ell)}(\widetilde{\mathbf{W}}_r^{(\ell)})^\top + \mathbf{b}_r^{(\ell)}\big) \odot \mathbf{e}_r^{(\ell)} \tag{1}$$

The above describes GliaLLM's generation and application of modulation signals to linear-layer weights. The next subsection specifies how these signals are produced by AS-Net and MG-Net.

3.2 AS-NET & MG-NET

AS-Net and MG-Net parameterize the two abstract mappings, $f_{\mathrm{AS}}(\cdot)$ and $f_{\mathrm{MG}}(\cdot)$, introduced in Sec. 3.1. The two branches comprise the same four modules: a Glial-Synaptic Local Encoder

(GS-LE), an Inter-Glial Global Spatial Coupler (IG-GSC), an Inter-Glial Temporal State Evolver (IG-TSE), and a Glial-Synaptic Decoder (GS-DEC), while maintaining independent parameters. They differ in their branch-specific input statistics and final output activations: AS-Net produces bounded continuous scaling factors, whereas MG-Net produces binary structural gates.

Both branches operate on row-wise statistics of the LLM's linear-layer weights $\mathcal{W}$. For each $\mathbf{W}_r^{(\ell)}$, we define $\mathcal{R}_p(\mathbf{W}_r^{(\ell)}) = [\|\mathbf{w}_1\|_p, \ldots, \|\mathbf{w}_{d_{\text{out}}^{(r)}}\|_p]^\top$, and let $\mathcal{R}_p(\mathcal{W})$ denote the row-wise statistics over all layers and projection types. AS-Net uses the row-wise ℓ_2 norm, $\mathcal{R}_2(\mathcal{W})$, to characterize row strength, whereas MG-Net uses the row-wise ℓ_1 sum, $\mathcal{R}_1(\mathcal{W})$, to assess row importance.

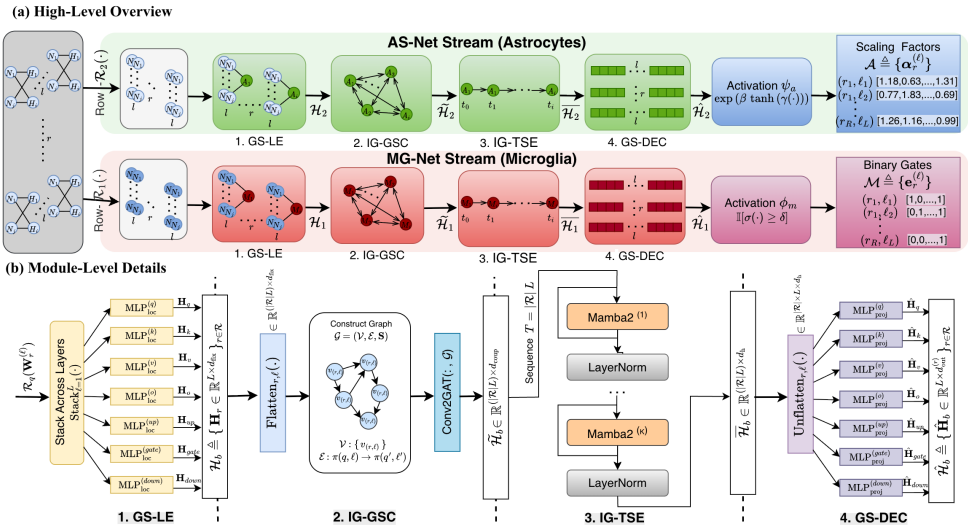


Figure 3: The architecture of AS-Net and MG-Net for glial signal generation.

3.2.1 Glial-Synaptic Local Encoder

Individual astrocytes and microglia respectively integrate activity information and aggregate synaptic-state cues within localized synaptic islands. Inspired by this non-overlapping, function-specific sensing scheme, GliaLLM first groups row statistics $\mathcal{R}_p(\mathbf{W}_r^{(\ell)})$ with the same projection type r across layers, allowing each glial unit to encode functionally homogeneous neuron groups. Since different projection types serve distinct roles in LLMs, each same-type cross-layer stack $\text{Stack}_{\ell=1}^{L}(\cdot)$ is encoded by a type-specific $\text{MLP}_{\text{loc}}^{(r)}(\cdot)$ and projected to a shared latent width d_{fix}. The local glial-state encoding is defined as

$$\mathcal{H}_b = \{\mathbf{H}_r\}_{r\in\mathcal{R}} = \left\{\text{MLP}_{\text{loc}}^{(r)}\left(\text{Stack}_{\ell=1}^{L}\left(\mathcal{R}_p(\mathbf{W}_r^{(\ell)})\right)\right) \in \mathbb{R}^{L\times d_{\text{fix}}} \;\middle|\; r \in \mathcal{R}\right\}, \quad b \in \{2,1\} \tag{2}$$

Here, $b = 2$ and $b = 1$ correspond to the AS-Net and MG-Net branches, respectively.

3.2.2 Inter-Glial Global Spatial Coupler

The Glial-Synaptic Local Encoder maps same-type weight groups to type-specific glial states with cross-layer structural evidence, but leaves them uncoordinated over the full layer–type topology. Motivated by inter-glial communication, where astrocytes are coupled via gap-junction networks and microglia coordinate through neighborhood-level interactions, we instantiate a graph-based coordination stage $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{S})$ over glial units to model dependencies spanning layers and projection types. Concretely, we first flatten $\mathcal{H}_b$ over projection types and layers into a node-feature matrix whose rows are layer–type glial nodes $v_{r,\ell}$: $\text{Flatten}_{r,\ell}(\mathcal{H}_b) \in \mathbb{R}^{(|\mathcal{R}|L)\times d_{\text{fix}}}$. The edge set $\mathcal{E}$ and edge-weight set $\mathbf{S}$ encode topological adjacency and coupling strength between the associated weight groups. We then apply a graph-attention operator Conv2GAT (Velickovic et al., 2018) to aggregate neighboring glial states into globally coordinated layer–type glial states. Let d_{coup} denote

the dimension of the spatially coupled glial state. We obtain

$$\widetilde{\mathcal{H}}_b = \mathrm{Conv2GAT}\left(\mathrm{Flatten}_{r,\ell}(\mathcal{H}_b);\ \mathcal{G}\right) \in \mathbb{R}^{(|\mathcal{R}|L)\times d_{\text{coup}}} \tag{3}$$

3.2.3 Inter-Glial Temporal State Evolver

Glial modulation is not instantaneous but history-dependent: astrocytic effects can accumulate across coupled synaptic domains through intercellular Ca^{2+}-dependent signaling, whereas microglial responses are conditioned on prior surveillance and synaptic-state cues, motivating an explicit model of temporal glial state evolution. We order the spatially coupled representations $\widetilde{\mathcal{H}}_b$ over layer–type units into a fixed sequence $\mathbf{H}_b^{(0)} = \mathrm{Seq}_\pi(\widetilde{\mathcal{H}}_b) \in \mathbb{R}^{(|\mathcal{R}|L)\times d_{\text{coup}}}$ with $T = |\mathcal{R}|L$ and $t = \pi(r,\ell)$, and model history-dependent state propagation and recurrent updates by stacking K_{tse} Mamba-2 (Dao & Gu, 2024) blocks to obtain a temporally evolved glial representation with hidden dimension d_{h}

$$\mathbf{H}_b^{(k)} = \mathrm{LN}^{(k)}\left(\mathbf{H}_b^{(k-1)} + \mathrm{Mamba2}^{(k)}(\mathbf{H}_b^{(k-1)})\right) \in \mathbb{R}^{(|\mathcal{R}|L)\times d_{\text{h}}}, \quad k = 1,\dots,K_{\text{tse}} \tag{4}$$

3.2.4 Glial-Synaptic Decoder

After state evolution, the representation remains in a shared glial latent space and is not yet neuron-executable. As a functional analogue of the transition from global glial states to local synaptic preparatory representations, the glial–synaptic decoder takes the temporally evolved state $\overline{\mathcal{H}}_b = \mathbf{H}_b^{(K_{\text{tse}})}$ as input and restores its layer–type structured organization via $\mathrm{Unflatten}_{r,\ell}(\overline{\mathcal{H}}_b) \in \mathbb{R}^{|\mathcal{R}|\times L\times d_{\text{h}}}$. For each type r, a type-specific decoder $\mathrm{MLP}_{\text{proj}}^{(r)}(\cdot)$ maps the restored local state to the corresponding neuron-wise output space $\mathbb{R}^{d_{\text{out}}^{(r)}}$, yielding branch-specific pre-activation states $\hat{\mathbf{H}}_b$ from which executable glial signals are instantiated.

$$\hat{\mathbf{H}}_b = \mathrm{MLP}_{\text{proj}}^{(r)}\Big(\mathrm{Unflatten}_{r,\ell}(\overline{\mathcal{H}}_b)[r,l,:]\Big) \in \mathbb{R}^{L\times d_{\text{out}}^{(r)}}, \qquad b \in \begin{cases} 1 & \Rightarrow f_m(\cdot) \\ 2 & \Rightarrow f_a(\cdot) \end{cases} \tag{5}$$

3.2.5 Branch-Specific Output Activation

Astrocytic gliotransmission can bidirectionally regulate synaptic plasticity toward LTP or LTD, whereas microglia selectively remove weak or aberrant synapses via complement-mediated phagocytosis (Henneberger et al., 2010; Paolicelli et al., 2011). Motivated by this distinction, we use branch-specific output activations to assign distinct functional semantics to the decoded neuron-wise states $\hat{\mathbf{H}}_b$, transforming them into executable modulation signals. For AS-Net, the activation ψ_a maps $\hat{\mathbf{H}}_2[\ell,:]$ to bounded positive scaling coefficients, enabling bidirectional yet numerically stable multiplicative modulation of neuron-wise responses.

$$\boldsymbol{\alpha}_r^{(\ell)} = \psi_a\big(\hat{\mathbf{H}}_2[\ell,:]\big) = \exp\Big(\beta\tanh\big(\gamma\hat{\mathbf{H}}_2[\ell,:]\big)\Big) \in \big[\exp(-\beta),\exp(\beta)\big]^{d_{\text{out}}^{(r)}} \tag{6}$$

For MG-Net, the activation ϕ_m maps $\hat{\mathbf{H}}_1[\ell,:]$ to discrete binary gates through sigmoid thresholding at δ, with gradients approximated during training using the straight-through estimator (STE).

$$\mathbf{e}_r^{(\ell)} = \phi_m\big(\hat{\mathbf{H}}_1[\ell,:]\big) = \mathbb{I}\Big[\sigma\big(\hat{\mathbf{H}}_1[\ell,:]\big) \geq \delta\Big] \in \{0,1\}^{d_{\text{out}}^{(r)}}. \tag{7}$$

These branch-specific outputs instantiate $\mathcal{A}$ and $\mathcal{M}$ and are then applied via Eq. (1).

3.3 Training and Optimization

Training GliaLLM is formulated as a joint optimization problem in which the LLM weights $\mathcal{W}$ govern task fitting and the glial modulation signals $(\mathcal{A},\mathcal{M})$ induce the effective computation structure. Given training inputs $\mathbf{X}$ and labels $\mathbf{Y}$, we first train the backbone language model from scratch, whose forward function $\mathcal{N}(\cdot;\mathcal{W})$ is defined without glial modulation, to obtain the initial neuronal circuit $\mathcal{W}^{(0)}$ on which glial modulation subsequently operates:

$$\mathcal{W}^{(0)} = \arg\min_{\mathcal{W}} \mathcal{L}_{\text{LM}}\big(\mathcal{N}(\mathbf{X};\mathcal{W}),\mathbf{Y}\big) \tag{8}$$

To reflect the closed-loop remodeling dynamics inspired by neuronal circuit initialization, astrocyte-mediated continuous modulation, microglia-mediated discrete clearance, and neuronal homeostatic compensation, we approximate this coordinated modulation mechanism via a three-stage alternating optimization scheme repeated over multiple cycles, each lasting K_{alt} steps. In the c-th cycle, given the current backbone weights $\mathcal{W}^{(c)}$, AS-Net and MG-Net parameterized by Θ_{AS} and Θ_{MG} produce the continuous modulation $\mathcal{A}^{(c)} = \mathcal{A}_{\Theta_{\text{AS}}}(\mathcal{W}^{(c)})$ and the discrete binary selection $\mathcal{M}^{(c)} = \mathcal{M}_{\Theta_{\text{MG}}}(\mathcal{W}^{(c)})$ according to Eqs. (6) and (7), respectively. Applying these two signals jointly via Eq. (1) yields the modulated backbone weights for the next cycle $\widetilde{\mathcal{W}}^{(c+1)} = \text{Mod}\left(\mathcal{W}^{(c)}, \mathcal{A}^{(c)}, \mathcal{M}^{(c)}\right)$. We use the same language-modeling task loss $\mathcal{L}_{\text{LM}}$ in all three stages and minimize with respect to Θ_{AS}, Θ_{MG}, or $\mathcal{W}$ separately, while freezing the other two parameters when they are not dominant. Specifically,

Stage I: Fix $\mathcal{W}^{(c)}$ and Θ_{MG}, and update only the AS-Net parameters Θ_{AS} by minimizing Eq. (9):

$$\min_{\Theta_{\text{AS}}} \mathcal{L}_{\text{AS}}^{(c)}(\Theta_{\text{AS}}) \triangleq \mathcal{L}_{\text{LM}}\Big(\mathcal{N}\big(\mathbf{X}; \text{Mod}(\mathcal{W}^{(c)}, \mathcal{A}_{\Theta_{\text{AS}}}(\mathcal{W}^{(c)}), \mathcal{M}^{(c)})\big), \mathbf{Y}\Big) \tag{9}$$

Stage II: Fix $\mathcal{W}^{(c)}$ and Θ_{AS}, and update only the MG-Net parameters Θ_{MG}. Minimize Eq. (10):

$$\min_{\Theta_{\text{MG}}} \mathcal{L}_{\text{MG}}^{(c)}(\Theta_{\text{MG}}) \triangleq \mathcal{L}_{\text{LM}}\Big(\mathcal{N}\big(\mathbf{X}; \text{Mod}(\mathcal{W}^{(c)}, \mathcal{A}^{(c)}, \mathcal{M}_{\Theta_{\text{MG}}}(\mathcal{W}^{(c)}))\big), \mathbf{Y}\Big) \tag{10}$$

Stage III: Fix the glial-network parameters Θ_{AS} and Θ_{MG}, apply the serial modulation to the backbone using $\mathcal{A}^{(c)}$ and $\mathcal{M}^{(c)}$ obtained from Stages I and II, and then minimize Eq. (11):

$$\min_{\mathcal{W}} \mathcal{L}_{\text{LM}}^{(c)}(\mathcal{W}) \triangleq \mathcal{L}_{\text{LM}}\Big(\mathcal{N}\big(\mathbf{X}; \text{Mod}(\mathcal{W}, \mathcal{A}^{(c)}, \mathcal{M}^{(c)})\big), \mathbf{Y}\Big) \tag{11}$$

Together, the three stages form one alternating optimization cycle, as summarized in Algorithm 1. After training, GliaLLM is converted into a smaller static LLM subnetwork by applying the final AS-Net/MG-Net signals to the LLM weights in a single offline forward pass according to Eq. (1), followed by neuron-level physical pruning, with AS-Net and MG-Net discarded at deployment.

Algorithm 1 Three-stage alternating optimization of GliaLLM

Input: $\mathbf{X}$, $\mathbf{Y}$; backbone parameters $\mathcal{W}$; AS-Net Θ_{AS}; MG-Net Θ_{MG}; cycles C; steps per stage K_{alt}.
Output: $\mathcal{W}^*$, Θ_{AS}^*, Θ_{MG}^*.

```
Optimize L_LM(N(X; W), Y) → W^(0) via Eq. (8)
for c = 0 to C − 1 do
    Stage I: update Θ_AS for K_alt steps with W^(c) and Θ_MG frozen via Eq. (9)
    A^(c) ← A_{Θ_AS}(W^(c))
    Stage II: update Θ_MG for K_alt steps with W^(c) and Θ_AS frozen via Eq. (10)
    M^(c) ← M_{Θ_MG}(W^(c))
    Stage III: update W for K_alt steps under frozen A^(c) and M^(c) via Eq. (11)
    W^(c+1) ← W
end for
```

4 Experiments

We evaluate GliaLLM from three complementary perspectives: accuracy retention, structural compression, and practical deployment efficiency. Under the same token budget as dense pretraining, we compare GliaLLM with representative baselines on Llama-style-1.1B and further examine its applicability across model scales and architectures using Llama-style-120M and Qwen3-0.6B. We additionally analyze the learned structural patterns and conduct comprehensive ablation studies.

4.1 Experimental Setup

Our main experiments are conducted on Llama-style-1.1B, a $L = 24$ layers dense Transformer with hidden size $d = 2048$, grouped-query attention (GQA) (Ainslie et al., 2023) ($n_q = 8$ query heads and $n_{kv} = 2$ key-value heads), and FFN dimension $d_{ffn} = 5504$. GliaLLM takes the full set of LLM

Table 1: Aggregate and task-wise acc_norm of GliaLLM on six English benchmarks: (a) comparison with dense baselines at Llama-style-1.1B, (b) cross-scale evaluation at Llama-style-120M, and (c) cross-architecture evaluation on Qwen3-0.6B.

Model/Setting	Overall	ARC	HellaSwag	PIQA	OpenBookQA	WinoGrande	MMLU
(a) 1.1B scale							
Llama-style-1.1B (Dense baseline)	0.358	0.509	0.551	0.727	0.336	0.509	0.340
LLM-Pruner (Ma et al., 2023)	0.309	0.389	0.458	0.670	0.280	0.500	0.294
Wanda (Sun et al., 2024)	**0.354**	0.503	0.546	**0.735**	0.314	**0.520**	**0.337**
SVD-LLM (Wang et al., 2025)	0.293	0.359	0.412	0.625	0.256	0.513	0.279
GRASPrune (Wang et al., 2026)	0.308	0.407	0.453	0.666	0.296	0.508	0.293
Sheared LLaMA (Xia et al., 2024)	0.344	0.480	0.508	0.711	**0.316**	0.519	0.327
GliaLLM (ours)	0.352	**0.514**	**0.558**	0.711	0.308	0.513	0.334
(b) 120M scale							
Llama-style-120M (Dense baseline)	0.294	0.372	0.355	0.640	0.258	0.505	0.281
GliaLLM (ours)	0.292	0.375	0.346	0.612	0.276	0.505	0.278
(c) Qwen3 architecture							
Qwen3-0.6B (Dense baseline)	0.329	0.470	0.476	0.676	0.300	0.509	0.313
GliaLLM (ours)	0.327	0.466	0.477	0.669	0.298	0.513	0.311

Table 2: Parameter reduction ratios (PRR, %) in GliaLLM at 1.1B and 120M scales.

Scale	q/k	v/o	$up/gate/down$	**Global**	Retained Params
1.1B	0.85%	0.28%	30.08%	**23.10%**	0.8681B
120M	11.33%	2.86%	28.42%	**23.35%**	90.54M

Table 3: Type-wise statistics of the AS-Net scaling factor α in the GliaLLM at 1.1B checkpoint.

Statistic	q	k	v	o	up	$gate$	$down$
Mean α	0.651	0.945	0.796	0.533	1.243	1.172	0.536
Std. α	0.342	0.635	0.405	0.117	0.507	0.635	0.114

linear-projection weights as input and is trained on the same 110B-token mixture as the dense baseline: FineWeb (Penedo et al., 2024), FineMath (HuggingFaceTB, 2025), Stack-Edu-Code (Kocetkov et al., 2023), and DCLM (ML Foundations, 2024). Unless otherwise stated, we use the default configuration of GliaLLM with $\beta = 0.7$, $\delta = 0.5$, and $K_{\text{alt}} = 10$. Further implementation details are provided in Appendix A.1, with data composition and evaluation protocols detailed in Appendix A.2.

4.2 Main Results

Overall Performance. Table 1(a) reports acc_norm on six English benchmarks under a unified LightEval (Hugging Face, 2024) protocol: ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), OpenBookQA (Mihaylov et al., 2018), WinoGrande (Sakaguchi et al., 2020), and MMLU (Hendrycks et al., 2021). Under the same 110B-token budget as dense pretraining, GliaLLM-1.1B achieves an Overall score of 0.352, close to the dense baseline's 0.358, while attaining a PRR of 23.10% (Table 2). GliaLLM also slightly exceeds the dense baseline on ARC, HellaSwag, and WinoGrande. Among the compression baselines, it outperforms LLM-Pruner, SVD-LLM, GRASPrune, and Sheared LLaMA (Ma et al., 2023; Wang et al., 2025; 2026; Xia et al., 2024) in Overall performance. Near the operating PRR, it remains competitive with Wanda (Sun et al., 2024) while yielding a hardware-aligned compact subnetwork deployable with standard dense operators (Appendix A.8). Table 1(b) shows a consistent result at the 120M scale: GliaLLM achieves 0.292 Overall versus 0.294 for the dense baseline, together with a 23.35% PRR. Finally, Table 1(c) shows that GliaLLM attains a 22.61% PRR on Qwen3-0.6B while retaining an Overall score of 0.327, only 0.002 below the dense baseline, demonstrating that GliaLLM maintains a favorable quality–compression trade-off across distinct model architectures. Multi-seed results and paired-bootstrap analyses are provided in Appendix A.4.

Deployment Efficiency. For hardware-efficient deployment, we first enforce shape consistency by intersecting retained indices within coupled q/k and $up/gate$ projections and propagating the relevant indices across the $v \to o$ and $up/gate \to down$ dependencies, yielding a static GliaLLM-1.1B subnetwork for physical export. We then align the retained FFN widths to multiples of 128, changing the effective PRR from 23.10% before hardware alignment to 21.04% in the exported compact model. Under identical FP16 inference settings on a single NVIDIA RTX 3090, the compact model reduces peak GPU memory by 16.36–20.10% across six workloads. For input lengths of at least 512 tokens, it reduces time to first token (TTFT) by 12.95–18.34% and improves prefill throughput by 14.88–22.46%. Decoding latency remains broadly comparable after export, with workload-dependent end-to-end latency changes ranging from -3.13% to $+2.86\%$. Thus, the deployment

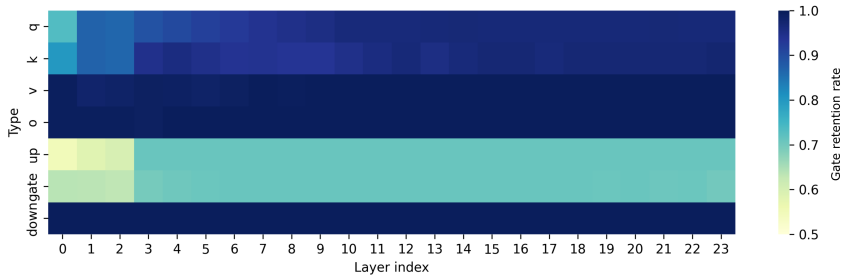


Figure 4: Layer–type heatmap of the MG-Net gate retention rate in GliaLLM-1.1B. Structural selection is strongly heterogeneous across both depth and projection type.

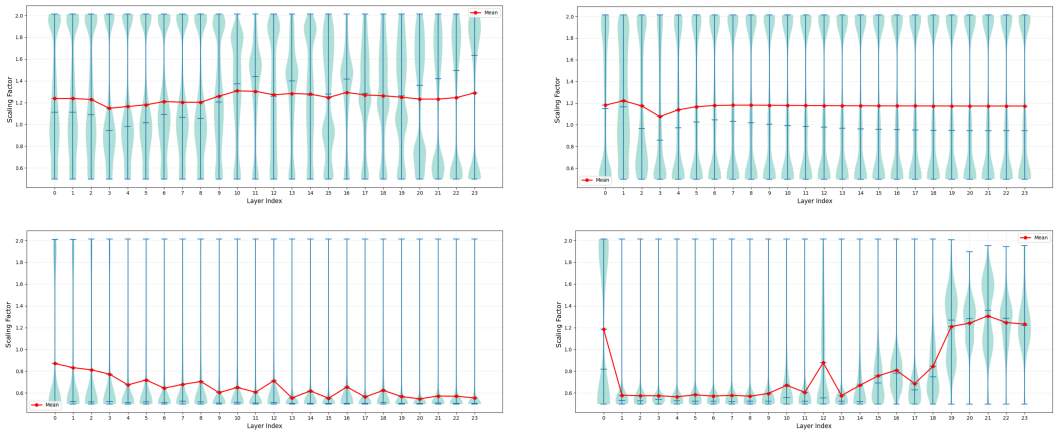


Figure 5: Layer-wise distributions of the AS-Net scaling factor α in GliaLLM-1.1B for up and $gate$ (top) and q and v (bottom) projections.

benefits primarily arise from reduced memory usage and accelerated prefill. Full results and the measurement protocol are provided in Appendix A.5.

Learned Structural Preference. The compression is highly non-uniform, with $up/gate/down$ accounting for most of the structural removal and reaching a PRR of 30.08%, whereas q/k and v/o are pruned by only 0.85% and 0.28%, respectively, revealing a learned structural preference. Fig. 4 further illustrates this pattern, showing the strongest sparsification in the FFN path, especially $up/gate$, with q/k in an intermediate regime and $v/o/down$ comparatively stable. Beyond projection type, this non-uniformity is also evident across layer depth: shallow layers are more readily sparsified, whereas middle-to-deep layers retain more stable computation. These results suggest that structural compression in GliaLLM is distributed non-uniformly across both projection type and layer depth.

Synergistic Effect. Taken together, Table 2 and Table 3 indicate that GliaLLM learns a complementary division of labor between discrete structural selection and continuous gain compensation, yielding a stable "retain–attenuate / sparse–enhance" regime (see Appendix A.7). Specifically, the broadly retained o and $down$ projections are systematically attenuated, with mean scaling factors $\bar{\alpha} = 0.533$ and 0.536, respectively, whereas the more sparsified up and $gate$ projections are compensatorily amplified to $\bar{\alpha} = 1.243$ and 1.172. The larger standard deviations of up and $gate$ further suggest that this compensation is not a uniform rescaling, but a heterogeneous adjustment determined jointly by projection type and layer depth. Consistent with this interpretation, Fig. 5 further reveals a clear distributional contrast across layers: the FFN projections up and $gate$ are centered predominantly above unit gain and exhibit substantially broader distributions, whereas o and $down$ remain concentrated below unit gain with much smaller variance. Moreover, q is progressively attenuated with depth, while v becomes increasingly amplified in deeper layers.

Table 4: Ablation results of GliaLLM: (a) AS-Net branch ablation at GliaLLM-120M, and (b) module and hyperparameter ablations at GliaLLM-1.1B.

Setting	PRR	Overall	ARC	HellaSwag	PIQA	OpenBook QA	Wino Grande	MMLU
(a) 120M: AS-Net branch ablation								
GliaLLM	**23.35%**	**0.2920**	0.3750	0.3460	0.6120	0.2760	0.5050	0.2780
MG-Net-only	10.30%	0.2849	0.3645	0.3320	0.6360	0.2680	0.5010	0.2716
(b) 1.1B: module and hyperparameter ablations								
GliaLLM	23.10%	**0.352**	0.514	0.558	0.711	0.308	0.513	0.334
w/o IG-GSC	<u>41.59%</u>	0.275	0.307	0.332	0.569	0.244	0.509	0.264
w/o IG-TSE	**41.93%**	0.322	0.441	0.458	0.687	0.282	0.517	0.306
$\delta = 0.6$	35.71%	0.339	0.465	0.522	0.714	0.292	0.527	0.322
$K_{\text{alt}} = 30$	28.86%	<u>0.343</u>	0.499	0.526	0.703	0.300	0.524	0.326

4.3 Ablation Studies

AS-Net branch ablation. At 120M, we fix $\alpha \equiv 1$ to remove continuous scaling in AS-Net while retaining MG-Net for structural gating. Table 4(a) shows that under otherwise identical settings, this ablation reduces Overall $\mathrm{acc_norm}$ from 0.2920 to 0.2849 and PRR from 23.35% to 10.30%. Thus, without continuous scaling, the model not only performs worse but also learns a substantially less compact structure, demonstrating that AS-Net is essential for enabling effective pruning while preserving predictive quality.

Module ablation. Table 4(b) evaluates the roles of IG-GSC and IG-TSE in GliaLLM-1.1B. Removing either module leads to a marked drop in performance, with Overall $\mathrm{acc_norm}$ decreasing from 0.352 to 0.275 without IG-GSC and to 0.322 without IG-TSE. At the task level, ARC, HellaSwag, and MMLU exhibit the most consistent degradation, whereas WinoGrande changes little. Both ablations sharply increase the global PRR, from 23.10% in the full GliaLLM model to 41.59% and 41.93%, respectively. Projection-wise PRRs in Appendix Table 9 reveal distinct pruning effects for the two ablations: removing IG-GSC extends pruning into attention projections, especially v/o, whereas removing IG-TSE primarily intensifies FFN-side pruning. Together, these results suggest that the distribution of pruning is as important as its overall magnitude and highlight the complementary roles of IG-GSC in cross-region coordination and IG-TSE in state accumulation and temporal modeling, both of which help prevent over-pruning and maintain coordinated structure learning.

Hyperparameter ablation. The hyperparameter-ablation rows of Table 4(b) analyze the sensitivity of GliaLLM to the gate threshold δ and the alternating-update interval K_{alt}. Increasing δ from 0.5 to 0.6 most strongly shifts the quality–compression balance, decreasing Overall $\mathrm{acc_norm}$ from 0.352 to 0.339 while increasing the global PRR from 23.10% to 35.71%. This confirms that δ is the primary control on pruning aggressiveness. By comparison, increasing K_{alt} to 30 yields a smaller but still consistent degradation, with Overall $\mathrm{acc_norm}$ falling to 0.343 and PRR rising to 28.86%, indicating that overly infrequent alternation impairs coordinated structure learning. This interpretation is also consistent with the biological motivation of GliaLLM: δ plays the role of a microglial clearance threshold, whereas a larger K_{alt} prolongs single-phase dominance and weakens closed-loop feedback between astrocytes and microglia. Overall, these results justify the default setting of $\delta = 0.5$ and $K_{\text{alt}} = 10$ as a robust trade-off between performance retention and deployable compression.

5 Conclusion

We propose the artificial Neuron–Astrocyte–Microglia (NAM), a computational model for neuron-level structure learning in neural networks, and develop GliaLLM based on NAM for large language models. During training, GliaLLM jointly optimizes the LLM, AS-Net, and MG-Net through a three-stage alternating strategy, ultimately yielding a deployable pruned subnetwork. On Llama-style-1.1B, GliaLLM achieves a 23.10% PRR while remaining close to the dense baseline in Overall $\mathrm{acc_norm}$ (0.352 vs. 0.358). Overall, glia-inspired neuron-level modulation is an effective principle for structural learning in LLMs, producing physically compact subnetworks with measurable memory and prefill-efficiency benefits.

AI USE STATEMENT

Generative AI tools, including OpenAI ChatGPT, were used during manuscript preparation to assist with language editing, improve readability and organization, check internal consistency, adapt the LaTeX source and bibliography, and support literature retrieval and discovery. AI-assisted searches were used only to identify candidate related work; the authors independently consulted the original publications and verified the accuracy and relevance of all cited sources. All AI-assisted text and formatting revisions were reviewed by the authors and checked against the implementation, experimental records, and cited sources; the authors take responsibility for the final manuscript.

ETHICS STATEMENT

Information on external assets, licenses, and terms of use is provided in Appendix A.10.

REPRODUCIBILITY STATEMENT

The experimental setup is described in the main text. Additional implementation details, data and evaluation settings, and compute resources are provided in Appendices A.1, A.2, and A.3, respectively. Physical export details and deployment measurements are provided in Appendix A.5. We plan to publicly release the GliaLLM implementation and the associated training, pruning, and evaluation configurations upon acceptance. During peer review, we may provide anonymized code to the reviewers and area chair upon request.

REFERENCES

Joshua Ainslie et al. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress Large Language Models by Deleting Rows and Columns. In *International Conference on Learning Representations (ICLR)*, 2024.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about Physical Commonsense in Natural Language. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Ami Citri and Robert C. Malenka. Synaptic plasticity: multiple forms, functions, and mechanisms. *Neuropsychopharmacology*, 33(1):18–41, 2008.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. arXiv:1803.05457, 2018.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. arXiv:2405.21060, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Re. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the Lottery: Making All Tickets Winners. In *International Conference on Machine Learning (ICML)*, 2020.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Elias Frantar and Dan Alistarh. SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot. In *International Conference on Machine Learning (ICML)*, 2023.

Christian Giaume, Annette Koulakoff, Lisa Roux, David Holcman, and Nathalie Rouach. Astroglial networks: a step further in neuroglial and gliovascular interactions. *Nature Reviews Neuroscience*, 11:87–99, 2010.

Jingyao Gong. MiniMind: Train a Tiny LLM from Scratch. GitHub repository, 2024. URL https://github.com/jingyaogong/minimind. Accessed 2026.

Jialong Guo, Xinghao Chen, Yehui Tang, and Yunhe Wang. SlimLLM: Accurate Structured Pruning for Large Language Models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20766–20776. PMLR, 2025. URL https://proceedings.mlr.press/v267/guo25a.html.

Mengqiao Han, Liyuan Pan, and Xiabi Liu. AstroNet: When Astrocyte Meets Artificial Neural Network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Mengqiao Han, Liyuan Pan, and Xiabi Liu. GliaNet: Adaptive Neural Network Structure Learning with Glia-Driven. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

Christian Henneberger, Thomas Papouin, Stephane H. R. Oliet, and Dmitri A. Rusakov. Long-term potentiation depends on release of D-serine from astrocytes. *Nature*, 463:232–236, 2010.

Jordan Hoffmann et al. Training Compute-Optimal Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Hugging Face. lighteval. GitHub repository, 2024.

HuggingFaceTB. FineMath. Hugging Face Datasets, 2025.

HuggingFaceTB. smollm3-configs. Hugging Face repository, 2026. URL https://huggingface.co/datasets/HuggingFaceTB/smollm3-configs/tree/main. Accessed 2026.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models. arXiv:2001.08361, 2020.

Nikolay Kocetkov et al. The Stack: 3 TB of permissively licensed source code. *Transactions on Machine Learning Research (TMLR)*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Xinyu Ma, Gongfan Fang, and Xinchao Wang. LLM-Pruner: On the Structural Pruning of Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Warren S. McCulloch and Walter Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 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 (EMNLP)*, 2018.

ML Foundations. DCLM: DataComp for Language Models. GitHub repository, 2024.

Decebal C. Mocanu, Elena Mocanu, Peter Stone, Phuong H. Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable Training of Artificial Neural Networks with Adaptive Sparse Connectivity Inspired by Network Science. *Nature Communications*, 9:2383, 2018.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact Language Models via Pruning and Knowledge Distillation. arXiv:2407.14679, 2024.

Rosa C. Paolicelli, Giulia Bolasco, Francesca Pagani, Laura Maggi, Maria Scianni, Patrizia Panzanelli, Maurizio Giustetto, Tiago Alves Ferreira, Eva Guiducci, Laura Dumas, Davide Ragozzino, and Cornelius T. Gross. Synaptic pruning by microglia is necessary for normal brain development. *Science*, 333(6048):1456–1458, 2011.

Guilherme Penedo et al. FineWeb. Hugging Face Datasets, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Dorothy P. Schafer, Emily K. Lehrman, Amanda G. Kautzman, Ryuta Koyama, Alan R. Mardinly, Ryo Yamasaki, Richard M. Ransohoff, Michael E. Greenberg, Ben A. Barres, and Beth Stevens. Microglia sculpt postnatal neural circuits in an activity and complement-dependent manner. *Neuron*, 74(4):691–705, 2012.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey Hinton, and Jeff Dean. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations (ICLR)*, 2017.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A Simple and Effective Pruning Approach for Large Language Models. In *International Conference on Learning Representations (ICLR)*, 2024.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. SVD-LLM: Truncation-aware Singular Value Decomposition for Large Language Model Compression. In *International Conference on Learning Representations (ICLR)*, 2025.

Ziyang Wang, Jiangfeng Xiao, Chuan Xiao, Ruoxiang Li, Rui Mao, and Jianbin Qin. GRASPrune: Global Gating for Budgeted Structured Pruning of Large Language Models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10719–10736. Association for Computational Linguistics, 2026.

Jiateng Wei et al. SoBP: Structured Optimal Brain Pruning for Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning. In *International Conference on Learning Representations (ICLR)*, 2024.

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. Outlier Weighed Layerwise Sparsity (OWL): A Missing Secret Sauce for Pruning LLMs to High Sparsity. In *International Conference on Machine Learning (ICML)*, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

A Appendix

A.1 Additional Implementation Details

Unless otherwise stated, in the 1.1B-scale experiments, the AS-Net and MG-Net branches share the same architecture but use independent parameters, with 41M parameters in each branch. The intermediate dimensions are set uniformly to $d_{\text{fix}} = d_{\text{coup}} = d_h = 768$. We use a per-device micro-batch size of 3 and gradient accumulation of 10, corresponding to an effective global batch size of 240. Training uses mixed precision (BF16/FP16) throughout, with FlashAttention (Dao et al., 2022) enabled for acceleration. We optimize with AdamW (Loshchilov & Hutter, 2019) and $\text{weight_decay} = 0.01$, using $lr_{\text{as}} = lr_{\text{mg}} = 1 \times 10^{-3}$ for the glial-modulation networks and $lr_{\text{lm}} = 5 \times 10^{-4}$ for the backbone language model. The learning-rate schedule uses 3000 warmup updates followed by cosine decay with $\text{min_lr_ratio} = 0.1$.

A.2 Additional Data and Evaluation Details

The 110B-token pretraining mixture is composed of FineWeb, FineMath, Stack-Edu-Code, and DCLM with proportions of 76.9%, 7.7%, 11.6%, and 3.9%, respectively. The maximum context length is 4K, and the tokenizer uses a 32K vocabulary. For external baselines, the shared 256-example calibration set is sampled from the same in-distribution mixture used for pretraining. Unless otherwise stated, each benchmark is evaluated using a fixed random seed and a cap of 1000 sampled examples. In addition to acc_norm, we track training-time loss/PPL and gate-retention statistics for the stability and structural analyses.

A.3 Compute Resources

Under an approximately 110B-token budget, dense baseline pretraining and GliaLLM-1.1B training take 127.70 and 133.44 wall-clock hours, respectively, indicating comparable training time (Table 5). The two runs use seven and eight NVIDIA H100 (80GB) GPUs, per-GPU micro-batch sizes of 4 and 3, and approximately 71GB and 77GB of peak memory per GPU, respectively.

Table 5: Compute accounting for the designated 1.1B runs on NVIDIA H100 (80G) GPUs. Micro batch and peak memory are reported per GPU.

Training setting	GPUs	Wall-clock	GPU-hours	Micro-batch/GPU	Peak memory/GPU
Dense baseline pretraining	7	127.70 h	893.93	4	∼71GB
GliaLLM-1.1B training (ours)	8	133.44 h	1067.51	3	∼77GB

A.4 Robustness and Uncertainty Analysis

We assess robustness with respect to both training-seed variability and finite-sample evaluation uncertainty.

Training-seed variability. Table 6 reports three GliaLLM runs per scale under identical training and evaluation settings. The Overall acc_norm SD is 0.0032 at 120M and 0.0006 at 1.1B, while the corresponding PRR SDs are 0.26 and 0.08 percentage points. At 1.1B, task-wise SDs range from 0.0007 on MMLU to 0.0069 on OpenBookQA, indicating stable performance and compression across seeds.

Evaluation-set uncertainty. Holding the reported GliaLLM-1.1B and dense checkpoints fixed, 10,000 paired bootstrap resamples over 19,008 matched evaluation records yield a GliaLLM-minus-dense difference of -0.0058 in Overall acc_norm, with a 95% confidence interval of $[-0.0123, +0.0007]$. Because the interval includes zero, the aggregate difference is not statistically distinguishable from zero under evaluation-item resampling; thus, no statistically significant performance degradation is detected under this evaluation protocol.

Table 6: Results across three training seeds per scale. Summary rows report across-run means and sample standard deviations (SD; $n = 3$). Accuracy is reported on the $[0, 1]$ scale, and PRR SDs are given in percentage points.

(a) Overall performance and PRR

Scale	Seed / summary	Overall acc_norm	PRR (%)
GliaLLM-120M	123	0.2907	22.60
	1337	0.2920	23.11
	2024	0.2860	22.80
	Mean ± SD	0.2896 ± 0.0032	22.84 ± 0.26
GliaLLM-1.1B	123	0.3483	22.81
	1337	0.3495	22.96
	2026	0.3493	22.94
	Mean ± SD	0.3490 ± 0.0006	22.90 ± 0.08

(b) Task-wise performance at GliaLLM-1.1B

Seed / summary	ARC	HellaSwag	PIQA	OpenBookQA	WinoGrande	MMLU
123	0.5110	0.5410	0.7140	0.3060	0.5240	0.3305
1337	0.5135	0.5510	0.7120	0.3180	0.5140	0.3315
2026	0.5075	0.5410	0.7110	0.3060	0.5210	0.3318
Mean	0.5107	0.5443	0.7123	0.3100	0.5197	0.3312
Sample SD	0.0030	0.0058	0.0015	0.0069	0.0051	0.0007

A.5 Hardware Alignment and Inference Efficiency

Hardware-alignment scope. We compare the full-width and hardware-aligned compact versions of the same GliaLLM-1.1B checkpoint; the reference is not the pretrained dense baseline. The compact conversion folds the learned scaling into the weights, removes AS-Net and MG-Net, and aligns the retained FFN widths to multiples of 128, yielding a hardware-aligned PRR of 21.04% (23.10% before alignment), while attention and KV-cache dimensions remain unchanged.

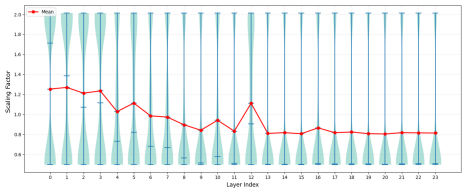
Measurement protocol. Both models are benchmarked on a single NVIDIA RTX 3090 with PyTorch 2.9.0, CUDA 12.6, FP16 eager execution, and TF32 disabled. Each of the six workloads generates 128 tokens per sequence and uses five paired runs with alternating model order, five warm-up iterations, and 30 timed iterations. Synchronized CUDA timing includes greedy argmax; E2E covers prefill and 127 decoding steps, while peak GPU memory is measured per workload.

Overall, the hardware-aligned compact model reduces peak GPU memory by 16.36–20.10% across all workloads and, for input lengths of at least 512 tokens, reduces TTFT by 12.95–18.34% while increasing prefill throughput by 14.88–22.46%. E2E latency changes remain limited to -3.13%–$+2.86\%$, indicating that the primary benefits arise from lower memory usage and faster prefill.

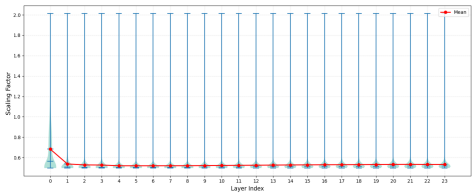
Table 7: Model-side inference measurements for GliaLLM-1.1B (full-width → compact; hardware-aligned PRR: 21.04%). E2E and peak-memory columns report relative changes, with negative values denoting reductions. All workloads generate 128 tokens per sequence.

Batch × input	TTFT (ms)	Prefill (ktok/s)	TPOT (ms)	E2E change (%)	Peak memory change (%)
1×128	$26.44 \to 26.05$	$4.84 \to 4.91$	$29.99 \to 29.83$	-0.54	-20.10
1×512	$35.35 \to 29.65$	$14.49 \to 17.31$	$30.49 \to 31.41$	$+2.86$	-19.75
1×2048	$103.61 \to 86.00$	$19.77 \to 23.81$	$29.83 \to 30.10$	$+0.42$	-18.53
1×4096	$209.94 \to 182.75$	$19.51 \to 22.41$	$30.61 \to 30.09$	-2.26	-17.03
4×512	$96.40 \to 78.72$	$21.24 \to 26.02$	$32.01 \to 31.12$	-3.13	-18.10
8×512	$174.63 \to 147.43$	$23.46 \to 27.78$	$32.03 \to 31.43$	-2.43	-16.36

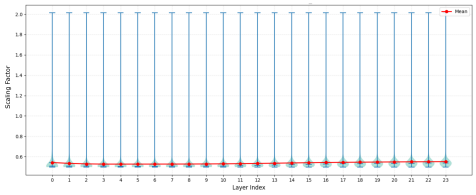
A.6 Additional Distributions of the AS-Net Scaling Factors



(a) k_proj



(b) o_proj



(c) down_proj

Figure 6: Layer-wise distributions of the AS-Net scaling factor α for additional projection types in GliaLLM.

A.7 Additional Coordination Dynamics of Selection and Scaling

Complementing the layer-wise distributions in Fig. 6, Fig. 7 tracks the training dynamics of the two modulatory signals. The mean gate retention rate $\bar{e}$ and mean scaling factor $\bar{\alpha}$ evolve toward stable late-training plateaus, indicating that the observed division of labor is learned progressively during optimization rather than induced only at the final checkpoint. Together with the compression results, these trajectories support a coordinated "retain–attenuate / sparse–enhance" regime.

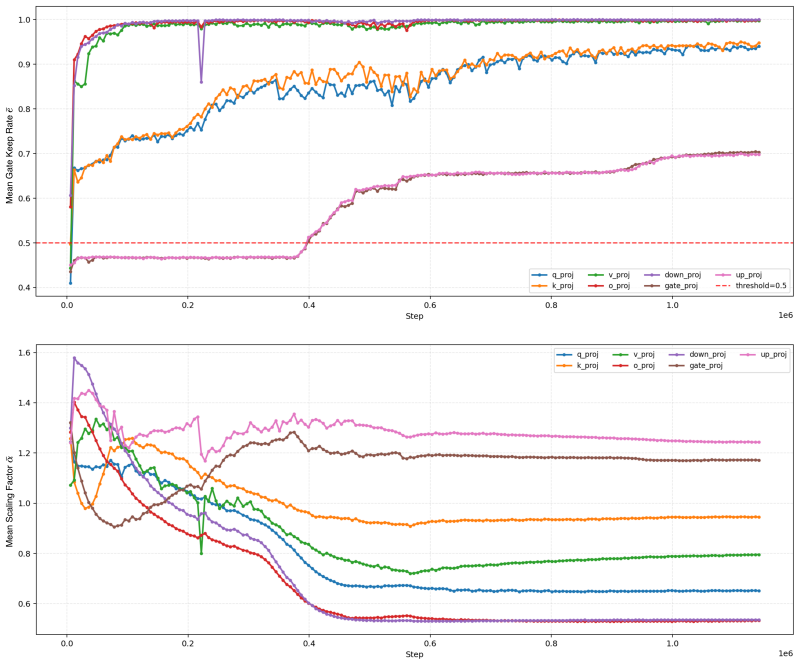


Figure 7: Training-time coordination between discrete selection and continuous scaling in GliaLLM. The trajectories show that the mean gate factor $\bar{e}$ and the mean scaling factor $\bar{\alpha}$ converge to stable late-training plateaus.

A.8 Practical Scope Relative to Wanda

GliaLLM dynamically searches for network structures during training; consequently, the pruning ratios in Table 8 arise from learned structures at different checkpoints rather than independently imposed post-training targets. We evaluate these checkpoints and compare them with Wanda (Sun et al., 2024) at approximately matched pruning ratios. Wanda applies element-wise unstructured pruning but retains the original tensor shapes under standard dense execution, so its nominal sparsity does not directly translate into memory or latency reductions without specialized sparse kernels. In contrast, GliaLLM produces hardware-aligned compact matrices that reduce memory usage and improve prefill efficiency with standard dense operators. At pruning ratios of 21.05–24.59%, the accuracy gap is only 0.14–0.70 percentage points, making GliaLLM preferable for memory-constrained, prefill-intensive, or broadly deployable inference settings when this modest trade-off is acceptable. At higher pruning ratios, however, the gap widens to 1.36–2.01 percentage points.

Table 8: Overall $\mathrm{acc_norm}$ of GliaLLM checkpoints and Wanda at approximately matched pruning ratios. GliaLLM ratios emerge from structures learned during training, whereas Wanda ratios denote post-training element-wise sparsity. Positive gaps indicate higher Wanda accuracy.

Pruning ratio (%)	GliaLLM	Wanda	Wanda − GliaLLM (pp)
21.05	0.35152	0.35290	+0.14
22.99	0.34710	0.35280	+0.57
24.59	0.34372	0.35070	+0.70
28.00	0.32930	0.34940	+2.01
36.85	0.32571	0.33930	+1.36

A.9 Additional Structural Details of Ablation Results

Table 9 reports the projection-group PRRs and removed parameter counts for the GliaLLM-1.1B module and hyperparameter ablations in Table 4(b). The module and hyperparameter comparisons share the same full GliaLLM reference. All values use the pre-hardware-alignment PRR convention defined in the experimental setup.

Table 9: Projection-group structural statistics for the GliaLLM-1.1B module and hyperparameter ablations in Table 4(b). Projection-group and global PRRs are reported before hardware alignment.

Setting	q, k	v, o	up, $gate$, $down$	Global PRR	Pruned Params
GliaLLM	0.85%	0.28%	30.08%	23.10%	0.2608 B
w/o IG-GSC	9.93%	**39.57%**	46.81%	41.59%	0.4695 B
w/o IG-TSE	1.40%	0.19%	**54.68%**	41.93%	0.4733 B
$\delta = 0.6$	9.02%	11.70%	43.58%	35.71%	0.4031 B
$K_{\mathrm{alt}} = 30$	2.02%	0.05%	37.49%	28.86%	0.3258 B

A.10 LICENSES OF EXISTING ASSETS

Table 10: External assets referenced or used in this work and their version, license, and terms-of-use information.

Asset	Citation	Version	License	Terms of use
MiniMind	(Gong, 2024)	GitHub repo `jingyaogong/minimind`, commit `1718e9a` (master; accessed 2026-05-04)	Apache-2.0	Used as the code basis for our custom Llama-style backbones. The dense baseline weights are trained from scratch; no official pretrained Llama or third-party MiniMind checkpoint is used for initialization.
FineWeb	(Penedo et al., 2024)	HF dataset `HuggingFaceFW/fineweb`, snapshot `9bb295d` (`v1.4.0`; accessed 2026-05-04)	ODC-By v1.0	Use is also subject to CommonCrawl Terms of Use.
FineMath	(HuggingFaceTB, 2025)	HF dataset `HuggingFaceTB/finemath`, snapshot `e92b25a` (main; accessed 2026-05-04)	ODC-By v1.0	Use is also subject to CommonCrawl Terms of Use.
Stack-Edu	(Kocetkov et al., 2023)	HF dataset `HuggingFaceTB/stack-edu`, snapshot `eeec5ca` (main; accessed 2026-05-04)	Mixed; see The Stack v2	The released artifact provides SWHIDs rather than file contents. Any downstream use must follow The Stack v2 terms and the original repository licenses, including attribution where applicable.
DCLM-Baseline 1.0	(ML Foundations, 2024)	HF dataset `mlfoundations/dclm-baseline-1.0`, snapshot `a3b142c` (main; accessed 2026-05-04)	CC-BY-4.0	Used under the dataset card release terms; attribution to the DCLM team is required.
LightEval	(Hugging Face, 2024)	GitHub tag `v0.13.0` (commit `7c1cd62`); PyPI package `lighteval==0.13.0`	MIT	Used under the MIT License.
SmolLM3 LightEval configs	(HuggingFaceTB, 2026)	HF dataset `HuggingFaceTB/smollm3-configs`, snapshot `8d27ec7` (main; accessed 2026-05-04)	Apache-2.0	Used as external evaluation-task configuration files; the referenced benchmark datasets remain subject to their own licenses, usage terms, and attribution requirements.