Diet Your LLM: Dimension-wise Global Pruning of LLMs via Merging Task-specific Importance Score

Anonymous ACL submission

Abstract

Large language models (LLMs) have demonstrated remarkable capabilities, but their massive scale poses significant challenges for practical deployment. Structured pruning offers a promising solution by removing entire dimensions or layers, yet existing methods face critical trade-offs: task-agnostic approaches cannot adapt to task-specific requirements, while task-aware methods require costly training to learn task adaptability. We propose DIET (**Di**mension-wise global pruning of LLMs via m**e**rging **T**ask-wise importance scores), a training-free structured pruning method that combines dimension-level granularity with task-aware selection. DIET profiles activation magnitudes across tasks using only 100 samples per task, then applies majority voting to construct a single global mask. DIET does not require large costs from pre-computation or training. Experiments on seven zero-shot benchmarks using Gemma-2 2B and 9B models demonstrate the effectiveness of DIET; for example, at 20% sparsity on Gemma-2 2B, DIET achieves near 10% average accuracy improvement, compared to previous state-of-the-art structured pruning methods. This advantage persists across various sparsity levels and model scales, positioning DIET as a practical and robust choice for structured LLM pruning.

1 Introduction

Large Language Models (LLMs) have rapidly advanced with increasing scale, yet deploying them on resource-constrained platforms remains challenging due to strict compute and memory limits. This mismatch between model size and hardware budgets has motivated approaches to reduce their computational and memory footprints. Among them, pruning is one of the representative techniques, which removes model parameters or activations (*i.e.*, masks them to zero to induce sparsity) while striving to preserve the original performance. Pruning has long been an active area of research in deep learning, and this interest has recently extended to LLMs. To make pruning practical for real-world deployment, inducing *structured* sparsity patterns, rather than simply increasing the sparsity ratio, is crucial, since unstructured pruning is often incompatible with efficient computation using sparse matrix operations.

Prior works on structured pruning in LLMs can be categorized based on (1) the granularity of pruning (*e.g.*, individual parameters, attention heads, or entire transformer blocks) and (2) whether the pruning decisions leverage task-specific information. For instance, SliceGPT (Ashkboos et al., 2024) exemplifies dimension-level, task-agnostic pruning by identifying less important dimensions through principal component analysis (PCA) of hidden embeddings using calibration data. However, its task-agnostic mask cannot adapt to task-wise variation, and PCA-based identification introduces non-trivial computational overhead. PuDDing (Wee et al., 2025), in contrast, adopts block-level, task-specific pruning by training a per-prompt router to dynamically skip selected Transformer blocks. While this approach achieves strong task-specific compression, it requires large training cost for the router and removes all attention and MLP computations within pruned blocks, limiting its generalization capability. These observations motivate us to explore an underexplored point in the design space: a *dimension-level, task-aware, training-free* pruning framework for LLMs.

In this work, we propose DIET, a framework for **Di**mension-wise global pruning of LLMs via m**e**rging **T**ask-wise importance scores. DIET requires no additional training, such as model fine-tuning or router training, for pruning; it only relies on streamed model activations from a few task-specific samples to generate a simple, deployable global mask. To be specific, for each task, DIET first profiles the outputs of MLP layers at every

Transformer block and computes a per-dimension importance vector by averaging absolute activations. It then converts the lowest-ranked dimensions into a task-specific binary pruning mask and stacks these selectors across tasks. By aggregating masks across tasks, DIET identifies dimensions with the highest cross-task agreement to achieve a target sparsity level, forming a single global pruning mask. This mask is uniformly applied to all residual-connected linear layers: input-dimension masking for embeddings, attention projections, MLP input projections, and the LM head and output-dimension masking for attention and MLP output projections, respectively.

We demonstrate the effectiveness of DIET on Gemma-2 (Team et al., 2024) (2B and 9B) at 10%, 20%, and 30% dimension sparsity under zero-shot evaluation on 7 different benchmarks. Across both model sizes and all sparsity levels, DIET significantly outperforms a recent structured pruning method and a router-based pruning approach.

Overall, our key contributions are as follows:

- ◦ We propose DIET, a dimension-wise global pruning framework that yields a single global mask which is easy to apply.
- ◦ We introduce a simple activation-based profiling requiring no additional training.
- ◦ DIET achieves comprehensive zero-shot results across two model scales, seven benchmarks, and sparsity up to 20% showing consistent gains over state-of-the-art pruning baselines.
- ◦ Empirically, DIET exhibits a clear advantage over task-wise depth pruning algorithms, achieving more than 20% accuracy increase on zero-shot commonsense reasoning tasks.

2 Dimension-wise Global Pruning via Merging Task-wise Importance Scores

In this section, we introduce DIET, a **Di**mension-wise structured pruning method for LLMs that merg**e**s **T**ask-wise importance signals to produce a single global pruning mask applicable across all tasks. The overview of DIET is presented in Fig. 1.

2.1 Preliminaries

Let $\mathcal{M}$ denote a transformer-based LLM with B transformer blocks. Without loss of generalization, we assume the following with respect to $\mathcal{M}$:

- Each transformer block has an identical structure which consists of a multi-head attention layer followed by an MLP layer, both with residual connections and layernorms (pre- or post-).
- It maintains a fixed hidden dimension d throughout all blocks, which serves as the input/output dimension for most linear sublayers: word embeddings, attention projections, MLP layers, and LM head. For example, Gemma-2 (Team et al., 2024) uses $d = 2304$ in 2B model and $d = 4608$ in 9B model across all B blocks.

Then, the goal of structured pruning is to remove entire computational units (*e.g.*, dimensions, attention heads, or blocks), while unstructured pruning zeroes individual weights of sublayers without such consideration. In this work, we mainly consider dimension-wise pruning, because the above architectural property enables a single global mask over d to be applied uniformly to all layers, without requiring per-layer or per-task customization.

Mathematical formulation. Let $y = Wx + b$ denote a generic linear sublayer in each block, where $W \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $x \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$. Then, depending on the type of sublayer, the dimensions m and n could be different:

- *Input* dimension $n = d$: word embeddings, attention query/key/value projections, MLP input projection, and LM head.
- *Output* dimension $m = d$: attention output projection and MLP output projection.

Then, for each case, we consider different operations for dimension-wise pruning. For layers with input dimension $n = d$, we apply *input-dimension masking* using a binary selector $\mathbf{z} \in \{0, 1\}^d$:

$$y \;=\; W P_{\mathbf{z}}\, x + b, \qquad P_{\mathbf{z}} = \mathrm{diag}(\mathbf{z}) \qquad (1)$$

which zeros the columns of W where $z_j = 0$.

For layers with output dimension $m = d$, we apply *output-dimension masking* using the same selector $\mathbf{z}$:

$$\tilde{y} \;=\; \tilde{P}_{\mathbf{z}}\,(Wx + b), \qquad \tilde{P}_{\mathbf{z}} = \mathrm{diag}(\mathbf{z}) \qquad (2)$$

which zeros the rows of W where $z_j = 0$. We refer to (1)–(2) collectively as the *generic masking operator*. The application of a masking operator reduces the effective dimension of $\mathcal{M}$ from d to $d' = \sum_{j=1}^{d} z_j$. Throughout this work, we apply the same global mask $\mathbf{z} \in \{0, 1\}^d$ to all parametrized sublayers (*i.e.*, except normalization) via the appropriate operator (input or output masking depending on the layer's input/output dimensions).

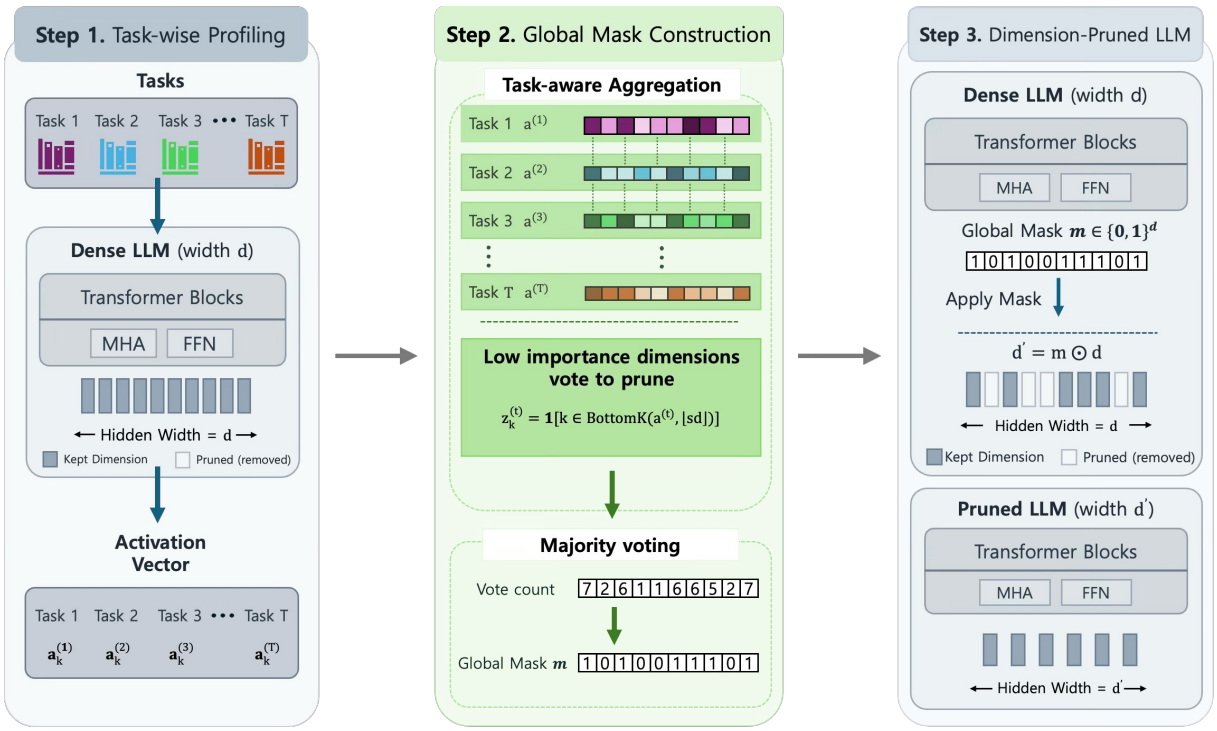

Figure 1: **Step 1.** Task-specific samples are used to profile a dense LLM and compute task-wise activation vectors. **Step 2.** Low-importance dimensions vote to prune, and the votes are merged by majority voting into a global mask $m \in \{0,1\}^d$. **Step 3.** The mask is applied across layers, producing a pruned LLM with reduced hidden width d'.

2.2 DIET: Merge Task-wise Importance for Global Dimension-wise Pruning of LLMs

DIET derives a global pruning mask through a two-phase process. First, DIET identifies the dimensions that contribute minimally to each task. Then, DIET aggregates these task-specific assessments: dimensions that most tasks agree to be uninformative are masked across all sublayers.

Task-wise importance scoring. For each task $t \in \mathcal{T}$, we assess dimension importance by measuring activation magnitudes using task-specific data. Activation magnitude serves as a widely-adopted proxy for feature contribution: dimensions with consistently low activations contribute little to the model's outputs and are candidates for removal (Sun et al., 2024; Molchanov et al., 2016).

Specifically, we profile the pretrained LLM f_θ on N samples $\{x_i^{(t)}\}_{i=1}^N$ from task t, recording activations at the output of the MLP layer in each transformer block (i.e., before the post-MLP layernorm and the residual add). Let $\mathcal{B}$ denote the set of transformer blocks (with $B = |\mathcal{B}|$). For sample $x_i^{(t)}$, let $h_{i,b}^{(t)} \in \mathbb{R}^{L_i \times d}$ represent the MLP outputs at block b, where L_i is the number of valid (non-padding) tokens and columns index residual dimensions. We compute a per-dimension importance score $a_k^{(t)}$ by averaging absolute activations first across all blocks (capturing cross-layer consistency) and then across all tokens and samples:

$$\bar{h}_i^{(t)}[j,k] \;=\; \frac{1}{B}\sum_{b\in\mathcal{B}} \big|\, h_{i,b}^{(t)}[j,k] \,\big|, \qquad (3)$$

$$a_k^{(t)} \;=\; \sum_{i=1}^{N}\sum_{j=1}^{L_i} \bar{h}_i^{(t)}[j,k] \;\Big/\; \sum_{i=1}^{N} L_i. \qquad (4)$$

The resulting vector $\mathbf{a}^{(t)} \in \mathbb{R}^d_{\geq 0}$ quantifies each dimension's contribution to task t. Given target sparsity $s \in [0,1]$, we identify the $\lfloor s \cdot d \rfloor$ dimensions with smallest $a_k^{(t)}$ as pruning candidates, producing a binary selector $\mathbf{z}^{(t)} \in \{0,1\}^d$ where $z_k^{(t)} = 1$ indicates a pruning vote for dimension k.

Merging into global pruning mask. Each task produces a pruning vote reflecting its own data distribution, but our objective is a single pruning decision applicable to all tasks. To this end, we employ a simple idea of *majority voting*: dimensions that multiple tasks independently flag as unimportant are likely safe to remove globally without significantly harming individual tasks.

Concretely, we aggregate votes by stacking task-wise selectors into a matrix $Z = [\mathbf{z}^{(t)}]_{t\in\mathcal{T}} \in \{0,1\}^{d\times T}$ and counting votes per dimension: $c_k = \sum_{t\in\mathcal{T}} z_k^{(t)}$. The indices for global pruning $\mathcal{O}$ consist of $\lfloor s \cdot d \rfloor$ dimensions with highest vote counts:

$$\mathcal{O} \;=\; \arg\operatorname{top}_{\lfloor s\cdot d\rfloor} \{\, c_k \,\}_{k=1}^{d}, \qquad (5)$$

where $\arg\operatorname{top}_k$ selects the indices of the k largest values. We define the omission mask $\mathbf{m} \in \{0,1\}^d$ by $m_k = \mathbb{1}\{k \in \mathcal{O}\}$, and let $\mathbf{u} = \mathbf{1} - \mathbf{m}$ denote the keep selector with $P = \operatorname{diag}(\mathbf{u})$ in Eq.1 and 2.

We apply this global mask to all parametrized sublayers uniformly: for every layer with input dimension d (embeddings, attention query/key/value projections, MLP input projections, LM head), we apply input-dimension masking via $W \leftarrow WP$ (Eq. 1), which zeros the columns of W corresponding to dimensions in $\mathcal{O}$. For every layer with output dimension d (attention output projection, MLP output projection), we apply output-dimension masking via $W \leftarrow \tilde{P}W$ (Eq. 2), which zeros the corresponding rows. This mask is applied identically across all B transformer blocks, yielding a single architectural modification: all sublayers now operate on the same pruned set of dimensions. This modification is task-agnostic, *i.e.*, the pruned model serves all tasks without per-task customization.

2.3 From Global Masking to Hard Pruning for Real-Time Efficiency

The global mask derived in Sec. 2.2 specifies which residual dimensions are removed across all sublayers. This mask is applied via the generic masking operators in Eq. 1–2, *i.e.*, zero masking, which is sufficient for evaluating the quality of dimension selection in terms of accuracy.

For efficiency evaluation, we additionally consider an *optional hard pruning* setting, where masked dimensions are physically removed and the hidden dimension is compacted from d to

$$d' = \sum_{j=1}^{d} u_j, \tag{6}$$

with $\mathbf{u} = \mathbf{1} - \mathbf{m}$ denoting the keep selector in Sec. 2.2. This compaction is applied uniformly across all residual-connected sublayers, resulting in a reduced hidden dimension. However, this hard pruning changes the input dimensionality of subsequent linear sublayers; under RMSNorm or LayerNorm, assuming independent and equally-scaled input features, reducing the input dimension from d to d' decreases the expected activation variance by a factor of d'/d. Without correction, this variance reduction can lead to degraded representations.

Variance correction via weight rescaling. To compensate for this effect, we rescale the weights of linear sublayers whose *input dimension* is reduced. Specifically, we apply the scaling factor

$$\alpha = \sqrt{\frac{d}{d'}}. \tag{7}$$

For a linear sublayer whose input dimension is sliced to d', the corresponding weight matrix is rescaled as

$$W_{\text{scaled}} = \alpha \cdot W_{\text{sliced}}, \tag{8}$$

where W_{sliced} denotes the weight matrix after removing masked input dimensions. For sublayers where only the output dimension is reduced (*e.g.*, attention output projection or MLP down-projection), the input dimensions remain unchanged and no rescaling is applied. Normalization parameters are sliced to the retained dimensions without scaling.

Unless stated otherwise, all accuracy-focused results are obtained using zero masking, while hard pruning with variance correction is applied only for real-time efficiency evaluations.

3 Experiments

3.1 Setups

Datasets. We evaluate DIET on seven widely-used NLP benchmarks covering diverse reasoning and language understanding capabilities: (1) *BoolQ* (Clark et al., 2019) contains yes/no questions derived from naturally occurring queries paired with Wikipedia passages. (2) *RTE* (Dagan et al., 2005) (Recognizing Textual Entailment) evaluates whether a hypothesis is entailed by a given premise. (3) *HellaSwag* (Zellers et al., 2019) tests commonsense reasoning through sentence completion requiring contextual understanding. (4) *WinoGrande* (Sakaguchi et al., 2021) measures commonsense reasoning via pronoun disambiguation. (5, 6) *ARC-Challenge and ARC-Easy* (Clark et al., 2018) provide grade-school science questions at varying difficulty levels. (7) *OpenBookQA* (Mihaylov et al., 2018) requires multi-hop reasoning over elementary science facts.

For task-wise importance scoring (Sec. 2.2), we randomly sample 100 examples per task from the training split of each dataset. Final evaluations use the native splits defined by the lm-evaluation-harness (Sutawika et al., 2024): BoolQ, RTE, HellaSwag, and WinoGrande are evaluated on their validation splits, while ARC-Easy, ARC-Challenge, and OpenBookQA use test splits.

Table 1: Zero-shot accuracy comparisons of DIET against baseline pruning algorithms on Gemma-2 2B on seven NLP benchmarks. The best performances are marked in **bold**, and the runner-up is marked with underline. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
No pruning (Original)	0	73.5	61.4	73.0	68.4	80.2	49.7	41.8	64.0
Magnitude-Dim	20%	38.2	55.2	31.1	50.0	39.8	21.3	25.0	37.2
SliceGPT	20%	37.8	53.4	25.3	50.4	26.2	24.9	27.0	35.0
PuDDing	20%	43.4	53.8	25.8	49.8	27.3	23.6	27.0	35.8
DIET (Ours)	20%	63.3	54.9	41.8	53.2	49.8	24.1	27.6	**45.0**
Magnitude-Dim	10%	51.3	52.7	44.8	53.9	59.6	32.5	31.2	46.6
SliceGPT	10%	37.8	53.4	25.2	50.2	26.3	25.0	27.0	35.0
PuDDing	10%	42.6	51.3	25.8	50.8	27.3	23.7	30.0	35.9
DIET (Ours)	10%	70.6	54.9	62.7	62.8	69.0	38.4	35.4	**56.3**

Metrics. We evaluate with two metrics: standard accuracy (*acc*) and length-normalized accuracy (*acc_norm*). For multiple-choice datasets with variable-length options—HellaSwag, ARC-Easy, ARC-Challenge, and OpenBookQA—we report *acc_norm* as the primary metric because it mitigates length bias by dividing each option's log-likelihood by its token length. For BoolQ, RTE, and WinoGrande, we follow the default `lm-eval-harness` task definitions and report *acc* only, since answers are extremely short (often single-token verbalizers), making length normalization unhelpful and potentially noisy. We present per-task scores and an unweighted average across all seven tasks.

Baselines. We compare DIET with three structured pruning baselines under matched sparsity ratios. (1) MAGNITUDE–DIM (Han et al., 2015): dimension-wise pruning that aggregates absolute-weight scores per residual dimension and applies a single global dimension mask uniformly to all residual-connected linear layers, zeroing corresponding rows or columns. (2) SLICEGPT (Ashkboos et al., 2024): rotate-and-slice compression that learns an orthogonal activation basis from a calibration corpus and slices MLP and attention projections to a reduced hidden width implied by the target sparsity. (3) PUDDING (Wee et al., 2025): prompt-conditioned depth pruning in which trained lightweight router selects, per input, which transformer blocks to skip. By omitting layers selected by trained router, it achieves target sparsity ratio.

Implementation details. In our experiments, we employ `Gemma-2` models (Team et al., 2024) with different sizes of 2B and 9B. For the `Task-wise importance scoring` phase, we randomly sampled 100 samples from each task to accelerate the activation profiling process (the effect of sample size on zero shot performance is analyzed in Figure 3). To ensure reproducibility and mitigate the influence of random sampling, we set the random seed to 42 throughout all experiments. For baseline methods, we follow the default configurations from their released implementations. SliceGPT (Ashkboos et al., 2024) uses WikiText2 (Merity et al., 2016) as the calibration corpus with 64 samples and a maximum sequence length of 2048. To ensure fair comparison with DIET, we also evaluate SliceGPT using 100 samples per task from the seven zero-shot benchmarks described in Section 3.1. PuDDing (Wee et al., 2025) trains a BERT-base (Devlin et al., 2019) router for 10 epochs using AdamW (Loshchilov and Hutter, 2017) optimizer with an 80/20 train/test split. We trained router on the same benchmark suite, mentioned in 3.1. Magnitude-based methods apply global L1 pruning, with the structured variant following (1)–(2) to mask dimensions uniformly across residual-connected layers. All methods are evaluated at 10%, 20%, and 30% sparsity ratios and all evaluations use lm-evaluation-harness (Sutawika et al., 2024) in zero-shot mode with fixed seed 42.

3.2 Main results

Table 1 presents zero-shot accuracy on seven benchmarks for Gemma-2 2B (Team et al., 2024), comparing DIET against structured pruning baselines SliceGPT (Ashkboos et al., 2024) and PuDDing (Wee et al., 2025), as well as magnitude-based methods (Han et al., 2015). Here, DIET consistently outperforms both published structured

Table 2: Zero-shot accuracy comparisons of DIET against baseline pruning algorithms on Gemma-2 9B on seven NLP benchmarks. The best performances are marked in **bold**, and the runner-up is marked with underline. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
No pruning (Original)	0	84.4	67.9	80.0	74.2	87.8	66.1	47.2	72.5
Magnitude-Dim	20%	68.5	53.8	49.5	60.6	62.0	30.8	31.6	51.0
SliceGPT	20%	37.8	52.7	25.8	50.5	27.4	25.0	27.6	35.3
PuDDing	20%	62.8	57.4	49.8	59.7	51.3	32.1	29.2	48.9
DIET (Ours)	20%	74.7	54.2	48.5	59.2	61.7	32.0	31.2	**51.6**
Magnitude-Dim	10%	82.7	62.1	71.1	71.4	80.2	50.6	43.2	**65.9**
SliceGPT	10%	37.8	52.7	25.7	50.5	27.2	25.0	27.4	35.2
PuDDing	10%	64.5	64.3	49.5	59.5	49.1	30.3	30.2	49.7
DIET (Ours)	10%	82.8	62.1	69.9	68.7	79.1	49.6	40.0	64.6

Table 3: Real-time efficiency comparison between original model and DIET-pruned model on Gemma-2 2B. Latency is measured in milliseconds (ms), memory denotes peak GPU memory usage in megabytes (MB), and FLOPs are computed per forward pass (10^{12}).

Task	Latency (ms)		Memory (MB)		FLOPs (10^{12})	
	Base	DIET	Base	DIET	Base	DIET
BoolQ	62.98	40.55	5466	4469	8.05	6.44
RTE	37.64	39.95	5263	4267	4.27	3.42
HellaSwag	38.16	39.28	5174	4179	4.23	3.39
WinoGrande	37.47	38.75	5094	4098	1.90	1.52
Arc-Easy	37.93	39.55	5127	4131	2.90	2.32
Arc-Challenge	38.47	39.76	5208	4212	3.83	3.06
OpenBookQA	38.02	39.38	5099	4103	2.44	1.95
Avg	41.52	**39.58**	5204	**4209**	3.95	**3.16**

pruning methods across all sparsity levels. For example, at 10% sparsity, DIET achieves 56.3% average accuracy, exceeding SliceGPT by 21.3 % and PuDDing by 20.4 %. This gap persists at 20% sparsity, where DIET (45.0%) surpasses SliceGPT by 10.0% and PuDDing (35.8%) by 9.2%. Notably, while SliceGPT and PuDDing exhibit severe performance degradation—dropping to near-random performance on several tasks—DIET preserves substantially more task-specific capabilities. Per-task analysis further reveals this advantage holds broadly: at 10% sparsity, DIET outperforms both structured baselines on all seven tasks, with particularly strong results on BoolQ (70.6%), HellaSwag (62.7%), and WinoGrande (62.8%). At 20%, DIET exceeds SliceGPT on six tasks and PuDDing on all seven tasks, demonstrating particular robustness on reasoning tasks like BoolQ and WinoGrande.

Next, we additionally conduct the experiments with larger model. Table 2 presents results on Gemma-2 9B model at 10% and 20% sparsity levels. Similar to the results on 2B model, DIET maintains its strong advantage over structured pruning baselines across model scales. For instance, at 10% sparsity, DIET achieves 64.6% average accuracy, substantially outperforming SliceGPT by 29.4% and PuDDing by 14.9%. At 20% sparsity, DIET (51.6%) continuously surpasses SliceGPT (35.3%) and PuDDing (48.9%).

The comparison across model scales reveals consistent patterns. DIET's advantage over SliceGPT persists on both 2B and 9B models, with gaps exceeding 16% at 20% sparsity. Against PuDDing, DIET demonstrates stronger improvements on the smaller 2B model (9.2%) compared to 9B (2.7%), suggesting that task-aware dimension selection becomes increasingly competitive as model capacity grows. While Magnitude-Dim achieves 51.0% on 9B at 20% sparsity—slightly below DIET's 51.6%—DIET still demonstrates competitive performance among dimension-level structured approaches. This consistent advantage across two model scales spanning over 4× in parameter count positions DIET as a robust structured pruning method across different architectures.

Real-time efficiency. We report real-time efficiency results in terms of latency, peak memory usage, and FLOPs in Table 3. All efficiency measurements are conducted on the Gemma-2 2B at a sparsity ratio of 20%, using the hard-pruned implementation described in Sec. 2.3. Overall, DIET achieves consistent reductions in memory footprint and FLOPs across all evaluated tasks, leading to an improvement in average latency from 41.52 ms to

Table 4: Zero-shot accuracy comparisons of the standard zero-masking DIET and the hard-pruning implementation (DIET-Hard) on Gemma-2 2B. DIET-Hard physically removes dimensions and applies variance rescaling to mitigate distribution shift. The better performance between the two pruned methods is marked in **bold**. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
No pruning (Original)	0	73.5	61.4	73.0	68.4	80.2	49.7	41.8	64.0
DIET (Zero-mask)	20%	63.3	54.9	41.8	53.2	49.8	24.1	27.6	45.0
DIET (Hard-prune)	20%	64.7	51.6	42.5	54.7	51.3	25.9	27.4	**45.4**

39.58 ms. While per-task latency may vary due to runtime effects, reductions in memory usage and computational cost are consistent across tasks, indicating practical efficiency gains of hard pruning.

In addition to efficiency improvements, we verify that hard pruning preserves predictive performance. Table 4 compares the zero-shot accuracy of the standard zero-masking implementation of DIET (reported in Table 1) with its hard-pruned counterpart under the same sparsity ratio. Despite physically removing dimensions and compacting the hidden size, DIET-Hard maintains accuracy comparable to the zero-masked model across tasks. In particular, the average accuracy of DIET-Hard (45.4%) closely matches that of the zero-masked DIET (45.0%), demonstrating that the proposed variance correction effectively mitigates the distribution shift induced by dimensionality reduction. These results indicate that hard pruning serves as a faithful and efficient realization of DIET, enabling real-time efficiency gains without sacrificing zero-shot performance.

3.3 More Analyses

In this section, we present additional analyses of DIET on the seven zero-shot benchmarks using Gemma-2 2B (Team et al., 2024) unless otherwise noted. Due to space constraints, we report the main findings here and defer the remaining experiments to Appendix B. These include ablation studies on two mask-construction choices: the use of a merged global mask instead of single-task masks, and binary vote-based merging instead of continuous score aggregation. Together, these results validate that aggregating task-wise pruning signals into a single global mask is a robust design choice.

Cross-task agreement via voting. We evaluate cross-task consensus on uninformative residual dimensions. For each dimension k, we compute total pruning votes $c_k = \sum_t z_k^{(t)}$ across the seven benchmarks (Figure 2). The histogram reveals striking consensus: dimensions flagged by nearly all tasks (7 or 6 votes) account for ≈10% of the hidden width d, naturally justifying a 10% sparsity target. Expanding this threshold to ≥ 4 votes encompasses ≈20% of dimensions, matching our target sparsity with strong agreement. This heavy vote concentration demonstrates that DIET uncovers universal architectural redundancies across diverse tasks, validating our core assumption that multi-task consensus robustly guides training-free pruning.

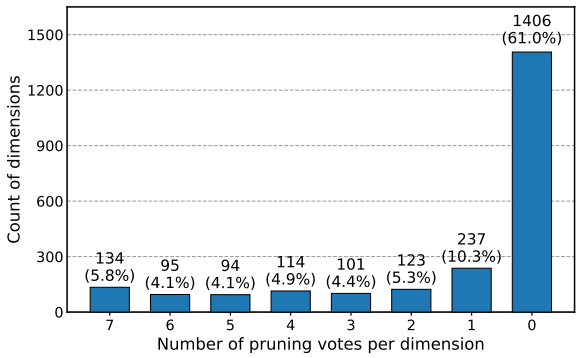

Figure 2: Consensus-vote histogram by dimension. Numbers show total counts (and share of d=2304).

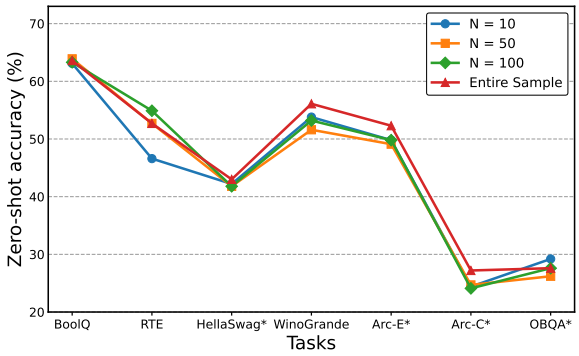

Figure 3: Zero-shot accuracy by activation-profiling sample size. Benchmarks with * use *acc_norm*.

Robustness to profiling sample size. We investigate the activation-profiling budget by varying samples per task $N \in \{10, 50, 100, \text{full}\}$ under

Table 5: Zero-shot task average accuracy comparison on Qwen2.5-7B and Phi-4-mini-reasoning, applying DIET. We have applied 10% and 20%.

Method	Sparsity	Qwen2.5-7B	Phi-4-mini
No pruning	0%	70.5	58.8
Magnitude - Dim	10%	38.6	36.1
DIET (Ours)	10%	60.9	46.3
DIET (Ours)	20%	37.0	37.2

20% sparsity. As shown in Figure 3, zero-shot accuracy improves consistently with N, rising from 44.2% ($N = 10$) to 46.1% (full split). Crucially, the performance gap between a modest budget of $N = 100$ (45.0%) and the full baseline is only 1.1%. This marginal difference indicates that DIET reaches near-saturation with minimal calibration data, effectively balancing efficiency and downstream performance.

Generalization of DIET to various LLMs. To verify cross-architecture generalizability, we evaluate DIET on Qwen2.5-7B (Qwen et al., 2025) and Phi-4-Mini-Reasoning (3.8B) (Xu et al., 2025) (Table 5). At 10% sparsity, DIET successfully preserves most original performance across both backbones without retraining. However, at 20% sparsity, performance drops more noticeably (≈33% on Qwen2.5-7B and 21% on Phi-4-Mini). This suggests that while activation-driven aggregation transfers robustly under light compression, aggressive pruning amplifies architecture-specific sensitivities, highlighting the need for adaptive layer-wise scaling mechanisms in future work.

4 Related Works

Structured pruning of LLMs. Structured pruning methods are commonly categorized by pruning granularity into dimension-wise and block-wise approaches (Ashkboos et al., 2024; Song et al., 2024). Dimension-wise methods reduce layer width by removing less informative channels or embeddings; SliceGPT (Ashkboos et al., 2024) uses PCA-based alignment, while LLM-Pruner (Ma et al., 2023) and LoRAShear (Chen et al., 2023) rely on gradient-based estimation or progressive structural optimization with recovery fine-tuning. Block-wise methods instead reduce depth by removing redundant Transformer blocks using perplexity drops or iterative selection metrics (Song et al., 2024; Kim et al., 2024).

Task-aware pruning. Recent works explore dynamic, task-aware pruning to adapt compression to downstream tasks or inputs. At the block level, PuDDing (Wee et al., 2025) trains a lightweight router to predict per-prompt block omission sets. At the dimension level, Instruction-Following Pruning (IFPruning) (Hou et al., 2025) co-trains a dynamic sparsity predictor with the backbone LLM to generate input-dependent dimension masks. Although these methods achieve high task adaptability, they require extensive training and introduce non-trivial runtime routing overhead.

Training-free pruning. Training-free approaches avoid the costs of router training, continued pre-training, or post-pruning recovery (Ma et al., 2023; Chen et al., 2023; Wee et al., 2025; Hou et al., 2025), making them practical for one-shot deployment. Existing methods such as SLEB (Song et al., 2024), Shortened-LLaMA (Kim et al., 2024), and SliceGPT (Ashkboos et al., 2024) compute fixed masks from general calibration corpora, making them task-agnostic and suboptimal for downstream demands. In contrast, DIET is both task-aware and training-free. By profiling activation magnitudes on a small budget of task-specific samples and aggregating them via majority voting, DIET yields a single global mask that captures multi-task signals without parameter updates or runtime routing.

5 Conclusion

We presented DIET, a training-free dimension-wise global pruning framework that merges task-wise importance estimates into a single global mask. By profiling MLP activations on small per-task samples and aggregating through majority voting, DIET systematically removes residual dimensions that multiple tasks deem uninformative. Across seven zero-shot benchmarks on Gemma-2 2B and 9B, DIET consistently outperforms previous state-of-the-art structured pruning baselines at matched sparsity levels, achieving improvements of up to 21% at 10% sparsity ratio. In addition, DIET achieves consistent reductions in FLOPs and peak memory under hard pruning, leading to measurable latency improvements without degrading zero-shot performance. These results indicate that a simple, router-free, activation-driven approach can deliver robust task-aware compression with modest profiling costs, positioning DIET as a practical choice for structured LLM pruning.

Limitation and Future Work

Our evaluation focuses on seven English-centric zero-shot benchmarks; broader coverage across multilingual domains, safety-critical tasks, long-context scenarios, and tool-use applications remains unexplored. At high sparsity levels on non-Gemma architectures, we observe notable accuracy degradation, suggesting the need for adaptive sparsity schedules, per-layer allocation strategies, or lightweight post-pruning recovery techniques. Also, task-weighted or distribution-aware voting mechanisms may improve robustness, while smarter calibration policies, such as difficulty-aware or uncertainty-driven sampling, could enhance sample efficiency.

Broader Impact and Ethical Implications

By reducing computational and memory requirements through a single global mask, DIET offers practical benefits for model deployment and environmental sustainability. Unlike task-specific pruning methods that require maintaining multiple pruned variants, DIET's global mask produces a single task-aware model capable of handling diverse tasks effectively. This consolidation simplifies deployment pipelines, reduces storage overhead, and lowers both energy consumption during inference and the carbon footprint associated with model distribution. By eliminating the need for multiple model variants, DIET makes large language models more accessible to resource-constrained settings while maintaining task-aware capabilities. However, pruning alters internal representations and may shift model behavior in fairness, safety, and robustness. Practitioners should audit pruned models for bias and distribution shift, rerun safety evaluations, and transparently document sparsity levels and calibration procedures. We recommend using publicly available calibration data to respect privacy and pairing deployments with appropriate monitoring and safeguards.

References

Saleh Ashkboos, Maximilian Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. Slicegpt: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations*, volume 2024, pages 11682–11701.

Tianyi Chen, Tianyu Ding, Badal Yadav, Ilya Zharkov, and Luming Liang. 2023. Lorashear: Efficient large language model structured pruning and knowledge recovery. *arXiv preprint arXiv:2310.18356*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2924–2936.

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

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2475–2485.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. volume 28.

Bairu Hou, Qibin Chen, Jianyu Wang, Guoli Yin, Chong Wang, Nan Du, Ruoming Pang, Shiyu Chang, and Tao Lei. 2025. Instruction-following pruning for large language models.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. 2024. Shortened llama: Depth pruning for large language models with comparison of retraining methods. *arXiv preprint arXiv:2402.02834*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. volume 36, pages 21702–21720.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2381–2391.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning convolutional neural networks for resource efficient inference.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. volume 64, pages 99–106. ACM New York, NY, USA.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. 2024. Sleb: Streamlining llms through redundancy verification and elimination of transformer blocks.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, volume 2024, pages 4942–4964.

Lintang Sutawika, Hailey Schoelkopf, Leo Gao, Baber Abbasi, Stella Biderman, Jonathan Tow, ben fattori, Charles Lovering, farzanehnakhaee70, Jason Phang, Anish Thite, Fazz, Thomas Wang, Niklas Muennighoff, Aflah, sdtblck, nopperl, gakada, tttyuntian, and 11 others. 2024. Eleutherai/lm-evaluation-harness: v0.4.2.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Juyun Wee, Minjae Park, and Jaeho Lee. 2025. Prompt-based depth pruning of large language models.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, and 1 others. 2025. Phi-4-mini-reasoning: Exploring the limits of small reasoning language models in math. *arXiv preprint arXiv:2504.21233*.

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.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

A Experimental Details

This section provides comprehensive details on our experimental setup, including dataset descriptions, prompt/evaluation policies, implementation specifics, and reproducibility aspects.

A.1 Datasets

We evaluate on seven English benchmarks that are standard in prior pruning/efficiency work: BoolQ, RTE, HellaSwag, WinoGrande, ARC-Easy, ARC-Challenge, and OpenBookQA. Additionally, to verify the structural robustness and generalization of our method across multilingual and instruction-following domains, we extend our evaluation to XNLI and IFEval benchmarks in the additional analyses (Appendix B). The comprehensive specifications for our evaluation pipeline, including the prompt wrapping policies and internal activation token lengths, are detailed in Table 15 and Table 16, respectively. Activation statistics for pruning are collected on the train split of each dataset, while final accuracy is reported with lm-eval-harness on its default split.

BoolQ BoolQ (Clark et al., 2019) is a yes/no reading comprehension dataset built from naturally occurring questions paired with supporting passages, primarily sourced from the web (e.g., Wikipedia). Each example contains a short question and a paragraph that may contain sufficient evidence to answer it. In our pipeline, we use the **train** split to collect activation statistics and the lm-eval-harness default configuration for zero-shot evaluation. During activation collection, we do not employ an explicit prompt; instead, inputs are formed by concatenating the passage and question. For evaluation, we rely on lm-eval's standard yes/no template and report accuracy (and acc_norm where applicable). Tokenization uses `max_length=512` with padding/truncation; padding tokens are masked in our forward-hook aggregation.

RTE RTE (Dagan et al., 2005) evaluates two-way natural language inference over (premise, hypothesis) pairs. Labels are *entailment* vs. *not entailment*. We collect activations on the **train** split by concatenating the premise and hypothesis without a handcrafted prompt, then evaluate zero-shot with lm-eval-harness' default NLI formatting and report accuracy. As with other tasks, we tokenize with `max_length=512` and mask padding during aggregation to avoid biasing per-dimension importance estimates.

Component	Content
Question	do iran and afghanistan speak the same language
Context	Persian language – Persian, also known by its endonym Farsi, is one of the Western Iranian languages... It is primarily spoken in Iran, Afghanistan (officially known as Dari since 1958), and Tajikistan...
Ground Truth	Yes

Table 6: Example data from BOOLQ.

Component	Content
Premise	No Weapons of Mass Destruction Found in Iraq Yet.
Hypothesis	Weapons of Mass Destruction Found in Iraq.
Ground Truth	not entailment

Table 7: Example data from RTE.

HellaSwag HellaSwag (Zellers et al., 2019) is a commonsense sentence completion benchmark where a short context must be continued with the most plausible of four candidate endings. It targets subtle lexical and pragmatic cues and is known to expose annotation artifacts if option order or formatting drifts from the original. In our setup, activation statistics are computed on the **train** split by concatenating the context with all four endings (no explicit instruction). Final accuracy is measured with lm-eval-harness zero-shot defaults that present the context and choices and parse answers as A–D. We keep the original option order and preserve special characters; long inputs are truncated at `512` tokens.

WinoGrande WinoGrande (Sakaguchi et al., 2021) is a large-scale pronoun/coreference commonsense benchmark cast as a cloze task: a sentence with a single blank must be filled with one of two candidate nouns. We gather activations on the **train** split using only the raw sentence (without a formatted instruction or candidate list) to reflect content-driven activations; zero-shot evaluation then follows lm-eval's default cloze template that shows the candidates and expects an answer in {1,2}. We report accuracy and escape underscores in LaTeX where necessary; truncation follows the same `512`-token policy.

Component	Content
Context	Then, the man writes over the snow covering the window of a car, and a woman wearing winter clothes smiles. then
Options	A. , the man adds wax to the windshield and cuts it. B. , a person board a ski lift, while two men supporting the head of the person wearing winter clothes snow as the we girls sled. C. , the man puts on a christmas coat, knitted with netting. D. , the man continues removing the snow on his car.
Ground Truth	D

Table 8: Example data from HELLASWAG.

Component	Content
Sentence	Ian volunteered to eat Dennis's menudo after already having a bowl because ___ despised eating intestine.
Options	1. Ian 2. Dennis
Ground Truth	2 (*Dennis*)

Table 9: Example data from WINOGRANDE.

ARC-Easy ARC-Easy (Clark et al., 2018) contains elementary-level science multiple-choice questions with four options. Items often involve factual recall plus light reasoning and unit/term fidelity. For activation collection, we use the **train** split and build inputs by concatenating the stem with the four option texts (no explicit prompt). Final zero-shot evaluation uses lm-eval-harness' standard MCQ formatting, preserving the original option order, and we report accuracy (and acc_norm where available). Scientific notation and symbols are kept verbatim; overly long stems are truncated after tokenization.

Component	Content
Question	Which factor will most likely cause a person to develop a fever?
Options	A. a leg muscle relaxing after exercise B. a bacterial population in the bloodstream C. several viral particles on the skin D. carbohydrates being digested in the stomach
Ground Truth	B

Table 10: Example data from ARC-EASY.

ARC-Challenge ARC-Challenge (Clark et al., 2018) is the harder subset of ARC that requires multi-hop reasoning or background knowledge beyond straightforward recall. We follow the same protocol as ARC-Easy: activations are collected on the **train** split using concatenated stem+choices without an explicit instruction, and evaluation is performed zero-shot with lm-eval defaults and original option order. As is typical, accuracy on ARC-Challenge is substantially lower than ARC-Easy for the same base model; our pruning analysis therefore reports both to illustrate difficulty sensitivity.

Component	Content
Question	George wants to warm his hands quickly by rubbing them. Which skin surface will produce the most heat?
Options	A. dry palms B. wet palms C. palms covered with oil D. palms covered with lotion
Ground Truth	A

Table 11: Example data from ARC-CHALLENGE.

OpenBookQA OpenBookQA (Mihaylov et al., 2018) is a four-choice science QA dataset designed around a small "open book" of core facts. Many items benefit from combining the provided stem with general science knowledge. In our experiments we do *not* augment with retrieval; activations on the **train** split are computed by concatenating the question stem and choice texts (no explicit prompt), and final zero-shot accuracy is obtained with lm-eval's default MCQ template. This isolates pruning effects from external knowledge pipelines and keeps evaluation comparable to prior efficiency work.

Component	Content
Question	The sun is responsible for
Options	A. puppies learning new tricks B. children growing up and getting old C. flowers wilting in a vase D. plants sprouting, blooming and wilting
Ground Truth	D

Table 12: Example data from OPENBOOKQA.

XNLI XNLI (Conneau et al., 2018) is a multilingual natural language inference benchmark constructed by extending the English MNLI dataset

to multiple languages via human translation. Each example consists of a premise–hypothesis pair labeled as entailment, contradiction, or neutral, and is commonly used to evaluate cross-lingual sentence representations. In our experiments, we evaluate zero-shot performance without any language-specific adaptation or fine-tuning. Model predictions are obtained using lm-eval's default NLI prompt template, and accuracy is reported by averaging results across languages. This setting allows us to assess how well pruning preserves distributed multilingual knowledge under a unified evaluation protocol.

Component	Content
Premise	Yes, I sang for him and beautiful spirituals he liked.
Hypothesis	I refused to sing for him.
Options	A. Entailment B. Contradiction C. Neutral
Ground Truth	B

Table 13: Example data from XNLI.

IFEval IFEval (Zhou et al., 2023) is a benchmark designed to evaluate instruction-following capability of large language models through verifiable constraints. Each instance specifies explicit requirements on the generated output, such as format, length, or content inclusion, which are automatically checked using rule-based evaluators rather than reference matching. In our evaluation, we follow the standard zero-shot setup without additional prompting or alignment techniques. Model outputs are assessed using the strict prompt-level accuracy metric provided by the benchmark. This evaluation focuses on whether pruning preserves the model's ability to reliably follow instructions, independent of semantic correctness or stylistic quality.

A.2 Measurement details

We report **accuracy** (and **acc_norm** where available) from `lm-eval-harness` with zero-shot (`-num_fewshot 0`). All results use the official task loaders and default post-processing of lm-eval to ensure comparability.

A.3 Baselines

As our main baseline, we evaluate the unpruned base model with the same lm-eval settings. No custom prompt is added (zero-shot, defaults). For completeness, the basic evaluation templates (rendered by lm-eval) are summarized in Table 17.

Component	Content
Prompt	Write a short essay about climate change. Your response must have exactly 3 paragraphs. Do not use the word "global" anywhere in the text.
Constraints	1. Number of paragraphs must be exactly 3. 2. Negative constraint: Forbidden word "global".
Ground Truth	Verified automatically via rule-based software program execution.

Table 14: Example data from IFEVAL.

A.4 Implementation details

We use Hugging Face `AutoModelForCausalLM`/`AutoTokenizer` with local checkpoints. For activation collection, we construct inputs from dataset fields, attach forward hooks to each MLP down-projection, and compute mean absolute activations over non-padded tokens (via the attention mask) to score hidden dimensions. Per task, dimensions are ranked and the lowest (r%) are selected; a simple vote across tasks yields a consensus set, which we prune by zero-masking channels in linear layers whose input/output size matches the hidden width. Final accuracy is reported with `lm_eval` (hf backend, zero-shot). All runs use a fixed seed (42) for reproducibility.

Stage	Basic Prompt / Source
Activation Collection (Pruning)	No explicit prompt. Inputs are formed by concatenating dataset fields (e.g., `premise + hypothesis`, `question + choices`, `ctx + endings`).
Final Evaluation (lm-eval)	**lm-eval-harness default prompts** with zero-shot (`-num_fewshot 0`). No custom instruction added.

Table 15: Prompt policy used in our pipeline. For pruning analysis we do not use explicit templates; for final evaluation we rely on lm-eval-harness defaults.

Dataset	Split	Input Format for Activation Collection
BoolQ	train	`{passage} Question: {question} Answer:`
RTE	train	`{premise} Question: {hypothesis} True or False?`
HellaSwag	train	`{context} {choice}`
WinoGrande	train	`{sentence}`
ARC-Easy	train	`Question: {question} Answer: {choice}`
ARC-Challenge	train	`Question: {question} Answer: {choice}`
OpenBookQA	train	`Question: {question} Answer: {choice}`
XNLI	train	`{premise} Question: {hypothesis} True, False, or Neither?`
IFEval	train	`{prompt}`

Table 16: Input formatting templates used for aggregating activation statistics across different benchmarks.

Benchmark	Basic Prompt
BoolQ	Answer the question with **Yes or No** based on the context. Q: {question} Context: {passage} Final format: 'Answer: [Yes or No]'
RTE	Determine whether the hypothesis is entailed by the premise. Premise: {premise} Hypothesis: {hypothesis} Final format: 'Answer: [entailment or not entailment]'
HellaSwag	Choose the most plausible ending. Context: {ctx} Choices: (A) {A} (B) {B} (C) {C} (D) {D} Final format: 'Answer: [A/B/C/D]'
WinoGrande	Fill the blank in the sentence with one candidate. Sentence: {sentence with ___ blank} Candidates: {option1}, {option2} Final format: 'Answer: [1 or 2]'
ARC-Easy / ARC-Challenge	Select one correct option. Question: {question} Choices: (A) {A} (B) {B} (C) {C} (D) {D} Final format: 'Answer: [A/B/C/D]'
OpenBookQA	Use basic science knowledge to answer. Question: {question_stem} Choices: (A) {A} (B) {B} (C) {C} (D) {D} Final format: 'Answer: [A/B/C/D]'
XNLI	Determine whether the hypothesis is entailed by the premise. Premise: {premise} Hypothesis: {hypothesis} Final format: 'Answer: [entailment, contradiction, or neutral]'
IFEval	{prompt} Final format: Generated free-form text output subject to strict rule-based constraint evaluation.

Table 17: Basic prompts used for our **evaluation** (lm-eval-harness defaults).

B Additional Analyses

SliceGPT calibration robustness. We examine the impact of different calibration strategies for SliceGPT, comparing the original approach (64 samples from WikiText2 with a maximum sequence length of 2048) against an alternative using 100 randomly selected samples from our seven zero-shot benchmarks. As shown in Table 18, the performance discrepancy between these two calibration data sources remains minimal (less than 1.0% across both 10% and 20% sparsity levels), indicating that SliceGPT is relatively robust to the choice of calibration distribution.

Performance under aggressive sparsification. To explore the limits of compression, we extend our evaluation to an aggressive 30% sparsity regime, with results summarized in Table 19 and Table 20. While the performance advantages of DIET persist over other pruning baselines, all evaluated methods suffer substantial accuracy degradation in this high-sparsity environment. This shared trend indicates that the drop stems from the inherent difficulty of extreme model sparsification rather than a structural flaw in DIET.

Robustness across diverse capabilities. To evaluate how well DIET preserves broader linguistic and functional dimensions, we report additional zero-shot results on XNLI (multilingual understanding) and IFEval (verifiable instruction-following) in Tables 21, 22, and 23. At 10% sparsity, DIET consistently outperforms weight-magnitude baselines on XNLI for both Gemma-2-2B and 9B backbones, showing a more pronounced margin in the smaller model scale. On IFEval, DIET demonstrates highly competitive or improved performance, particularly on the 9B model, suggesting that global dimension-wise pruning effectively maintains instruction-following boundaries as model capacity scales.

Global mask vs. per-task masks. We compare our merged global mask to models pruned with a single-task mask derived from *arc_easy* and *arc_challenge*. As shown in Table 24, the global mask improves average zero-shot accuracy over single-task mask applied models. The gains are largest on *RTE*, as the global mask model (54.9%) outperforms model with *arc_easy* mask with 2.4% and model with *arc_challenge* mask with 2.5%.

Continuous score merging vs. binary voting. Beyond binary selectors, we also merge *continuous* task-wise importance scores: for each dimension, we construct a vector of normalized activation magnitudes across tasks, aggregate it to obtain a scalar score, and rank dimensions accordingly (smaller scores indicate less informative dimensions). We then select a global fraction of dimensions according to the target sparsity (20%) based on this ranking and prune the model. As reported in Table 25, the pruned model under this continuous merging scheme attains accuracy comparable to the binary vote–based DIET, showing that activation magnitudes provide sufficiently informative signals and that DIET is robust to the merging granularity (binary vs. continuous).

C Usage of AI Tools

During manuscript preparation, we used AI-based language tools for editorial revisions such as grammar correction and readability improvements. These tools did not contribute to the conception of the study, methodological development, experimental design, data analysis, or the articulation of scientific conclusions. All substantive content originates from the authors; AI assistance was restricted to copy-editing and has no bearing on the work's originality.

Table 18: Zero-shot accuracy of *SliceGPT* on Gemma-2 2B on seven NLP benchmarks, using WikiText2 and seven benchmarks. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
No pruning (Original)	0	73.5	61.4	73.0	68.4	80.2	49.7	41.8	64.0
SliceGPT - Zero-shot Benchmark	20%	37.8	53.4	25.3	50.4	26.2	24.9	27.0	35.0
SliceGPT - WikiText2	20%	37.8	52.7	24.5	50.0	26.1	26.4	25.8	34.8
SliceGPT -Zero-shot Benchmark	10%	37.8	53.4	25.2	50.2	26.3	25.0	27.0	35.0
SliceGPT - WikiText2	10%	37.9	47.3	25.2	50.4	25.8	25.3	27.4	34.2

Table 19: Zero-shot accuracy comparisons of DIET against baseline pruning algorithms on Gemma-2 2B on seven NLP benchmarks. The best performances are marked in **bold**, and the runner-up is marked with underline. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
No pruning (Original)	0	73.5	61.4	73.0	68.4	80.2	49.7	41.8	64.0
Magnitude - Dim	30%	37.8	50.5	28.7	49.3	34.5	20.7	25.6	35.3
SliceGPT	30%	37.8	50.2	25.2	50.7	25.7	24.5	27.4	34.5
PuDDing	30%	39.6	46.2	26.3	49.6	28.6	24.5	27.2	34.6
DIET (Ours)	30%	41.3	49.1	28.8	49.3	35.9	21.5	24.2	**35.7**

Table 20: Zero-shot accuracy comparisons of DIET against baseline pruning algorithms on Gemma-2 9B on seven NLP benchmarks. The best performances are marked in **bold**, and the runner-up is marked with underline. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*. All values are reported as percentages (%).

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
No pruning (Original)	0	84.4	67.9	80.0	74.2	87.8	66.1	47.2	72.5
Magnitude - Dim	30%	53.9	52.7	32.6	51.9	44.1	22.6	25.0	40.4
SliceGPT	30%	37.8	52.7	25.6	50.1	27.8	24.9	28.0	35.3
PuDDing	30%	50.5	66.1	49.2	60.0	52.1	32.2	31.6	**48.8**
DIET (Ours)	30%	61.2	53.1	32.3	52.9	43.1	22.5	24.2	41.3

Table 21: Zero-shot accuracy comparisons of DIET against baseline pruning algorithms on Gemma-2 2B on XNLI (15 languages). The best average performances at each sparsity level are marked in **bold**. All values are reported as percentages (%).

Method	Sparsity	Per-language Accuracies (%)															Average Acc. (%)
		en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	
No pruning (Original)	0	53.9	49.3	46.8	49.0	39.1	44.1	49.0	45.0	33.2	47.0	44.4	35.5	43.5	41.0	34.1	43.7
Magnitude-Dim	10%	37.1	37.2	34.5	34.3	34.2	34.7	36.4	33.6	34.1	33.5	33.5	34.0	34.8	34.1	34.3	34.7
DIET (Ours)	10%	49.8	41.9	45.3	42.6	35.6	39.2	38.7	34.2	33.5	34.3	37.6	32.8	37.7	34.2	33.3	**38.0**
Magnitude-Dim	20%	35.0	34.9	33.7	33.4	33.6	33.7	36.4	32.9	33.6	33.1	33.5	33.3	36.1	33.6	33.4	34.0
DIET (Ours)	20%	40.0	34.1	36.6	33.7	33.4	34.1	33.5	33.4	33.2	33.4	33.6	33.3	35.1	34.6	33.3	**34.4**

Table 22: Zero-shot accuracy comparisons of DIET against baseline pruning algorithms on Gemma-2 9B on XNLI (15 languages). The best average performances at each sparsity level are marked in **bold**. All values are reported as percentages (%).

Method	Sparsity	Per-language Accuracies (%)															Average Acc. (%)
		en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	
No pruning (Original)	0	54.0	51.6	51.0	51.1	45.3	48.6	49.0	50.4	33.8	45.2	47.5	33.4	46.6	46.5	44.4	46.6
Magnitude-Dim	10%	52.3	44.0	49.5	49.0	42.7	38.6	41.5	42.9	33.6	47.4	38.8	33.3	40.2	40.7	34.1	41.9
DIET (Ours)	10%	52.6	50.2	47.0	48.7	41.6	37.4	47.3	44.7	34.0	42.9	40.8	33.9	44.5	38.8	38.1	**42.8**
Magnitude-Dim	20%	45.0	35.2	43.5	42.3	38.8	35.1	35.6	35.3	33.7	39.9	35.9	33.1	39.2	37.1	33.4	**37.5**
DIET (Ours)	20%	43.7	43.3	37.1	43.5	38.6	34.9	37.2	38.8	33.5	36.1	34.5	33.7	37.6	34.5	34.1	37.4

Table 23: Zero-shot accuracy comparisons on IFEval for Gemma-2 2B and 9B. All values are reported as accuracy scores.

Method	Sparsity	**Gemma-2 2B**				**Gemma-2 9B**			
		Strict Acc		Loose Acc		Strict Acc		Loose Acc	
		Prompt	Inst	Prompt	Inst	Prompt	Inst	Prompt	Inst
Original	0	0.1368	0.2710	0.1386	0.2758	0.1497	0.2698	0.1590	0.2770
Magnitude-Dim	10%	0.1331	0.2698	0.1405	0.2758	0.1497	0.2614	0.1608	0.2710
DIET (Ours)	10%	0.1035	0.2158	0.1183	0.2278	0.1553	0.2842	0.1590	0.2902
Magnitude-Dim	20%	0.1054	0.2302	0.1054	0.2302	0.1165	0.2398	0.1220	0.2434
DIET (Ours)	20%	0.1091	0.2206	0.1109	0.2230	0.1146	0.2326	0.1146	0.2326

Table 24: Zero-shot accuracy on Gemma-2 2B with masks derived from *arc_easy*, *arc_challenge*, and all tasks. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
Arc-Easy Mask	20%	62.9	52.5	39.0	53.1	48.6	25.2	26.8	44.0
Arc-Challenge Mask	20%	63.4	52.4	39.0	51.5	48.0	24.5	26.8	43.6
All Tasks (Ours)	20%	63.3	54.9	41.8	53.2	49.8	24.1	27.6	45.0

Table 25: Zero-shot accuracy comparisons of continuous score merging and our discrete (vote-based) mask on Gemma-2 2B at 20% sparsity. Tasks with * are presented with *acc_norm* and remaining tasks with *acc*.

Method	Sparsity	Per-task Accuracies (%)							Average Acc. (%)
		BoolQ	RTE	HellaSwag*	WinoGrande	Arc-E*	Arc-C*	OBQA*	
Continuous	20%	63.3	52.7	42.1	52.6	50.4	24.5	26.8	44.6
Voting (Ours)	20%	63.3	54.9	41.8	53.2	49.8	24.1	27.6	45.0