ShadowLLM: Predictor-based Contextual Sparsity for Large Language Models

Yash Akhauri[1,2], **Ahmed F. AbouElhamayed**[1], **Jordan Dotzel**[1,2], **Zhiru Zhang**[1], **Alexander M. Rush**[1], **Safeen Huda**[2], and **Mohamed S. Abdelfattah**[1]
[1]Cornell University [2]Google
{ya255, afa55, jad443}@cornell.edu
{zhiruz, arush, mohamed}@cornell.edu, safeen@google.com

Abstract

The high power consumption and latency-sensitive deployments of large language models (LLMs) have motivated efficiency techniques like quantization and sparsity. *Contextual sparsity*, where the sparsity pattern is input-dependent, is crucial in LLMs because the permanent removal of attention heads or neurons from LLMs can significantly degrade accuracy. Prior work has attempted to model contextual sparsity using neural networks trained to predict activation magnitudes, which can be used to dynamically prune structures with low predicted activation magnitude. In this paper, we look beyond magnitude-based pruning criteria to assess attention head and neuron importance in LLMs. We develop a novel predictor called ShadowLLM, which can *shadow* the LLM behavior and enforce better sparsity patterns, resulting in over 15% improvement in end-to-end accuracy compared to prior methods. In addition, ShadowLLM achieves up to a 20% speed-up over the state-of-the-art DejaVu framework. These enhancements are validated on Llama-2 and OPT models with up to 30 billion parameters. Our code is available at ShadowLLM.

1 Introduction

Large language models (LLMs) are emerging as a core component of many computing applications. Their ability to perform in-context learning, i.e., to perform a task by conditioning on examples without any gradient updates (Brown et al., 2020; Liang et al., 2022; Min et al., 2022), make them broadly applicable to numerous applications. Yet, their large size combined with the latency-sensitivity of LLM-based applications make them expensive to deploy (Hoffmann et al., 2022).

A key optimization in LLM deployment is sparsification, where weights or activations are pruned to reduce the computation and memory requirements at run time. Sparsification can either be *static*, which permanently removes an attention head or neuron, or *contextual*, which prunes based on the current input and context. While some works investigate task-specific static pruning methods (Bansal et al., 2022; Michel et al., 2019), they typically have a large impact on in-context learning, reducing downstream task accuracy compared to contextual sparsity.

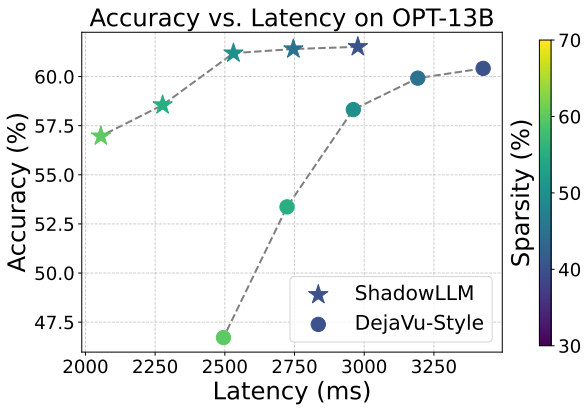

Figure 1: ShadowLLM uses more accurate pruning criteria and a simpler sparsity predictor compared to DejaVu. Its pruning criteria results in a stronger accuracy-sparsity trade-off (geomean) across seven downstream evaluation tasks, and its unified predictor improves the execution latency compared to the layerwise predictor of DejaVu.

Contextual sparsity can be leveraged at run time to dynamically prune LLMs, yet it requires making fast and accurate predictions based on predetermined *pruning criteria*. These criteria can have large effects on the overall accuracy and performance of the model, as shown in Figure 1. Our method ShadowLLM uses more accurate pruning criteria and a unified predictor at the beginning of the model, which leads to a stronger accuracy-performance tradeoff compared to the recent work DejaVu (Liu et al., 2023).

Both of these methods dynamically vary their

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 19154–19167
November 12-16, 2024

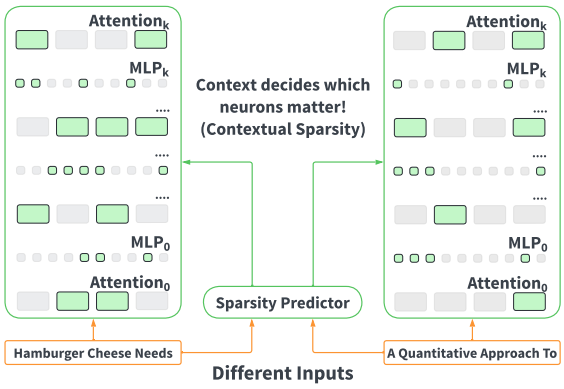

Figure 2: Contextual sparsity prunes neurons and attention heads based on the context (input) itself. Training a predictor to dynamically predict the sparsity pattern dependent on the input tokens can improve model quality.

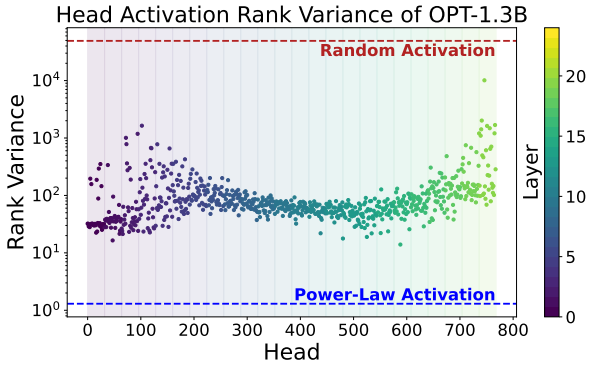

Figure 3: Heads with higher rank variance, calculated using `GradNorm`, indicate greater context dependence. This context dependence, or contextual sparsity, is most noticeable in the early and later layers of the OPT-1.3B model. We measured the variance in rank for each head across 5000 inputs in seven five-shot evaluation tasks.

sparsity patterns given different inputs using sparsity predictors, as shown in Figure 2. The inputs are passed into a sparsity predictor, which then outputs the corresponding per-layer masks on the attention and MLP layers. For DejaVu, the sparsity pattern is generated with neural-network predictors at each layer. This gives access to more local information, but layerwise predictors come with an expensive run-time cost.

On the model quality side, contextual sparsity exists if there is a significant variance on head and neuron importance as the input changes. Figure 3 quantifies this variance on the importance (ranks) of attention heads on OPT-1.3B across different inputs. It demonstrates the relative importance changes significantly, especially in the earlier and the later layers. Naturally, this variance across inputs necessitates a dynamic pruning strategy to ensure an appropriate quality–latency trade-off.

In this work, we explore the effects of different pruning criteria and predictor design on LLM accuracy and latency. Our contributions are summarized below:

1. **Pruning Criteria:** We evaluate approaches from prior pruning research to find head and neuron pruning criteria that can improve downstream zero-shot accuracy by 15% without affecting performance.

2. **Early Prediction:** We use a single predictor at the first layer of the LLM to model the entire LLM sparsity pattern, improving performance by 20.6% without affecting accuracy.

2 Related Work

2.1 Pruning Criteria

Research in the area of designing criteria for pruning neurons has focused on using the activations, weights, and gradients of neural networks to assess the relative importance of neurons. Several pruning criteria have been designed to utilize light-weight computations, such as a single forward-backward pass through the network. For instance, some works use parameter magnitudes as a proxy for parameter saliency (Frankle and Carbin, 2018; Han et al., 2015), whereas others use gradient-based information (LeCun et al., 1989; Hassibi and Stork, 1992; Molchanov et al., 2016; Bansal et al., 2022). Further, research in Neural Architecture Search (NAS) adapts these pruning criteria to assess and compare entire architectures. Such initialization-based measures like NASWOT (Mellor et al., 2021) aim to study other properties of the architecture, and can be used to study neuron importance as well.

In this work, we adapt several neuron importance criteria from research in pruning and NAS (Abdelfattah et al., 2021; Lopes et al., 2021; Mellor et al., 2021; Turner et al., 2019) to evaluate which methods work well for dynamic pruning of large language models at run time.

2.2 LLM Inference Optimization

Given the recent exponential increase in model size, significant research has been dedicated to optimizing NN inference to decrease compute, power, and latency. Quantization reduces the precision

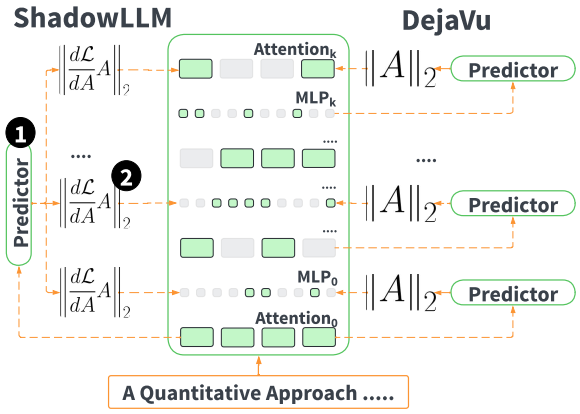

Figure 4: **(1)** A single predictor to model the entire LLM improves model performance, while **(2)** utilizing gradient based information when evaluating pruning criteria for neurons improves model quality.

of model parameters, embeddings, and key-value caches (Zhang et al., 2023; Dotzel et al., 2024; Zhao et al., 2024). Orthogonal to quantization, there has been research on accelerating sparse language models, which either statically or dynamically trim portions of compute throughout the model (Hua et al., 2019; Schuster et al., 2022; Elbayad et al., 2020).

Within these works, DejaVu (Liu et al., 2023) leverages dynamic sparsity by building predictors to estimate sparsity patterns. In this paper, we investigate how the predictor can be improved, both in terms of performance and model quality.

3 Pruning Criteria

Contextual sparsity requires dynamically understanding which neurons to prune (i.e. assessing the neurons importance relative to an input) and ranking the neurons relative to each other. Figure 4 depicts how we can use information about the activations and gradients to prune a LLM for this contextual sparsity.

Consider a model $\mathcal{M}$ and dataset $\mathcal{D}$, containing prompts (inputs) along with the target output sequence. We then wish to define performance on the dataset as $\mathcal{P}_{\mathcal{M}}(\mathcal{D})$. Now suppose a subset of the model $\mathcal{C} \subset \mathcal{M}$ is pruned out. Ideally, we would like to be able to estimate $\mathcal{P}_{\mathcal{M}}(\mathcal{D}) - \mathcal{P}_{\mathcal{M}\setminus\mathcal{C}}(\mathcal{D})$ (Bansal et al., 2022).

The optimal pruning strategy is found in Equation 1. If we look at aggressive attention head pruning of even small transformers (prune 56 out of 64 heads in each layer), exhaustive search in a single layer would require ${}^{64}C_8$ evaluations, and this would have to be repeated for every layer, making the problem intractable.

$$\arg\min_{\mathcal{C}\subset\mathcal{M}} \quad \mathcal{P}_{\mathcal{M}}(\mathcal{D}) - \mathcal{P}_{\mathcal{M}\setminus\mathcal{C}}(\mathcal{D}) \tag{1}$$

We can feed a subset of the dataset $d \in \mathcal{D}$ to the model $\mathcal{M}$, and calculate the loss $\mathcal{L}$. Further, we can also get access to the activations (A), as well as the parameters of the up-projection FFN of transformer at layer l as θ_l. The activation at layer l for the k^{th} head or neuron is denoted as $A_{l,k}$. The gradients for these activations are denoted as $\frac{\partial \mathcal{L}}{\partial A_{l,k}}$. The gradient for the weight parameters of the k^{th} neuron in the up-projection FFN at layer l is given as $\frac{\partial \mathcal{L}}{\partial \theta_{l,k}}$.

Current predictor-based sparsity research investigates the impact of magnitude-based criteria, such as the `L2Norm` of the head and neuron activation on a subset of data d. The intuition is that the heads that are more *activated* should be more important. There is significant research on other criteria for pruning weights and activations (Molchanov et al., 2016). Beyond activation magnitude being a criterion for importance, the process of pruning can also be framed as an optimization problem, with the goal of approximating the change in loss from removing parameters. Methods such as optimal brain damage (OBD) (LeCun et al., 1989) rely on the gradient of the loss with respect to the feature maps. While OBD evaluates the second-order term (Hessian), works such as (Figurnov et al., 2016; Molchanov et al., 2016) come up with similar metrics based on the Taylor expansion of the change in loss.

In this paper, we evaluate pruning criteria of varying complexity, that use **(1)** Activation Methods, **(2)** First-Order Gradient (Jacobian) Methods, **(3)** Activation + Jacobian Methods, **(4)** OBD-style Hessian Methods and **(5)** Sensitivity-Based Methods for pruning LLMs.

Among these methods, we find that a gradient-based sensitivity method we call `plainact` outperforms activation-based magnitude methods adapted in prior dynamic pruning research (Liu et al., 2023). The `L2Norm` activation-magnitude based criterion assesses the importance of neurons by simply taking the L2 Norm of the head and neuron activation as $||A_{l,k}||_2$. The `plainact` criterion measures the expected sensitivity of the model on the loss if a head or neuron is removed. For the head and neuron, this can be described simply as $||A_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial A_{l,k}}||_1$ and $||\theta_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial \theta_{l,k}}||_1$ respectively. We

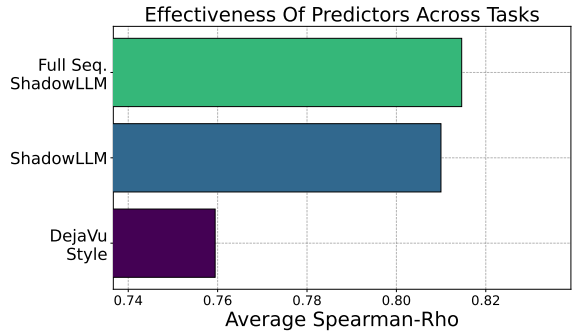

Figure 5: Head importance ranking ability of different sparsity predictors on 500 queries across 7 downstream tasks. A single predictor at the start of the transformer can accurately model the global relative head and neuron importance.

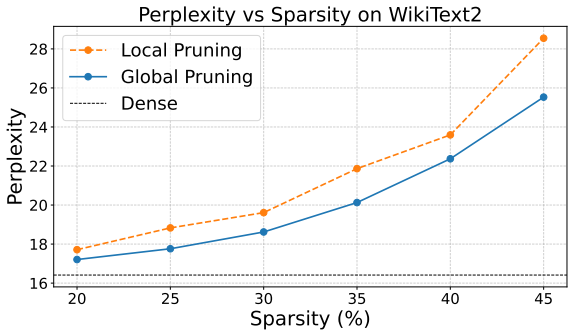

Figure 6: Global pruning outperforms local (per-layer) pruning strategies using ShadowLLM trained on the `plainact` criteria (OPT-1.3B). Global pruning accommodates the varying importance of different layers, allowing for unbalanced pruning across layers.

perform in-depth ablations across several pruning criteria in Section 6, and find that `plainact` empirically performs well as a pruning criteria.

4 Predictors For Neuron Ranking

When deploying a large language model, for a given input, we will not have access to the activations or the gradients. Thus, calculating the `L2Norm` or `plainact` criterion is not possible. However, it is possible to create a calibration dataset of inputs and their corresponding `L2Norm` or `plainact` for each head and neuron. Such a dataset can be used to train a predictor, which can take the input and predict the sparsity pattern of the model at deployment. Sparsity prediction can reduce the end-to-end latency of transformer inference by predicting which operations can be skipped.

We propose a method called ShadowLLM that uses the first layer's attention output to predict the sparsity pattern for the entire model. This reduces the overhead of repeatedly calling the predictor at each layer and cuts the total FLOPs of the predictor by 20%, as shown in Table 1. ShadowLLM uses the activation of the first layer, which is not pruned, to predict the sparsity pattern for subsequent layers.

We also explore a *Full Sequence ShadowLLM*, which uses a small transformer to take in the entire input token embedding and predict the sparsity pattern, allowing pruning of the first transformer layer as well. However, the *Full Sequence ShadowLLM* requires an additional $2(2E^2 + EL^2)$ FLOPs, making it as costly as running an entire dense attention layer and impractical due to the high computational cost.

DejaVu employs a two-layer MLP, taking the activation from the final token at every alternating layer and predicting the sparsity of the next layer. A significant portion of the complexity of the DejaVu system arises from its asynchronous look-ahead predictor which can be expensive in wall clock time despite aggressive optimizations within DejaVu. The predictor itself only takes 2% of the total FLOPs for OPT-1.3B, but having a per-layer predictor adds significant overhead due to additional GPU kernel launches and memory bandwidth constraints, leading to an end-to-end latency increase of 25% over static sparsity (same sparsity but fixed, without a predictor).

DejaVu's per-layer approach to pruning, where a fixed sparsity is enforced per layer, can be suboptimal as true contextual sparsity should be independent of layers. To study our proposed predictor designs in a contextual-sparsification setting, we evaluate the Spearman-ρ (rank correlation coefficient) between the relative importance order

Predictor	**FLOPs Equation**
DejaVu	$N\left(Ep_1 + p_1(H+F)\right)$
ShadowLLM	$\left(Ep_1 + p_1(N(H+F))\right)$
Model	**ShadowLLM FLOPs Reduction**
OPT-1.3B	19.11%
OPT-30B	19.55%
OPT-175B	19.76%

Table 1: For a transformer with E embedding dimension, N layers, H heads, F FFN neurons per layer, ShadowLLM uses (N-1)Ep_1 fewer FLOPs, where p_1 is the predictor hidden dimension. The table also shows the percentage improvement in predictor FLOPs for ShadowLLM vs. DejaVu for different models.

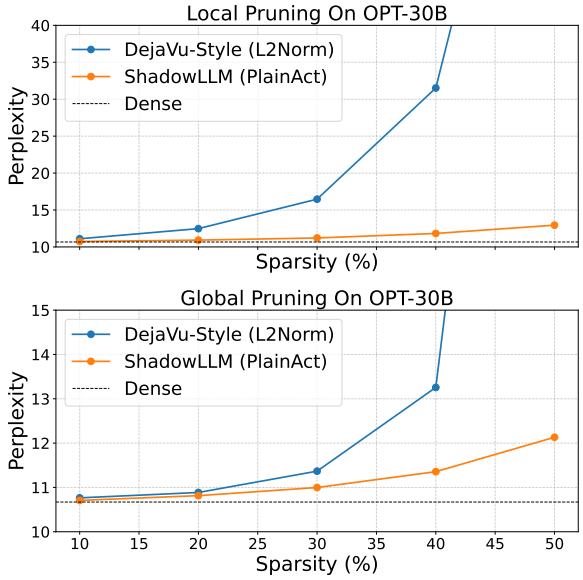

Figure 7: Comparison of the DejaVu-style predictor trained on a magnitude-based metric (`L2Norm`) with ShadowLLM on the best pruning criteria (`plainact`) on WikiText2. In both local and global settings, ShadowLLM performs well due to better pruning criteria.

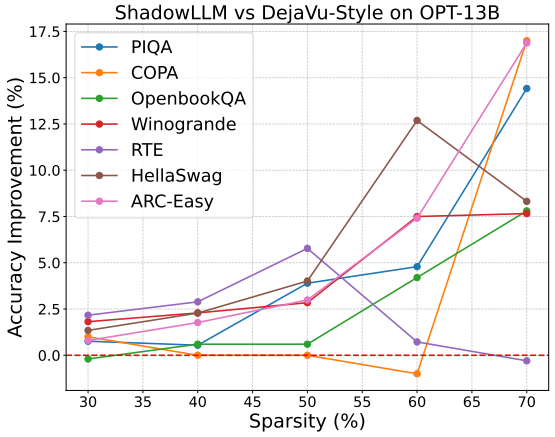

Figure 8: Consistent accuracy improvement of ShadowLLM over DejaVu across seven downstream eval tasks in the zero-shot shot setting.

Method	Latency (ms)	Accuracy (%)
Static	2014	55.34
Dense	2609	58.32
DejaVu	2981	59.28
ShadowLLM	**2562**	**61.19**

Table 2: Latency and accuracy comparison of different methods at 50% sparsity. Average zero-shot accuracy across 7 downstream tasks reported on OPT-13B.

of neurons and heads given by the predictor, and the relative importance order given by the pruning criterion. Additionally, this is done on a global head-ranking task. From Figure 5, we see that DejaVu-style layer-wise predictors are not trained for global pruning. We find that *Full Seq. ShadowLLM* performs similarly to ShadowLLM, but with a significant increase in overall FLOPs.

To analyze the ability of ShadowLLM predictors to assess neuron importance in a global and local (per-layer) setting, we train ShadowLLM predictors for `plainact` across all seven down-stream tasks in the 5-shot setting, with a per-layer (local) output normalization scheme. We then evaluate the WikiText2 perplexity [1] of the OPT-1.3B model as we increase sparsity for both the global and local (per-layer) pruning strategies. For local pruning, every layer achieves the target sparsity, and relative importance are only compared intra-layer. In Figure 6, we find that ShadowLLM is able to preserve perplexity in both global and local cases. However, we find that global pruning generally performs better than per-layer pruning. This can be attributed to the fact that some layer heads are more important than others, and forcing equal pruning ratios for all layers may cause over-parameterization in some layers and more important head to be pruned out from an under-parameterized layer.

5 Evaluation

We find that the activation-gradient based pruning criteria that we use in ShadowLLM are effective for downstream evaluation tasks as well as perplexity. Further, we demonstrated in Section 4 that ShadowLLM can predict the sparsity pattern for the entire LLM given just the input to the first layer. We find that the accuracy and *predictability* trade-off is excellent for the `plainact` criterion, whereas other criteria were harder to learn due to outliers and high variance. In this section, we evaluate the effectiveness of combining ShadowLLM predictor design with the `plainact` criterion, compared to our implementation of DejaVu-style[2] predictors, trained on a magnitude based pruning criteria.

5.1 Experimental Setup

We evaluate the perplexity for the WikiText2 (Merity et al., 2016) language modeling dataset, and accuracy on 7 few-shot downstream tasks:

[1] For effective context-sparsity evaluation, perplexity calculations are performed on a per-document basis, differing from standard concatenation methods; see Section B for details.

[2] To enable comparisons across pruning criteria, we have implemented our own DejaVu-style predictor in ShadowLLM.

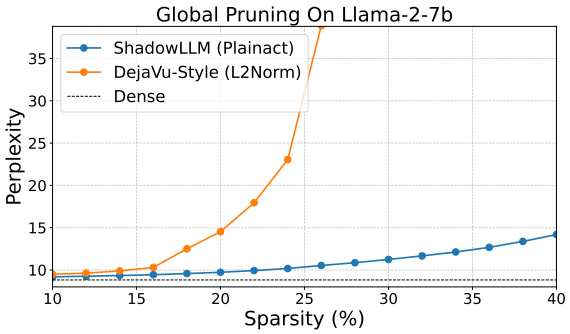

Figure 9: Gradient-informed criteria (`Plainact`) improves global pruning on Llama-2-7b, resulting in an end-to-end perplexity improvement on WikiText2.

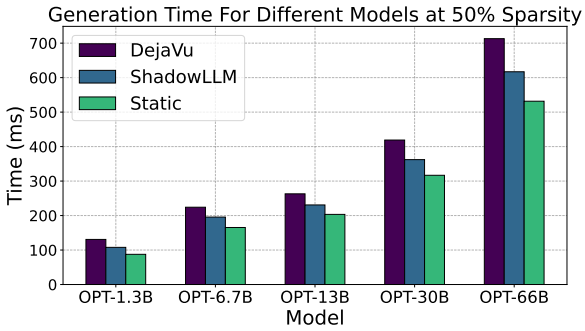

Figure 10: Average time per-inference with prompt length = 128 and generation length = 128 across model sizes. Sparsity is around 50%.

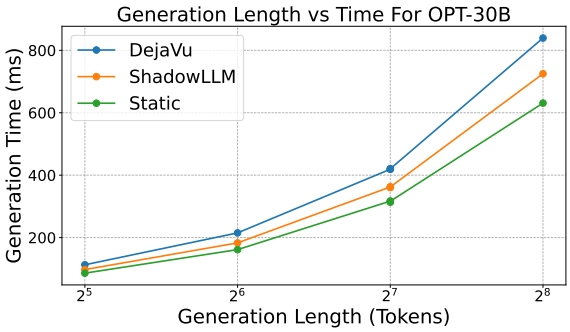

Figure 11: Average generation time on OPT-30B with prompt length = 128 as generation length increases. Sparsity is around 50%.

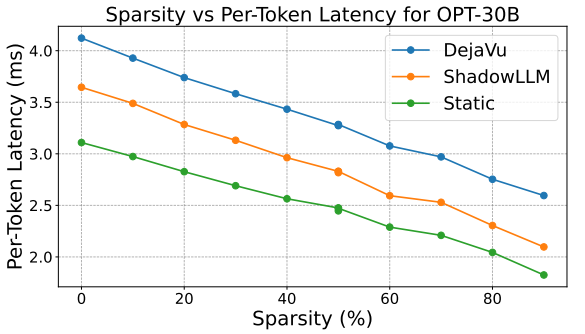

Figure 12: Per-token latency of OPT-30b with prompt length = 128 and generation length = 128 as sparsity increases.

PIQA (Bisk et al., 2020), COPA (Gordon et al., 2012), OpenBookQA (Mihaylov et al., 2018), Winogrande (Sakaguchi et al., 2019), RTE (Giampiccolo et al., 2007), HellaSwag (Zellers et al., 2019), and ARC-Easy (Clark et al., 2018) with lm-eval-harness (Gao et al., 2023).

Our ablation studies to identify good pruning criteria, as well as test the efficacy of predictors is conducted on OPT-1.3B. Further, local and global pruning strategies are tested on OPT-13B and OPT-30B, and global pruning on Llama-2-7b (Touvron et al., 2023). Our downstream evaluation across seven tasks is reported on OPT-13B.

5.2 Model Quality

In Figure 7, we train the DejaVu-style and ShadowLLM predictors on their respective pruning criterion (`L2Norm` and `plainact` respectively) on 2720 input-output examples across 7 downstream tasks in the five-shot setting. The perplexity is evaluated in a local and global pruning setting on WikiText2.

Global pruning enables better model quality - sparsity trade-off. Figure 9 compares the perplexity-sparsity trade-off on Llama-2-7b model, with `Plainact` significantly improving perplexity. Further, in Figure 8 we evaluate OPT-13B by training the ShadowLLM and DejaVu-style predictors in the same setting, and doing downstream evaluation in the zero-shot setting across seven tasks. We show that there is a consistent accuracy improvement across tasks, attributed to better pruning criteria.

We also validate these findings on OPT-13B in Figure 17 in the global pruning setting. These improvements are largely due to an improved pruning criterion, emphasizing the importance of pruning criteria that go beyond magnitude based strategies. From Table 2 we see that ShadowLLM with the `plainact` metric delivers 14% lower latency with 1.91% higher accuracy than DejaVu-style predictor.

5.3 Performance

DejaVu-style predictors can also be trained on better pruning criteria (`plainact`), giving improvements in accuracy. However, a single predictor can model these criteria and also offer improved end-

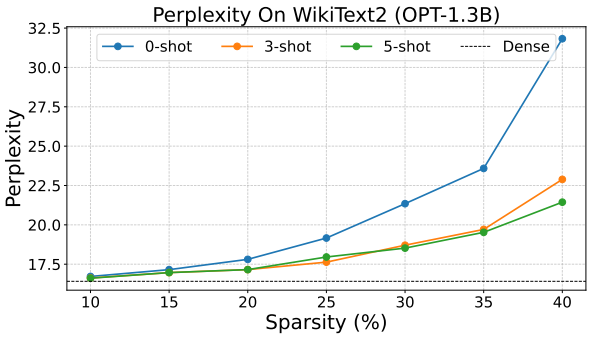

Figure 13: Calculating pruning criteria in a few-shot setting improves its ability to identify important heads and neurons.

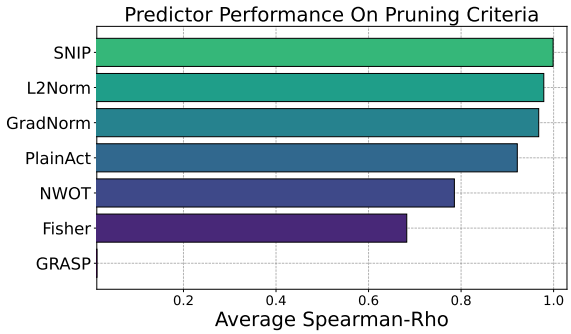

Figure 14: `plainact` is a good pruning criterion, and is also easy to learn. `grasp` has $\approx 4\times$ more outliers in proxy scores, making prediction more difficult.

to-end latency due to early prediction. It is also easier to integrate, without concerns for continuous pipelining and scheduling of a layer-wise predictor. In this section, we investigate the performance improvement ShadowLLM delivers by simplifying the DejaVu sparsity predictor implementation, and compare with DejaVu (Liu et al., 2023)

DejaVu implements hardware-efficient sparsity acceleration, which employs kernel fusion and memory coalescing to reduce overhead of the sparse matrix-vector multiply. These techniques already yield a $2\times$ speed-up over prior SoTA FasterTransformer implementations. However, the interleaved sparsity predictors have a significant overhead, leading to a performance degradation of over 25% with only a 2% increase in total FLOPs.

We implement the ShadowLLM predictor along with the prior enhancements introduced by DejaVu and conduct our performance experiments on up to 4 A100 GPUs. In Figure 10, we measure the end-to-end time taken for a generation length of 128 tokens at 50% sparsity and observe an average 16.2% improvement over DejaVu. Figure 11 shows a consistent improvement in generation time as output tokens increase. Further, Figure 12 shows that ShadowLLM is on average 21.25% faster than DejaVu in the decode phase specifically. Finally, we profile model sizes from 1.3B to 66B, observing up to a 21.3% improvement in time per-inference.

6 Analysis

Overview of Pruning Criteria

In Section 3, we categorize pruning criteria into five primary methods: Activation Methods, First-Order Gradient (Jacobian) Methods, Activation + Jacobian Methods, OBD-style Hessian Methods, and Sensitivity-Based Methods.

We begin by looking at activation magnitude based pruning methods akin to (Frankle and Carbin, 2018; Han et al., 2015). One such criterion, the `L2Norm` of the k^{th} attention head and FFN neuron is simply $||A_{l,k}||_2$. More advanced methods that use gradients may provide better information about neuron importance. `GradNorm` of the k^{th} attention head and FFN neuron is defined simply as $||\frac{\partial \mathcal{L}}{\partial A_{l,k}}||_2$ and $||\frac{\partial \mathcal{L}}{\partial \theta_{l,k}}||_2$ respectively. In our analysis, we found that methods that combine both the activation and Jacobian (Gradient) information perform the best. The `plainact` criterion adapted from (Bansal et al., 2022; Molchanov et al., 2016) can be defined as $||A_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial A_{l,k}}||_1$ and $||\theta_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial \theta_{l,k}}||_1$ respectively. Similar to `plainact` the `fisher` criterion can be defined as $\langle (A_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial A_{l,k}})^2 \rangle$ and $\langle (\theta_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial \theta_{l,k}})^2 \rangle$ respectively, denoting a similar criterion but aggregated in a different manner.

The `grasp` criterion approximates the change in gradient norm, which requires the Hessian H and is calculated as $|| - (H_{l,k} \cdot \frac{\partial \mathcal{L}}{\partial A_{l,k}}) \odot A_{l,k}||_1$. This OBD-style Hessian method (Wang et al., 2020) worked well in downstream-evaluation tasks, but did not deliver good perplexity.

NASWOT (Mellor et al., 2021) introduces a sensitivity based method called `jacov`. The `jacov` criterion measures the covariance of the Jacobian matrices across a mini-batch of data. `epenas` (Lopes et al., 2021) follows the same principles as `jacov`. Naturally, as `jacov` rely on aggregated Jacobian matrices over a batch of data, this criterion cannot trivially exist for input-dependent (contextual sparsity) use-case. To test these criteria, we register the activations for the heads and neurons across the entire downstream task dataset, and generate a

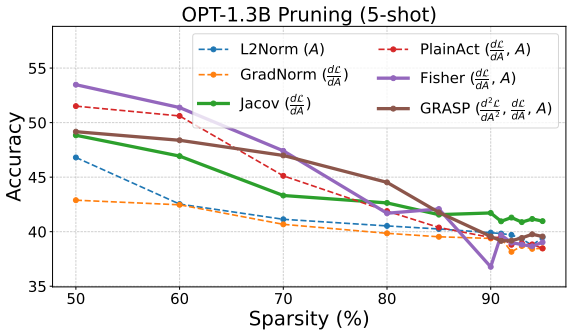

Figure 15: For every criterion, the corresponding aggregated neuron importance is used to conduct static pruning of the LLM at test time, and the average accuracy is reported.

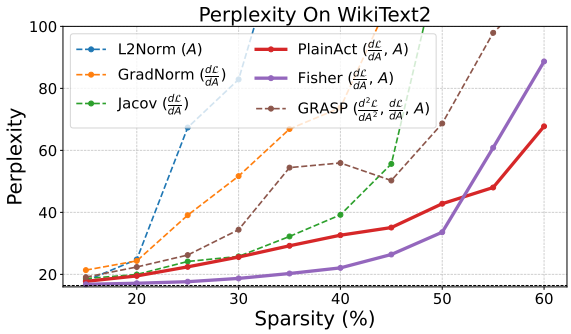

Figure 16: The aggregate importance score per neuron is used to conduct static pruning of the LLM. `fisher` and `plainact` preserve model quality better than other criteria. Dashed black line is dense baseline.

single *aggregate head importance.*

We evaluate the effectiveness of several pruning criteria by using them as metrics for removing less important heads/neurons. Our analysis includes evaluating the perplexity for the WikiText2 (Merity et al., 2016) language modeling dataset and accuracy on 7 few-shot downstream tasks.

Enhancing Pruning with Few-Shot Examples

In Figure 13, we calculate the `fisher` criteria for every neuron and head on 2720 input-output examples from the downstream tasks for the 0-shot, 3-shot, and 5-shot settings. We average the criteria for each neuron and head across these examples and evaluate WikiText2 perplexity as model sparsity is increased. The results indicate that providing more in-context examples when registering the criteria improves model quality during pruning.

Advantages of Gradient-Informed Criteria

In Figure 15, we use the task pruning criterion averaged over their respective examples for each head and neuron to do a static sparsification of the OPT-1.3B model and test it in the 0-shot setting. We report the mean accuracy across the downstream tasks for each pruning criteria. We find that `jacov` is a stable criteria to preserve model performance in the static case. However, `jacov` does not have a context-dependent equivalent, as it relies on the covariance of Jacobian matrices across examples. We evaluate these proxies in Figure 16, and find that `fisher` and `plainact` preserve model quality well, with `jacov` performing worse. `jacov` might have higher task-dependence for static pruning, and does not translate to better general model quality.

Learning Pruning Criteria with Predictors

While we can *shadow* activation magnitudes with a predictor, we need to balance finding the best pruning criteria for assessing neuron importance, ensuring the criteria is easy to learn. To identify such a criteria, we measure each criteria for each head and neuron on 2720 input-output examples across the 7 downstream tasks in a 5-shot setting. We train our predictor to use the output of the first attention layer's last sequence index to predict per-head and neuron importance. Figure 14 reports the average Spearman-ρ rank correlation on 680 input-output examples. From Figure 16, we see that `fisher` delivers the best perplexity for up to 50% sparsity, but delivers a Spearman-ρ of under 0.7. Similarly, `grasp` is difficult to predict due to its high range and outliers. In contrast, we find that the `plainact` criterion is easy to predict and performs well in a contextual setting.

7 Conclusion

In this paper, we present ShadowLLM, a novel approach that realizes contextual sparsity in large language models by using a gradient-informed pruning criterion. We demonstrate that these criteria can be effectively modeled by a single predictor at the first layer of the LLM, eliminating the need for per-layer prediction. Our findings, validated on models with up to 30 billion parameters, show that relatively small predictors can model contextual sparsity in LLMs. This approach, combining an improved pruning criterion with an early predictor, enables over 15% improvement in accuracy without a latency trade-off and a 20% improvement in performance across different model sizes.

Limitations

In this paper, we work towards significantly simplifying the predictor design, and study several pruning criteria. However, our study is limited to smaller models, up to 30B parameters on only OPT style models. Further, criteria like `nwot` are designed for the ReLU activation function, which may not be directly applicable to attention maps. We profile these for completeness regardless, however, more research in pruning criteria is needed. Finally, we train predictors on less than 10000 input-output examples, more examples may enable better sparsity pattern modeling.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 2339084, in addition to funding by Intel Corporation. We would like to thank Nilesh Jain, Juan Pablo Munoz, Sameh Gobriel, and Vui Seng Chua for helpful discussions and feedback.

References

Mohamed S Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas Donald Lane. 2021. Zero-cost proxies for lightweight {nas}. In *Int. Conf. Learn. Represent.*

Hritik Bansal, Karthik Gopalakrishnan, Saket Dingliwal, S. Bodapati, Katrin Kirchhoff, and Dan Roth. 2022. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. In *Annual Meeting of the Association for Computational Linguistics*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Adv. Neural Inform. Process. Syst.*, 33:1877–1901.

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

Jordan Dotzel, Yuzong Chen, Bahaa Kotb, Sushma Prasad, Gang Wu, Sheng Li, Mohamed S. Abdelfattah, and Zhiru Zhang. 2024. Learning from students: Applying t-distributions to explore accurate and efficient formats for llms. *Int. Conf. Machine Learning*.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. 2020. Depth-adaptive transformer. *Int. Conf. Learn. Represent.*

Mikhail Figurnov, Aizhan Ibraimova, Dmitry P Vetrov, and Pushmeet Kohli. 2016. Perforatedcnns: Acceleration through elimination of redundant convolutions. *Advances in neural information processing systems*, 29.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. Semeval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

Babak Hassibi and David Stork. 1992. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Weizhe Hua, Yuan Zhou, Christopher De Sa, Zhiru Zhang, and G. Edward Suh. 2019. Channel gating neural networks. *Adv. Neural Inform. Process. Syst.*

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. In *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. 2018. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for llm compression and acceleration. *Preprint*, arXiv:2306.00978.

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

Vasco Lopes, Saeid Alirezazadeh, and Luís A. Alexandre. 2021. *EPE-NAS: Efficient Performance Estimation Without Training for Neural Architecture Search*, page 552–563. Springer International Publishing.

Joseph Mellor, Jack Turner, Amos Storkey, and Elliot J. Crowley. 2021. Neural architecture search without training. *Preprint*, arXiv:2006.04647.

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

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Adv. Neural Inform. Process. Syst.*, 32.

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. *arXiv preprint arXiv:1809.02789*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *Preprint*, arXiv:1907.10641.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. 2022. Confident adaptive language modeling. *Adv. Neural Inform. Process. Syst.*

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Jack Turner, Elliot J Crowley, Michael O'Boyle, Amos Storkey, and Gavin Gray. 2019. Blockswap: Fisher-guided block substitution for network compression on a budget. *arXiv preprint arXiv:1906.04113*.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. 2020. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*.

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*.

Yichi Zhang, Ankush Garg, Yuan Cao, Łukasz Lew, Behrooz Ghorbani, Zhiru Zhang, and Orhan Firat. 2023. Binarized neural machine translation. *Adv. Neural Inform. Process. Syst.*

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. 2024. Atom: Low-bit quantization for efficient and accurate llm serving. *Conf. on Machine Learning and Systems*.

A Appendix

A.1 Predictor Design

Hyper-parameter	Value
Hidden Layers	1
Hidden Layer Neurons	2048
Activation Function	ReLU
Input Dimension	Model Embedding
Output Dimension	Number Of Neurons
Number of Epochs	100
Batch Size	32
Optimizer	AdamW
Learning Rate	0.001
Scheduler	CosineAnnealingLR
Criterion	MSELoss

Table 3: Hyperparameters for DejaVu-style and ShadowLLM predictor training.

A.2 Additional Pruning Criteria

In this section, we provide a more complete view of the proxies we investigate and their results.

While some criteria were designed for activations (`Fisher`), whereas others for weights (`snip`), we extend the pruning criteria to both activations for attention heads and weights for FFN neurons. A side-effect of this is that criteria such as `plainact` and `fisher` look similar, but are aggregated in different ways (L1 Norm versus Mean). We maintain both variants in our analysis for completeness.

Similar to `jacov`, `epenas` is also a viable method for non-contextual sparsity. `epenas` measures the intra- and inter-class correlations of the Jacobian matrices. We modify `epenas` by treating next-tokens as the class that the Jacobians are registered as.

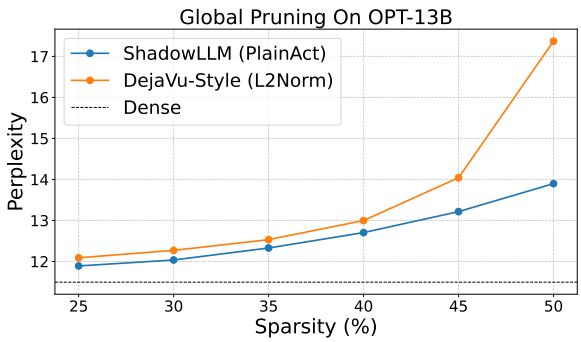

Figure 17: A better criteria (`plainact`) with the ShadowLLM predictor improves perplexity-sparsity trade-off on WikiText2.

Model	Ours	Reported (Lin et al., 2024)
Llama2-7B	8.82	5.47
OPT-1.3B	16.4	14.6
OPT-13B	11.5	10.1
OPT-30B	10.7	9.56

Table 4: Comparison of our perplexity and reported perplexity for various models. Following (Bansal et al., 2022), we calculate per-document perplexity, which increases model perplexity.

We also investigate sensitivity-based methods, such as `snip` (Lee et al., 2018), defined in Equation 2, which investigates how removing a single neuron in isolation will impact the loss.

$$\texttt{snip} = \lim_{\varepsilon \to 0} \left| \frac{\mathcal{L}_\theta - \mathcal{L}_{\theta+\varepsilon\delta_q}}{\varepsilon} \right| \tag{2}$$

Further, we adapt proxies from neural architecture search for neuron saliency. The NASWOT (Mellor et al., 2021) paper introduces two criteria, the first we refer to as `nwot`. `nwot` calculates the determinant of a Hamming distance-based kernel matrix, which measures the similarity of binary codes that result after the ReLU, given an input in the neural network. This uses the intuition that if two similar inputs lie within the same linear region of the network, they will be difficult to disentangle. `nwot` is defined in Equation 3.

$$\texttt{nwot}_{l,k} = \log\left(\frac{1}{\text{seqlen}} \sum_{i=1}^{\text{seqlen}} (1 - A^i_{l,k})^2\right) \tag{3}$$

B On Perplexity Calculation

In our experiments, we evaluate the perplexity of language models on the WikiText-2 dataset **by computing the log-likelihood of each document individually, rather than concatenating all documents into a single continuous text stream.** Specifically, we process each document separately, calculating perplexity within the that document's context. This approach limits the context to within individual documents, without leveraging cross-document dependencies that the standard concatenation method from reference works provide (e.g. `"\n\n".join(wikitext_docs['text'])`). As a result, our perplexity scores reflect the model's performance on isolated text segments, which may

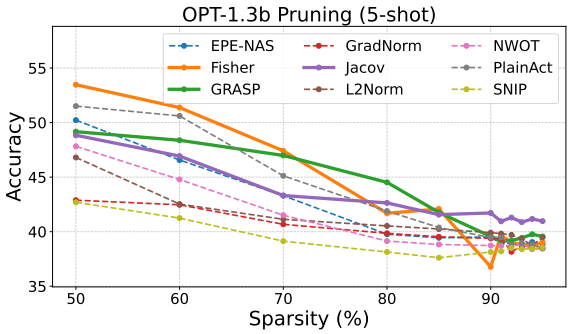

Figure 18: Each pruning criterion is measured and averaged per neuron and head over 3500 training examples in a 5-shot setting across 7 downstream tasks. For every criterion, the corresponding aggregated neuron and head importance is used to conduct static pruning of the LLM at test time. For each criterion, mean of accuracy is reported as sparsity is increased.

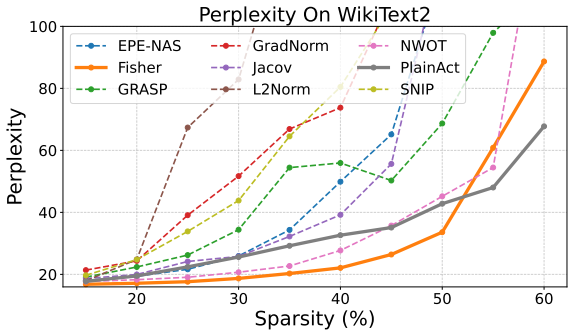

Figure 19: Each pruning criterion is measured and averaged per neuron and head over 2720 training examples in a 5-shot setting from all downstream tasks. This aggregate importance score per neuron and head is used to conduct static pruning of the LLM when testing perplexity on WikiText2. `fisher` and `plainact` preserve model quality better than other criteria.

differ from scores obtained using the more conventional concatenated approach. While this methodology deviates from standard practice, it offers a consistent evaluation of the model's capabilities within document-level context, aligning with the setting considered in our study as well as the in-context learning literature we build our study on (Bansal et al., 2022). We quantify this difference in perplexities in Table 4.

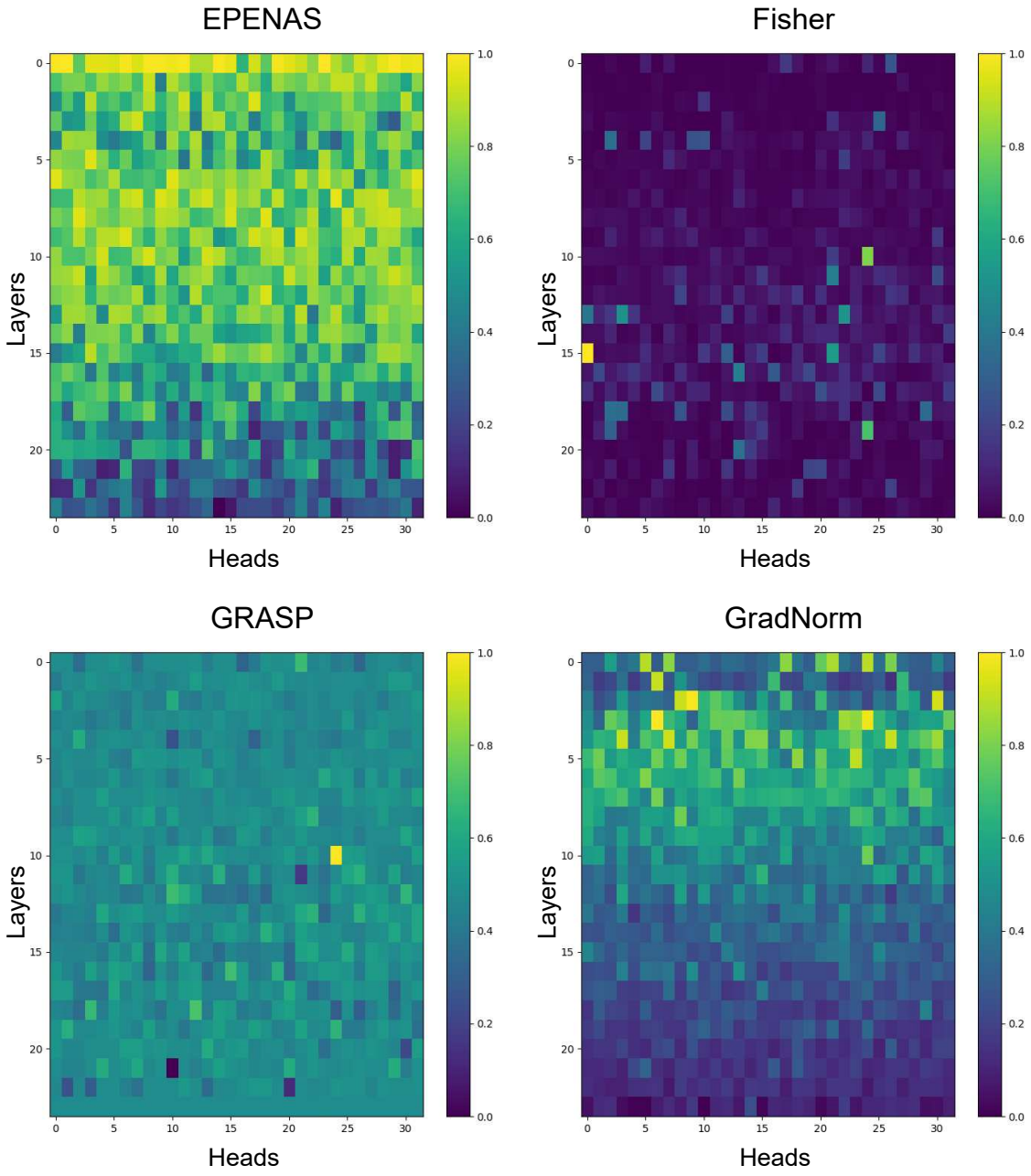

Figure 20: Aggregated pruning criteria scores per-head for the OPT-1.3B model, over the ARC-Easy training task in a five-shot setting.

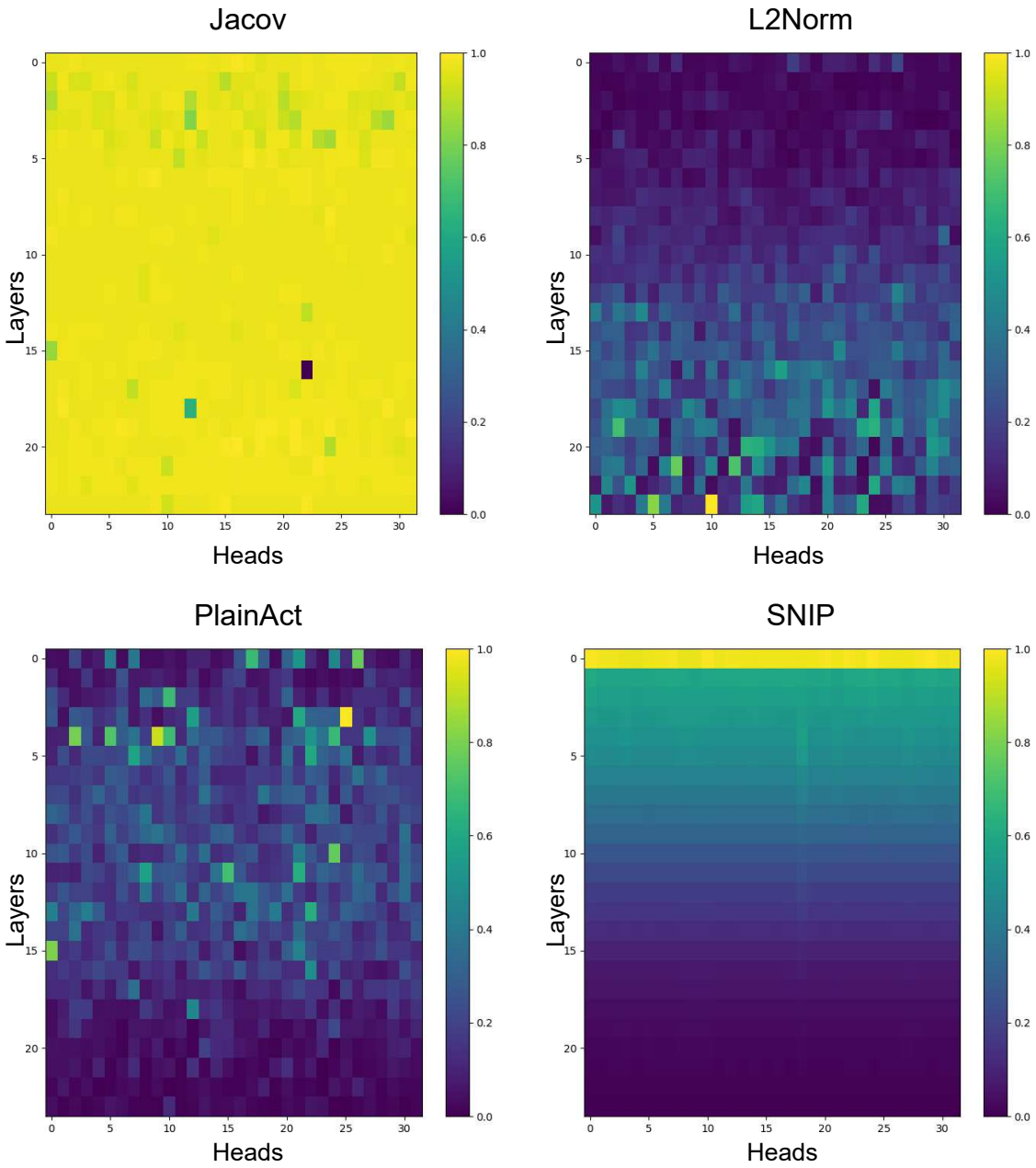

Figure 21: Aggregated pruning criteria scores per-head for the OPT-1.3B model, over the ARC-Easy training task in a five-shot setting.