A deeper look at depth pruning of LLMs

Shoaib Ahmed Siddiqui [1] **Xin Dong** [2] **Greg Heinrich** [2] **Thomas Breuel** [2] **Jan Kautz** [2] **David Krueger** [1]
Pavlo Molchanov [2]

Abstract

Large Language Models (LLMs) are not only resource-intensive to train but even more costly to deploy in production. Therefore, recent work has attempted to prune blocks of LLMs based on cheap proxies for estimating block importance, effectively removing 10% of blocks in well-trained LLaMa-2 and Mistral 7b models without any significant degradation in downstream metrics. This work explores different block importance metrics by considering adaptive metrics such as Shapley value in addition to static ones explored in prior work. We show that *adaptive metrics exhibit a trade-off in performance between tasks i.e., improvement on one task may degrade performance on the other due to differences in the computed block influences*. Furthermore, we extend this analysis from a complete block to individual self-attention and feed-forward layers, highlighting the propensity of the self-attention layers to be more amenable to pruning, even allowing ***removal of up to 33% of the self-attention layers without incurring any performance degradation on MMLU for Mistral 7b*** (significant reduction in costly maintenance of KV-cache). Finally, we look at simple performance recovery techniques to emulate the pruned layers by training lightweight additive bias or low-rank linear adapters. *Performance recovery using emulated updates avoids performance degradation for the initial blocks (up to 5% absolute improvement on MMLU)*, which is either competitive or superior to the learning-based technique[1].

1. Introduction

The utility of training language models with an increasing number of parameters is currently undisputed [19; 6; 31; 18].

[1]University of Cambridge, UK [2]NVIDIA Research, USA. Correspondence to: Shoaib Ahmed Siddiqui <msas3@cam.ac.uk>.

Work presented at TF2M workshop at ICML 2024, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]Code to reproduce experiments is publicly available: `https://github.com/shoaibahmed/llm_depth_pruning`.

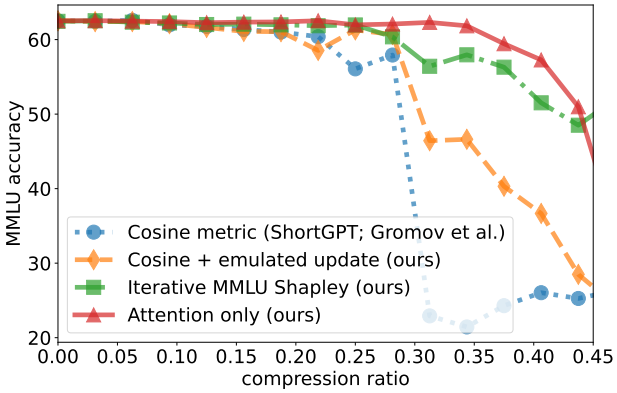

Figure 1: **Our self-attention pruning, adaptive metrics, as well as emulated updates in comparison to cosine block pruning** visualized w.r.t. compression ratios on Mistral 7b (see Fig. 5 for results on LLaMa-2 7b).

This has led to tremendous gains in performance, including the emergence of in-context learning [6]. However, aside from training, which is a one-time cost, deploying these models in production presents unique challenges, particularly high inference costs. Therefore, efficiency research for language models has gained significant popularity in the recent past [10; 2; 32; 22].

An extreme form of (structured) pruning is full-block pruning from a pretrained model. Men et al. [22]; Gromov et al. [12] recently showed that it is possible to completely drop blocks from a range of pretrained language models based on cheap proxies for block importance/influence such as computing the cosine distance between the input and the output representations for each block in a transformer.

This paper explores this direction further by analyzing the impact of different metrics on block identification, specifically focusing on adaptive metrics such as the one based on Shapley-value [29] in addition to static ones such as cosine block influence mainly used in prior work [22; 12]. We further look at the impact of individual self-attention and the feed-forward layers, which together form a block as analyzed by prior work. Finally, we evaluate the effectiveness of simple strategies for performance recovery, including a simple baseline of an additive bias (called 'emulated update') which is based on the empirical mean of the update applied by the block, as well as learning-based techniques

such as the training of a low-rank adapter, following prior literature on layer-stitching [3].

We visualize our main findings in Fig. 1 where 'block cosine' represents the method employed in prior work [22; 12]. Note that the x-axis represents the compression ratio computed based on the total number of layers. Therefore, it does not represent an equal number of parameters being pruned as a block comprises both a self-attention and a feed-forward layer. The figure highlights that the model can tolerate a higher fraction of pruning for self-attention layers in comparison to feed-forward layers or complete model blocks (computed using loss-shapley block influence – see Fig. 3 for a direct comparison). Pruning self-attention layers can provide a significant boost in model efficiency due to the costly maintenance of KV-cache at inference time. We further include an upper bound on MMLU performance by iteratively computing Shapley-value-based block influence directly on the MMLU test set to understand the possible improvements achievable when leveraging information about the task. Our main findings can be summarized as:

Finding 1. We highlight the impact of different block influence metrics on downstream model performance (Fig. 2). Our analysis reveals a trade-off in performance between tasks when evaluating adaptive metrics e.g., Shapley-value [29] (Fig. 8). Furthermore, we find that performance on some tasks degrades significantly, such as GSM-8k [9] even after pruning a single block from the model (Fig. 6 and Fig. 7).

Finding 2. We further dissect a block into attention and feed-forward layers to separately evaluate their influence, and highlight a higher propensity to prune attention layers in contrast to feed-forward layers while preserving MMLU accuracy (Fig. 3).

Finding 3. We evaluate two simple performance recovery techniques: (i) simple empirical mean of the block update, and (ii) training of a low-rank linear adapter in place of the missing block (Fig. 4). Surprisingly, our results show that applying a simple average block update is either competitive or superior in performance as compared to the learning-based approach potentially due to overfitting and catastrophic forgetting.

2. Related work

He et al. [14] popularized residual layers for training deep models. Veit et al. [35] hypothesized residual networks to be ensembles of smaller sub-networks, exhibiting layer dropping at inference time with minimal performance degradation. Veit & Belongie [34] extended this to dynamic input-conditioned layer-skipping. Vaswani et al. [33] introduced the famous transformer architecture by combining ideas of attention and residual networks to develop highly performant architectures. While there is a large body of work on pruning for efficient inference, we focus on depth pruning and refer the readers to Wan et al. [36] for a more comprehensive treatment of the literature.

Samragh et al. [28] initialized a subnetwork using blocks from a pretrained GPT-2 [25]. Sheared LLaMa [37] proposed an optimization-based view for simultaneous depth and width pruning of LLaMa-2 7b model [31]. Shortened LLaMa [20] showed that depth pruning is competitive against width-only pruning, or a sophisticated combination of both, while exploring several different metrics for estimating block influence. Men et al. [22] focused on LLaMa-2 [31] by using cosine distance (between activations before and after a block) as a proxy of block importance. Gromov et al. [12] also showed similar results on both LLaMa-2 [31] and Mistral [18], while also proposing a healing method by optimizing low-rank adapters [16] for the remaining blocks. This healing process is aimed at minimizing performance degradation with block pruning.

Jaiswal et al. [17] extended this by computing the cosine similarity of the representations at inference time and skipping only the feed-forward layers in the less important regions of the network (particularly in the middle of the network). In a similar spirit, Raposo et al. [26] trained a router to decide which blocks to skip i.e., reduce network depth (similar to the expert router in MoE [30]).

This paper attempts to take a deeper look at depth pruning of LLMs by looking at multiple metrics, datasets, block granularities (going to individual feed-forward and self-attention layers), and recovery techniques to establish the utility of each of these decisions on the resulting model.

3. Methods

3.1. Block influence metrics

We consider different block influence metrics including cosine that computes the angular distance between the input and output representations to a block [22; 12], relative L_1/L_2 that computes the norm of the update with respect to the norm of the input representation [28; 22], and Shapley-value-based [29] estimate which computes the marginal contribution of a block by computing the difference in performance between all subsets where a particular block is present and the block is absent. We focus on computing Shapley-value for the distance to the full model logits or the language modeling loss, except Fig. 1 where we compute it directly for the 0-1 loss on the MMLU test set. In this case, the Shapley computation is:

$$\text{SHAP}^l = \mathbb{E}_{\mathbf{x}, s \subseteq S \setminus \{l\}} \left[\mathcal{L}(O^{s \cup \{l\}}(\mathbf{x})) - \mathcal{L}(O^s(\mathbf{x})) \right]$$

where S represents the full set of blocks, s represents a subset of the blocks, and $O^s(\mathbf{x})$ represents the output of the

model on input $\mathbf{x}$ when using a subset of the blocks s.

3.2. Performance recovery techniques

As dropping blocks can induce a distribution shift in the representation, we consider two simple performance recovery techniques.

3.2.1. Emulated update

Emulated update is a particularly simple strategy that computes the average update applied by each block on a small calibration set, and applies that average additive update at inference time when a particular block is dropped. This can be viewed as having a 'bias' term instead of the full block.

3.2.2. Low-rank linear adapters

Following ideas from layer-stitching literature [3], we evaluate the efficacy of training low-rank linear adapters as a performance recovery measure. More precisely, we introduce two low-rank matrices per block to be learned (aside from the per-channel weightings learned as part of the normalization layer): This is similar in spirit to the healing process based on LoRA adapters by Gromov et al. [12], but we apply it in place of the missing blocks instead of applying it on the remaining blocks.

We consider three different ways of training these linear adapters: (i) minimize MSE between the output of the block and the low-rank adapter, (ii) use supervised fine-tuning (SFT) on the final output, and (iii) use logit-distillation on the final model output. We use the calibration set to train this low-rank adapter independently one block at a time.

4. Experiments

We use a subset of $150k$ sequences (context window of 2048 tokens) with cross-document concatenation from OpenWebText [1] as our full calibration set, and another $15k$ sequences as our validation set. We report 5-shot MMLU [15] accuracy as commonly reported in the literature [22; 12]. We use LLaMa-2 7b [31] and Mistral 7b [18] for our evaluations. We focus on 7b models primarily due to them being a sweet spot between compute cost and model quality. Larger models are more amenable to block pruning, as already established in prior work [22; 12]. Therefore, we expect our findings to be equally applicable to larger models.

4.1. Block influence metrics

We compare different block influence metrics in Fig. 2 (see Fig. 8 for more detailed results), where we apply min-max normalization on the block influences for visualization. As cosine distance-based block influence matches the proposed approach in [22; 12], our results also closely match theirs

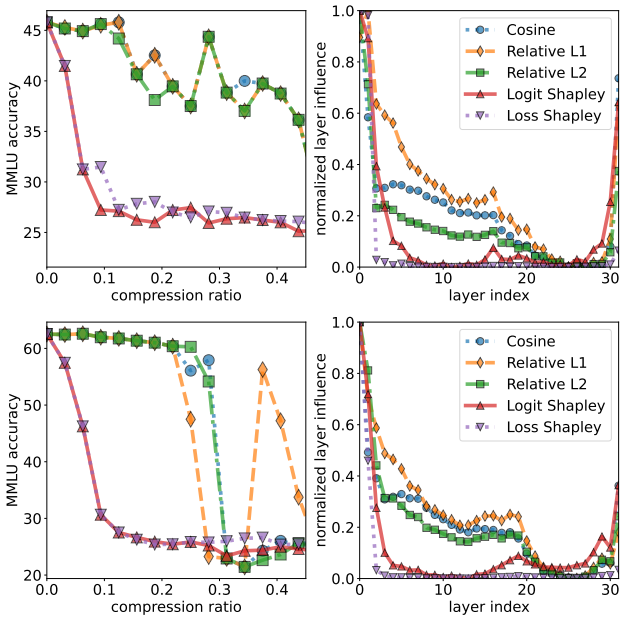

Figure 2: **Comparison of different block influence metrics** used for block pruning and their impact on downstream performance in terms of MMLU accuracy on LLaMa-2 7b (top) and Mistral 7b (bottom).

in terms of robustness against block pruning. Furthermore, the blocks that are assigned the least importance are located towards the end of the network i.e., the second half. Relative L_p norm metrics also closely follow this trend, and hence, achieve robustness comparable to that of cosine distance.

A significant deviation however is shown by the Shapley-value-based estimation, which is an adaptive metric, providing significant gains in terms of reduction of the average loss (visualized in Fig. 8) as it is specifically computed to reduce the model's language modeling loss. However, this results in a drastic reduction in accuracy when evaluating MMLU. Block influence plots show that in contrast to all other metrics that primarily focus on the second half of the network for removal, Shapley instead focuses on removing blocks from the first half of the network. When computing loss-based Shapley-value directly on the MMLU test set, we see that performance is significantly better than the cosine baseline (see Fig. 1), highlighting that task-specific (Shapley-based) block dropping might retain higher performance at the same number of dropped blocks as compared to task-agnostic techniques but at the expense of sacrificing more performance on tasks not directly considered.

We also visualize the results on other tasks from OpenLLM leaderboard [4] using lm-eval-harness [11] in Fig. 6 and Fig. 7. These results highlight that despite just a relatively minor impact on performance for MMLU, some metrics are significantly more impeded (such as GSM-8k). The proxy used in this case i.e., 'cosine', matches closely the right influence values to optimize MMLU, which is not aligned

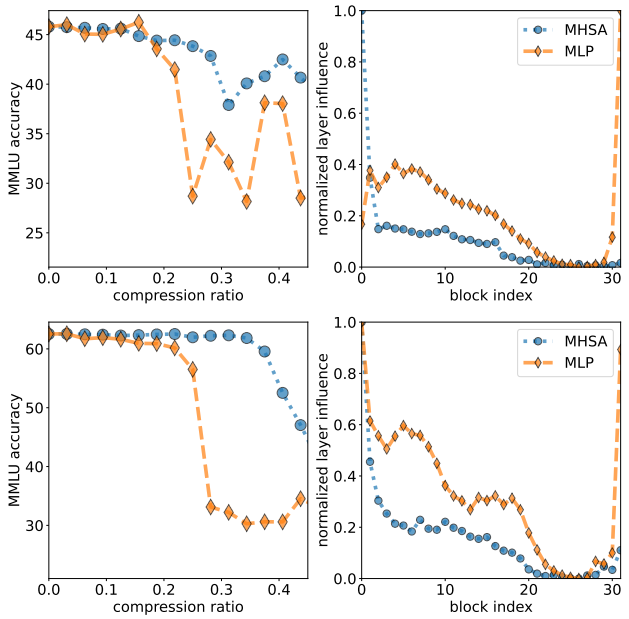

Figure 3: **Comparison of pruning the self-attention and feed-forward layers of a network using cosine metric** on LLaMa-2 7b (top) and Mistral 7b (bottom), highlighting a higher propensity for the self-attention layers to be pruned.

with the importance ideal for other datasets. On the other hand, adaptive metrics can be optimized individually for each task at hand.

4.2. Disentangling the impact of individual layers

As each block in a transformer is composed of self-attention as well as a feed-forward network, one can apply the same influence techniques to understand the impact of performance when pruning individual layers rather than complete blocks. The results are highlighted in Fig. 3 (see Fig. 9 and Fig. 10 for complete results where we also visualize the results with joint layer ranking).

We see that models exhibit higher resilience against the dropping of self-attention layers in contrast to feed-forward layers. In contrast to all other experiments, loss shapley in the case of self-attention layers achieves competitive influence as compared to other approaches, highlighting that it might be easier to estimate in contrast to feed-forward layers (Fig. 9 and Fig. 10). Furthermore, Mistral 7b exhibits a higher propensity for self-attention layer removal compared to LLaMa-2 7b.

Although most transformer parameters are in the feed-forward layers, pruning the self-attention layer is equally useful due to its quadratic dependency on sequence length and the complex management of key-value caches, which introduces significant latency [38; 23]. Developing techniques that combine block and layer pruning, potentially pruning only a single layer in some blocks, is an exciting direction for future research.

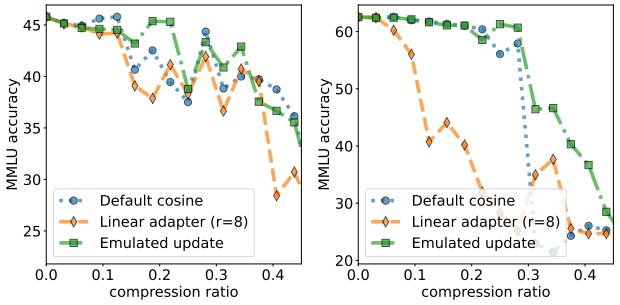

Figure 4: **Impact on using performance recovery techniques with cosine block influence** on LLaMa-2 7b (left) and Mistral 7b (right). The linear adapter was trained using logit-distillation with a rank of 8.

4.3. Performance recovery techniques

We trained all our adapters for 800 steps with an effective batch size of 8, where the training loss plateaued. The results for emulated update and linear adapter with a rank of 8 trained using logit-distillation are shown in Fig. 4 (see Appendix B for full results with different ranks and training strategies). We observe a clear improvement for both LLaMa-2 and Mistral when using emulated update where performance improves significantly at some stages ($\geq 5\%$ on MMLU). However, performance with a linear adapter is either comparable (LLaMa-2 7b) or worse (Mistral 7b) than the performance improvement observed by the simple emulated block update. This can be partially attributed to the overfitting of the adapter on the training corpus, which is not reflective of the true pertaining distribution. Gromov et al. [12] observed a similar performance degradation when trying to do parameter-efficient fine-tuning after dropping blocks from Mistral as compared to LLaMa-2.

These results indicate minor differences introduced by later blocks can be recovered using simple techniques. However, this mitigation technique is unable to change the tipping point in performance.

5. Conclusion

This paper explores depth pruning of LLMs, highlighting performance differences with various influence techniques, including adaptive ones. Block-sensitivity metrics like Shapley improve perplexity but degrade performance on tasks like MMLU, showing block relevance tension. Our results indicate self-attention layers are more amenable to pruning than feed-forward layers, with performance unaffected by pruning many self-attention layers. We also address misalignment from block pruning using basic performance recovery techniques. Our simplest baseline, using an average update, matches or outperforms low-rank adapters with different training formulations.

References

[1] OpenWebText corpus. https://github.com/jcpeterson/openwebtext, 2019. Accessed: 2024-04-16.

[2] Ashkboos, S., Croci, M. L., Nascimento, M. G. d., Hoefler, T., and Hensman, J. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*, 2024.

[3] Bansal, Y., Nakkiran, P., and Barak, B. Revisiting model stitching to compare neural representations. *Advances in neural information processing systems*, 34: 225–236, 2021.

[4] Beeching, E., Fourrier, C., Habib, N., Han, S., Lambert, N., Rajani, N., Sanseviero, O., Tunstall, L., and Wolf, T. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

[5] Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

[6] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.

[8] Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457, 2018.

[9] Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[10] Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[11] Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. A framework for few-shot language model evaluation, 12 2023. URL https://zenodo.org/records/10256836.

[12] Gromov, A., Tirumala, K., Shapourian, H., Glorioso, P., and Roberts, D. A. The unreasonable ineffectiveness of the deeper layers. *arXiv preprint arXiv:2403.17887*, 2024.

[13] Hartvigsen, T., Gabriel, S., Palangi, H., Sap, M., Ray, D., and Kamar, E. Toxigen: A large-scale machine-generated dataset for implicit and adversarial hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

[14] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[15] Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[16] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[17] Jaiswal, A., Hu, B., Yin, L., Ro, Y., Liu, S., Chen, T., and Akella, A. Ffn-skipllm: A hidden gem for autoregressive decoding with adaptive feed forward skipping. *arXiv preprint arXiv:2404.03865*, 2024.

[18] Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[19] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[20] Kim, B.-K., Kim, G., Kim, T.-H., Castells, T., Choi, S., Shin, J., and Song, H.-K. Shortened llama: A simple depth pruning for large language models. *arXiv preprint arXiv:2402.02834*, 2024.

[21] Lin, S., Hilton, J., and Evans, O. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May

2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229.

[22] Men, X., Xu, M., Zhang, Q., Wang, B., Lin, H., Lu, Y., Han, X., and Chen, W. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*, 2024.

[23] Nawrot, P., Łańcucki, A., Chochowski, M., Tarjan, D., and Ponti, E. M. Dynamic memory compression: Retrofitting llms for accelerated inference. *arXiv preprint arXiv:2403.09636*, 2024.

[24] Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q. N., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

[25] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[26] Raposo, D., Ritter, S., Richards, B., Lillicrap, T., Humphreys, P. C., and Santoro, A. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024.

[27] Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

[28] Samragh, M., Farajtabar, M., Mehta, S., Vemulapalli, R., Faghri, F., Naik, D., Tuzel, O., and Rastegari, M. Weight subcloning: direct initialization of transformers using larger pretrained ones. *arXiv preprint arXiv:2312.09299*, 2023.

[29] Shapley, L. S. et al. A value for n-person games. 1953.

[30] Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[31] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[32] van der Ouderaa, T. F., Nagel, M., Van Baalen, M., Asano, Y. M., and Blankevoort, T. The llm surgeon. *arXiv preprint arXiv:2312.17244*, 2023.

[33] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[34] Veit, A. and Belongie, S. Convolutional networks with adaptive inference graphs. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–18, 2018.

[35] Veit, A., Wilber, M. J., and Belongie, S. Residual networks behave like ensembles of relatively shallow networks. *Advances in neural information processing systems*, 29, 2016.

[36] Wan, Z., Wang, X., Liu, C., Alam, S., Zheng, Y., Qu, Z., Yan, S., Zhu, Y., Zhang, Q., Chowdhury, M., et al. Efficient large language models: A survey. *arXiv preprint arXiv:2312.03863*, 1, 2023.

[37] Xia, M., Gao, T., Zeng, Z., and Chen, D. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

[38] Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

[39] Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

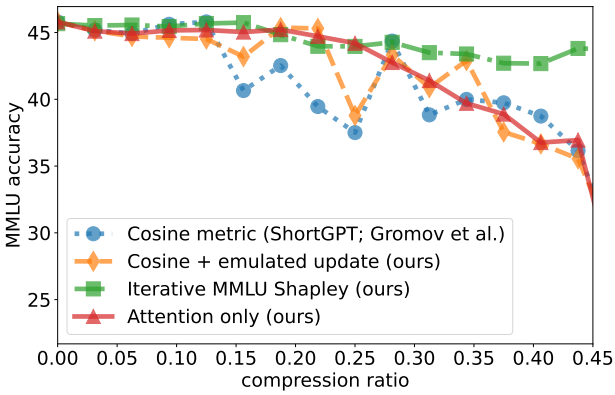

Figure 5: **Our self-attention pruning, adaptive metrics, as well as emulated updates in comparison to cosine block pruning** visualized w.r.t. compression ratios on LLaMa-2 7b (see Fig. 1 for results on Mistral 7b).

A. Results on other datasets

We include tasks from OpenLLM leaderboard [4] evaluated using lm-eval-harness package for reproducibility [11]. In particular, we evaluate performance on MMLU [15], GSM-8k [9], ARC (easy as well as challenging) [8], BoolQ [7], HellaSwag [39], Lambada [24], PiQA [5], Toxigen [13], TruthfulQA (MC2) [21], and Winogrande [27].

Fig. 6 presents our results on LLaMa-2 7b [31] while Fig. 7 presents the results on Mistral 7b [18]. We only plot the results for cosine block influence and loss shapley (computed on OpenWebText language modeling task). The results highlight that the model suffers from performance degradation on some tasks such as GSM-8k [9] and ARC challenge [8] even after pruning just a single block from the model. This indicates that just looking at MMLU provides only a partial picture of the true impact of depth pruning on model performance.

B. Low-rank linear adapters.

Results for LLaMa-2 7b and Mistral 7b with an adapter rank of 8 on a relative scale are visualized in Fig. 13 and Fig. 14 respectively. Results for LLaMa-2 7b and Mistral 7b with an adapter rank of 8 on the original scale are visualized in Fig. 15 and Fig. 16 respectively.

Results for LLaMa-2 7b and Mistral 7b with an adapter rank of 32 on a relative scale are visualized in Fig. 17 and Fig. 18 respectively. Results for LLaMa-2 7b and Mistral 7b with an adapter rank of 32 on the original scale are visualized in Fig. 19 and Fig. 20 respectively.

Results for LLaMa-2 7b and Mistral 7b with an adapter rank of 256 on a relative scale are visualized in Fig. 21 and Fig. 22 respectively. Results for LLaMa-2 7b and Mistral 7b with an adapter rank of 256 on the original scale are visualized in Fig. 23 and Fig. 24 respectively.

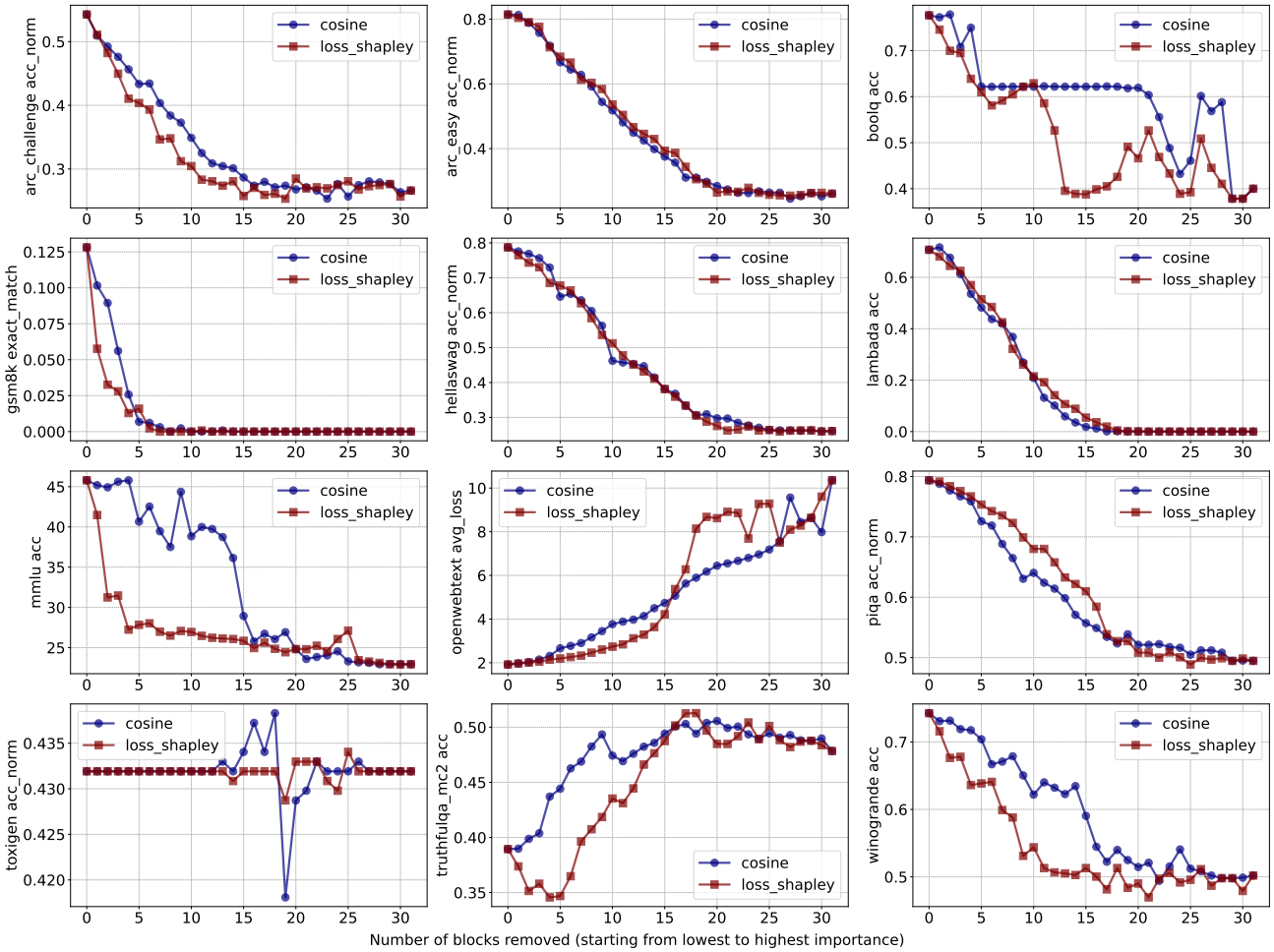

Figure 6: **LLaMa-2 7b evaluation on multiple datasets from eval-harness [11]**, highlighting the impact of pruning on some tasks is more significant (such as GSM-8k [9]) as compared to others.

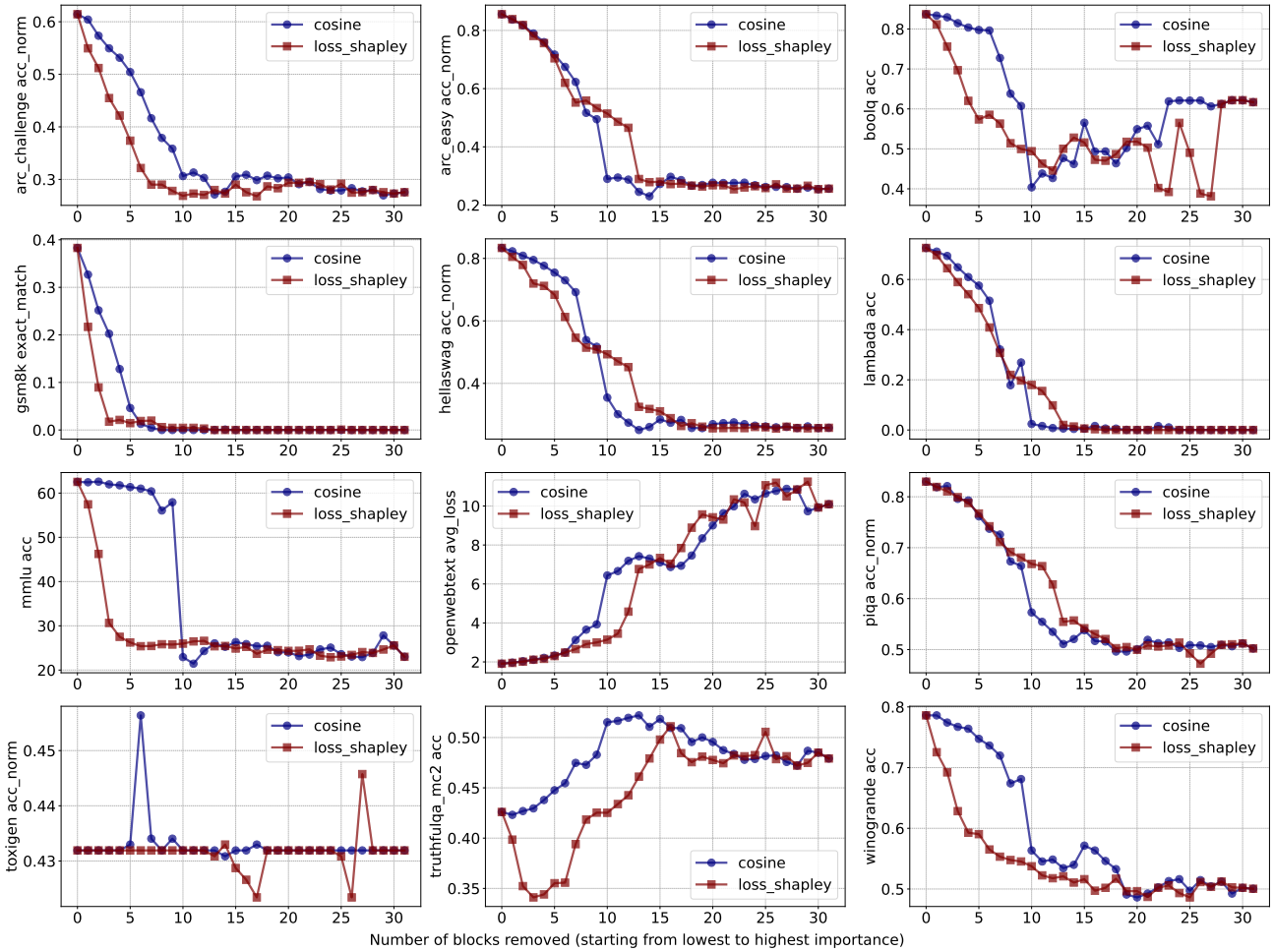

Figure 7: **Mistral-7b evaluation on multiple datasets from eval-harness [11]**, highlighting the impact of pruning on some tasks is more significant (such as GSM-8k [9]) as compared to others.

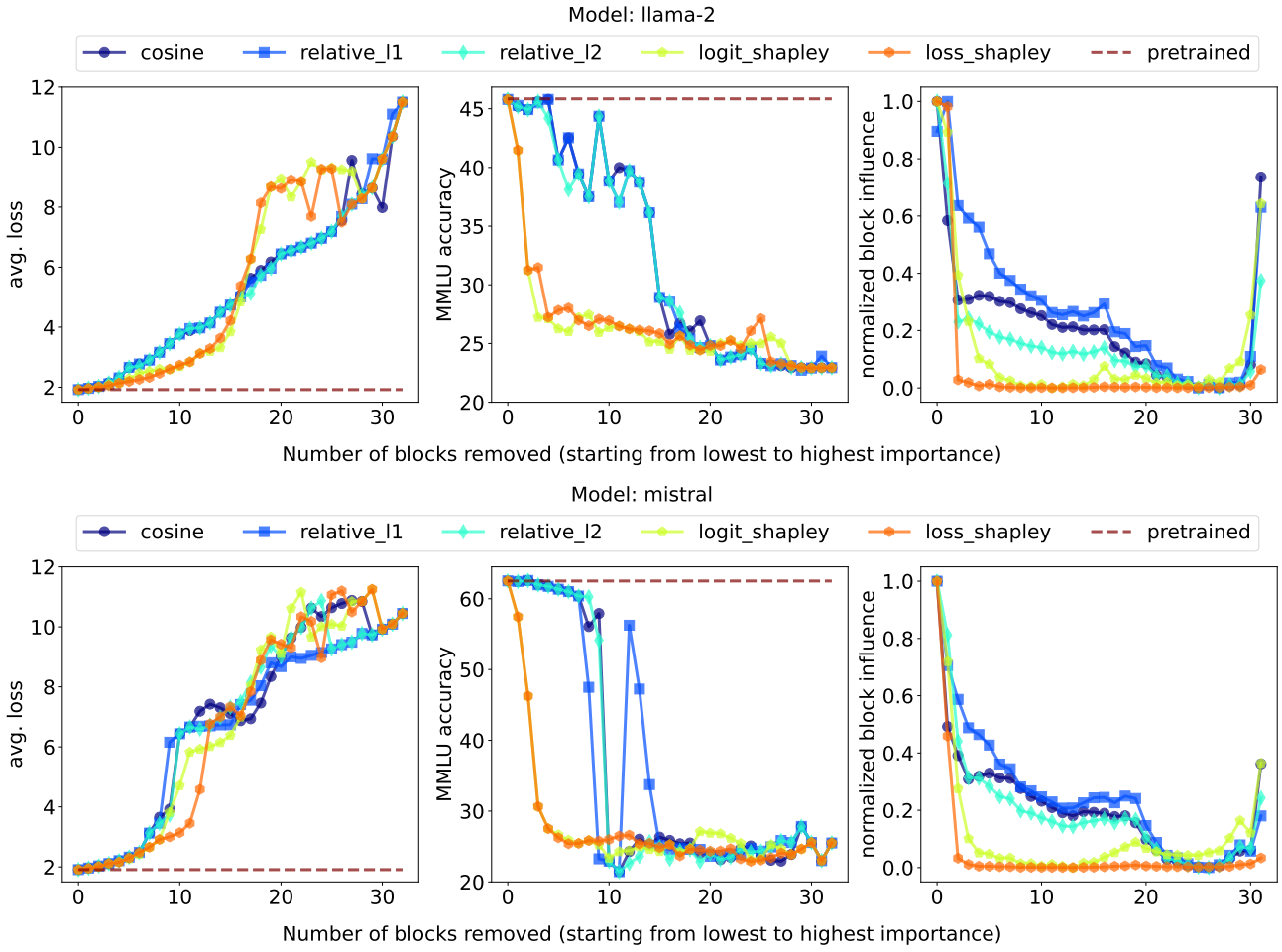

Figure 8: **Comparison of different block influence metrics** (detailed in Section 3.1) used for block pruning and their impact on downstream performance on LLaMa-2 7b (top) and Mistral 7b (bottom) in terms of average loss on a small curated validation set from OpenWebText and accuracy on MMLU.

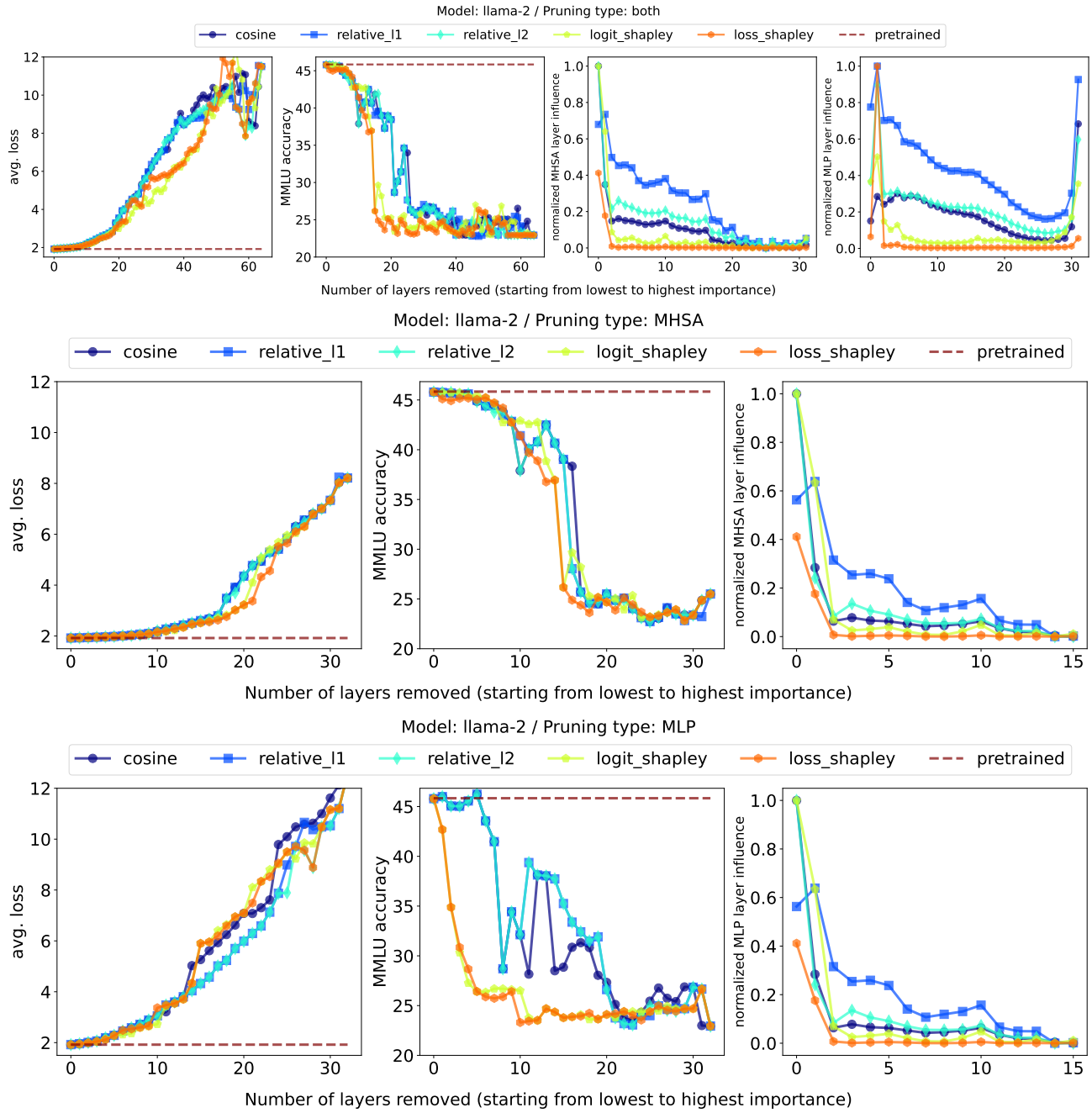

Figure 9: **Comparison of different layer influence metrics on LLaMa-2 7b** (detailed in Section 3.1) and their impact on downstream performance in terms of average loss on a small curated validation set from OpenWebText and accuracy on MMLU. The first row highlights joint pruning, while the second and third row highlights pruning of just self-attention or feed-forward layer respectively.

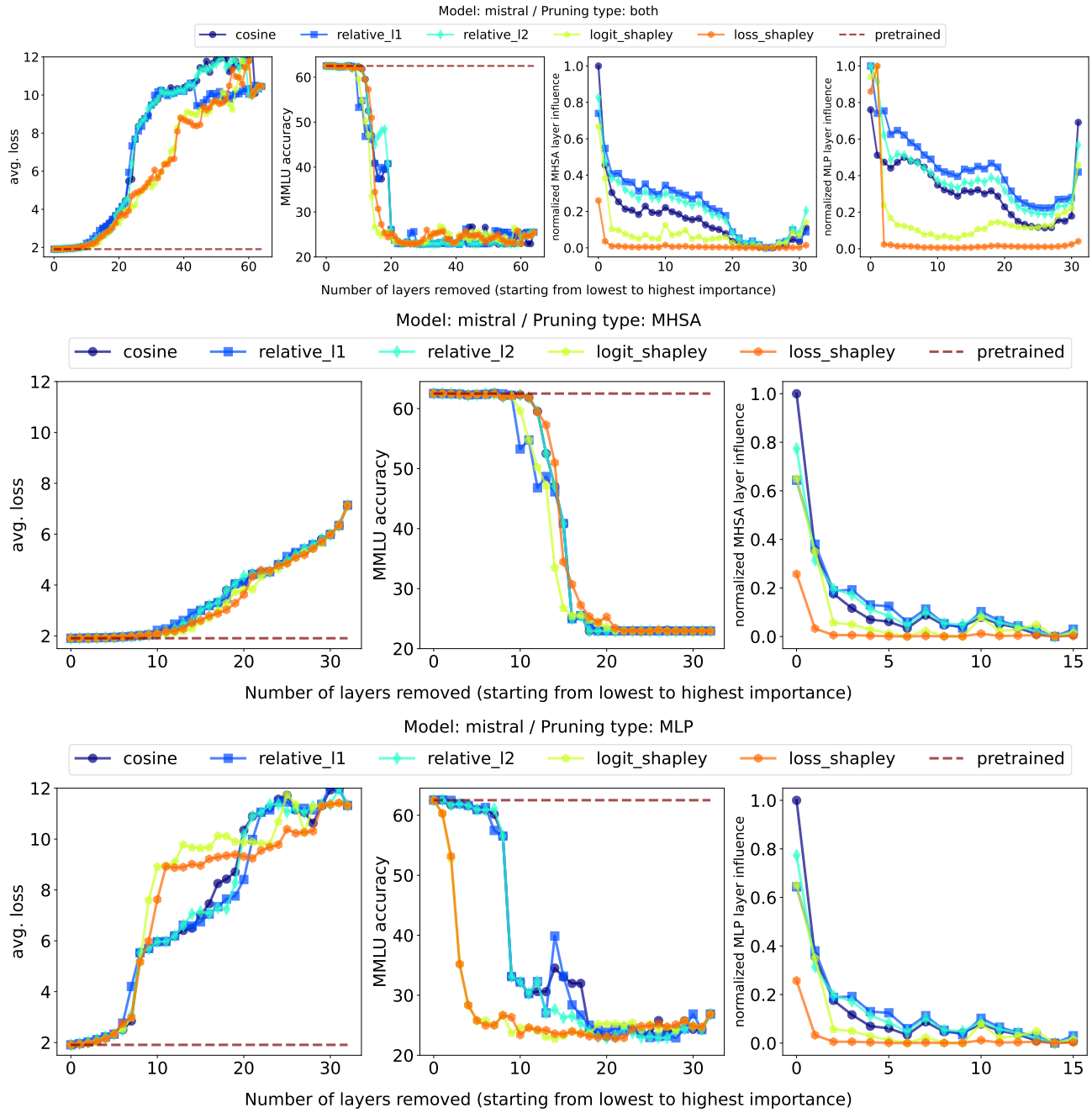

Figure 10: **Comparison of different layer influence metrics on Mistral 7b** (detailed in Section 3.1) and their impact on downstream performance in terms of average loss on a small curated validation set from OpenWebText and accuracy on MMLU. The first row highlights joint pruning, while the second and third row highlights pruning of just self-attention or feed-forward layer respectively.

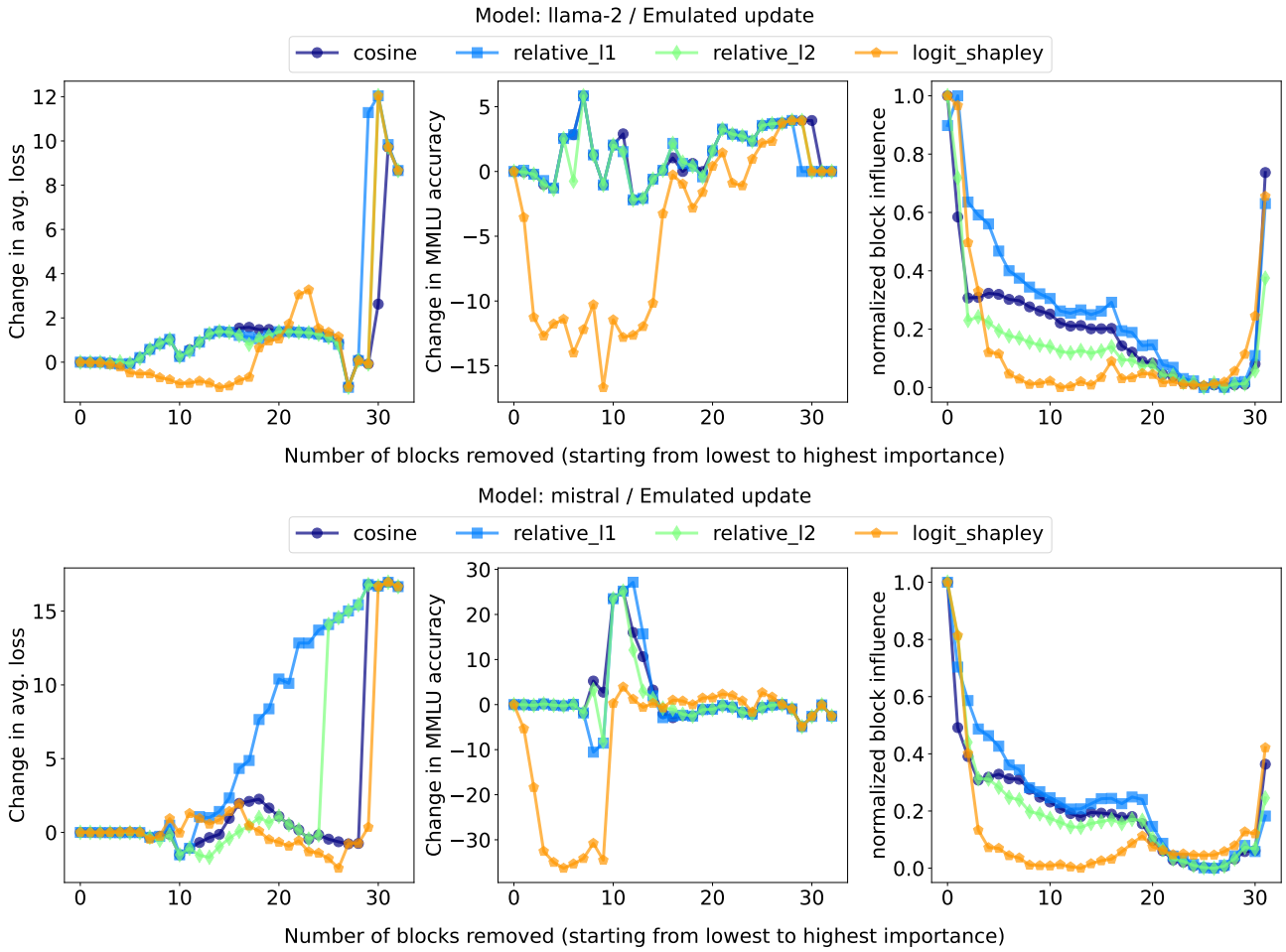

Figure 11: **Relative impact on performance with emulated update as a performance recovery method.** The plot with the original values is presented in Fig. 12.

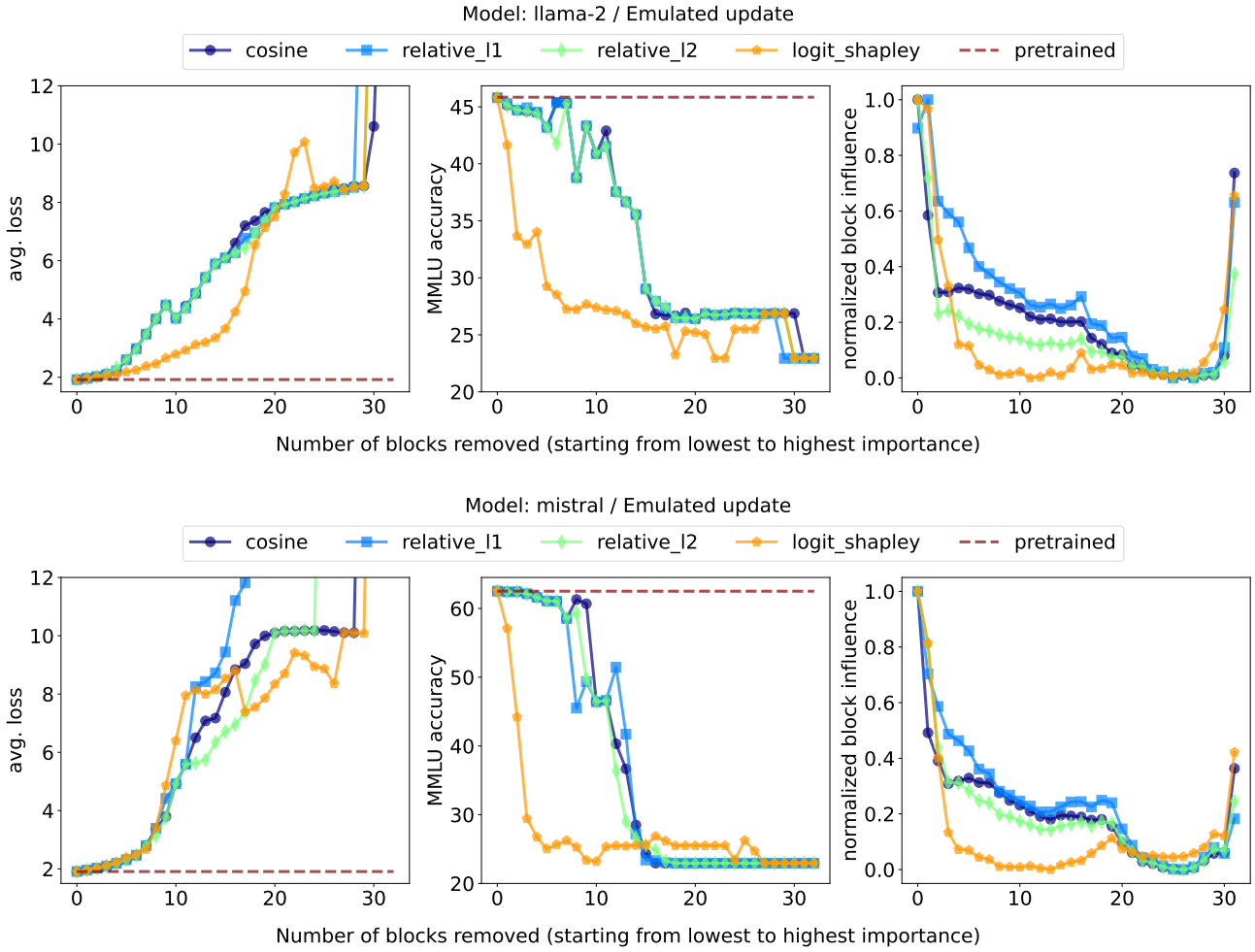

Figure 12: **Impact on performance with emulated update as a performance recovery method.**

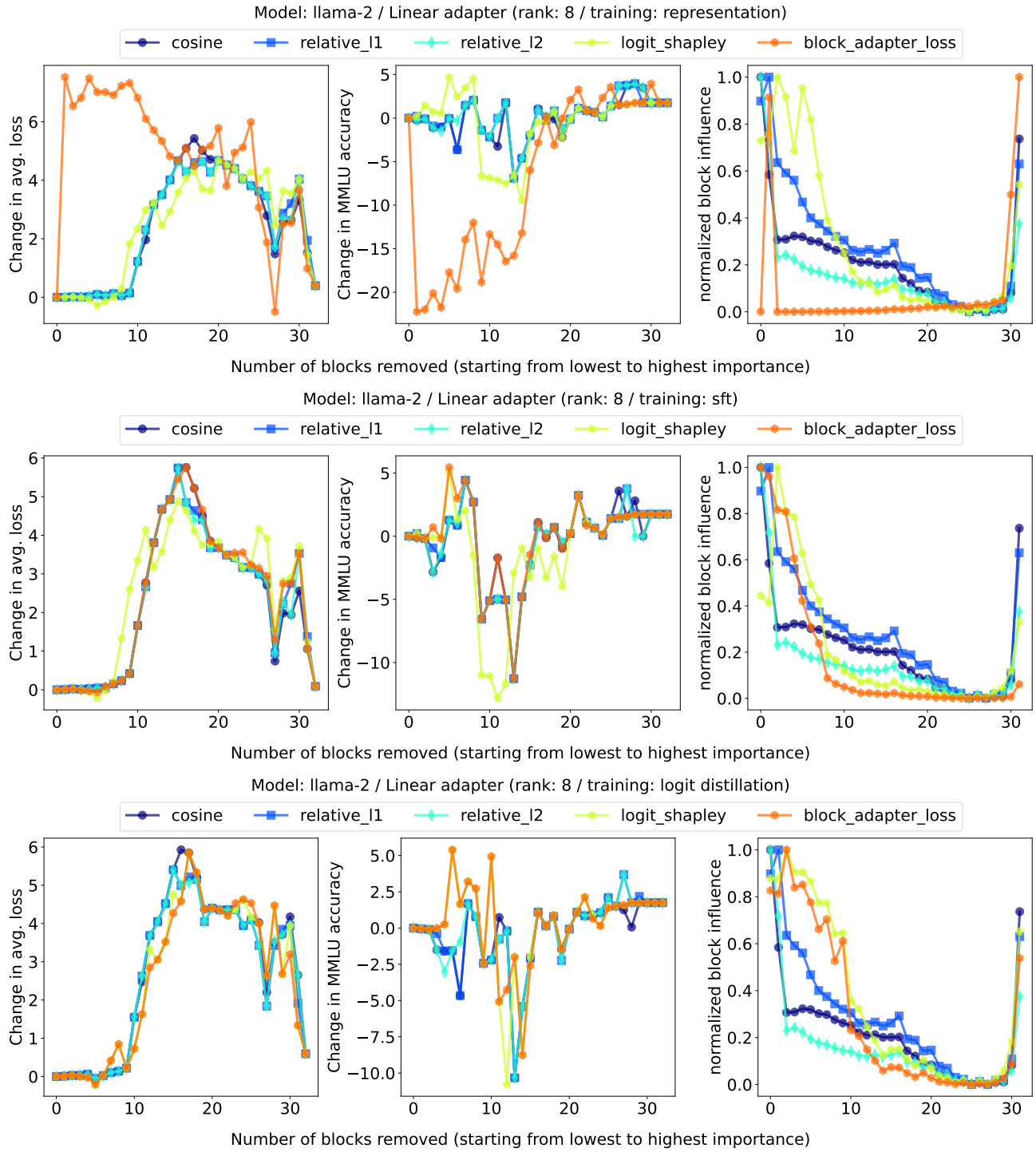

Figure 13: **Evaluating the relative impact of linear adapters for LLaMa-2 7b with a rank of 8** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model. The plot with the original values is presented in Fig. 15.

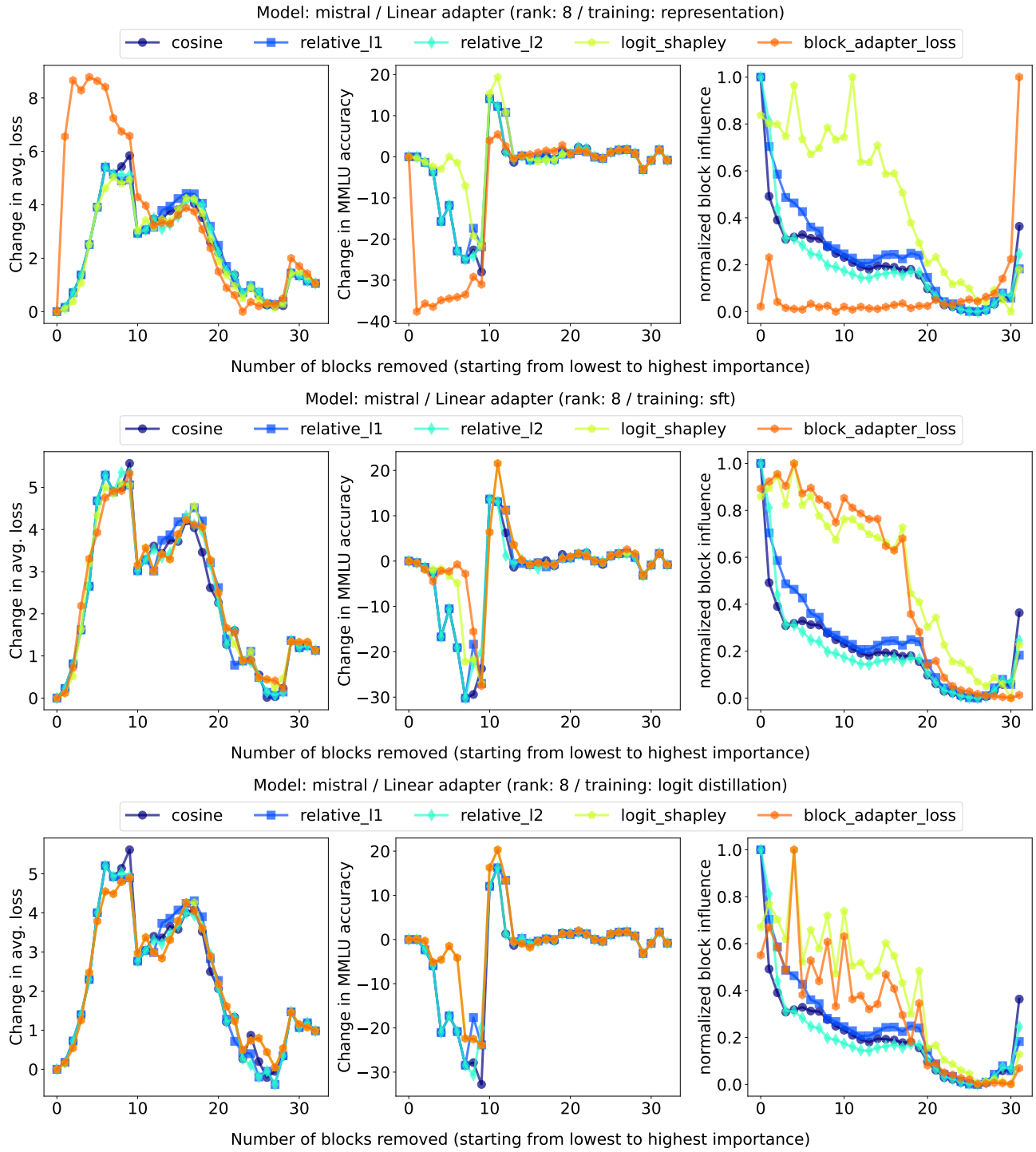

Figure 14: **Evaluating the relative impact of linear adapters for Mistral 7b with a rank of 8** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model. The plot with the original values is presented in Fig. 16.

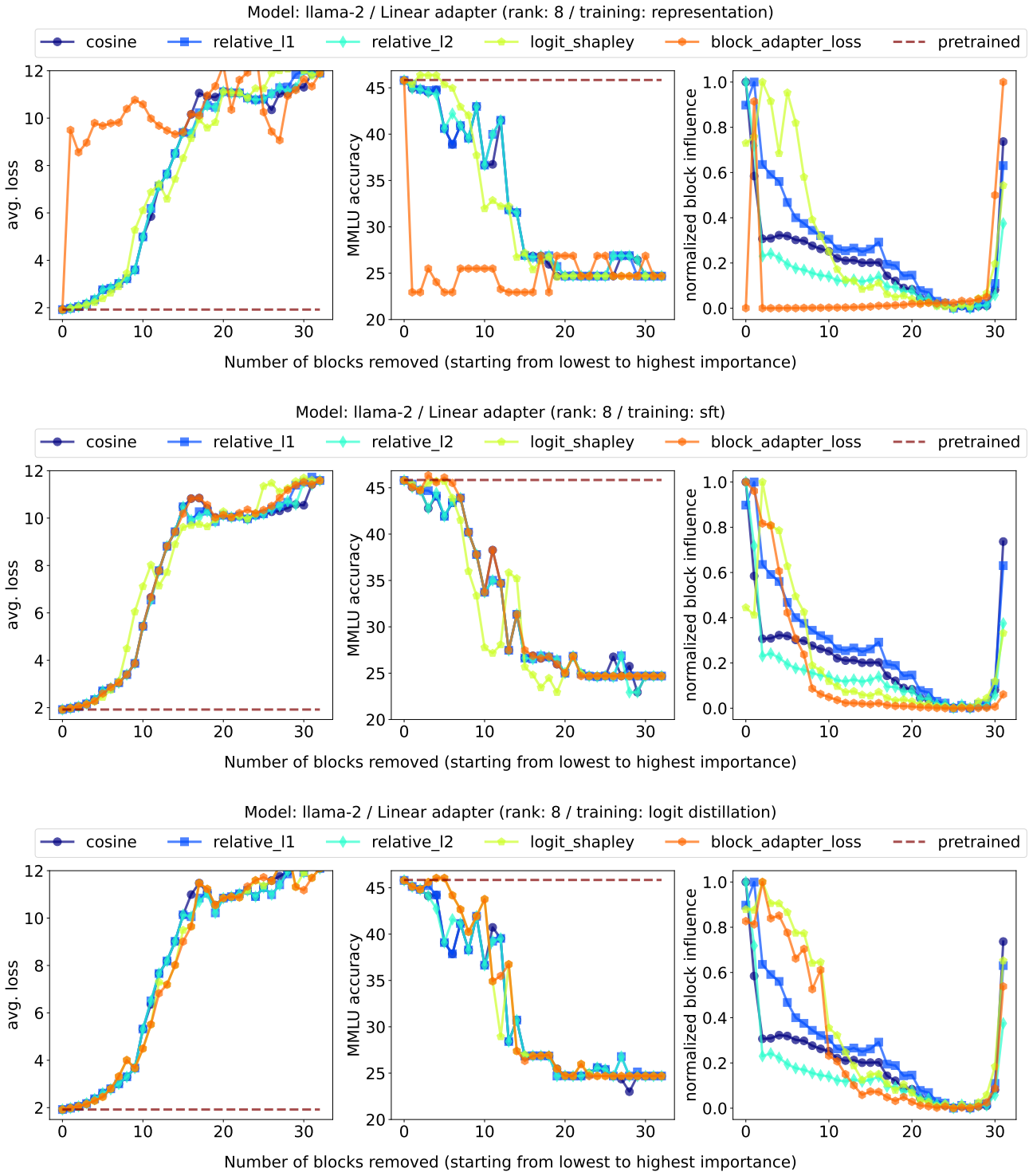

Figure 15: **Evaluating the impact of linear adapters for LLaMa-2 7b with a rank of 8** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model.

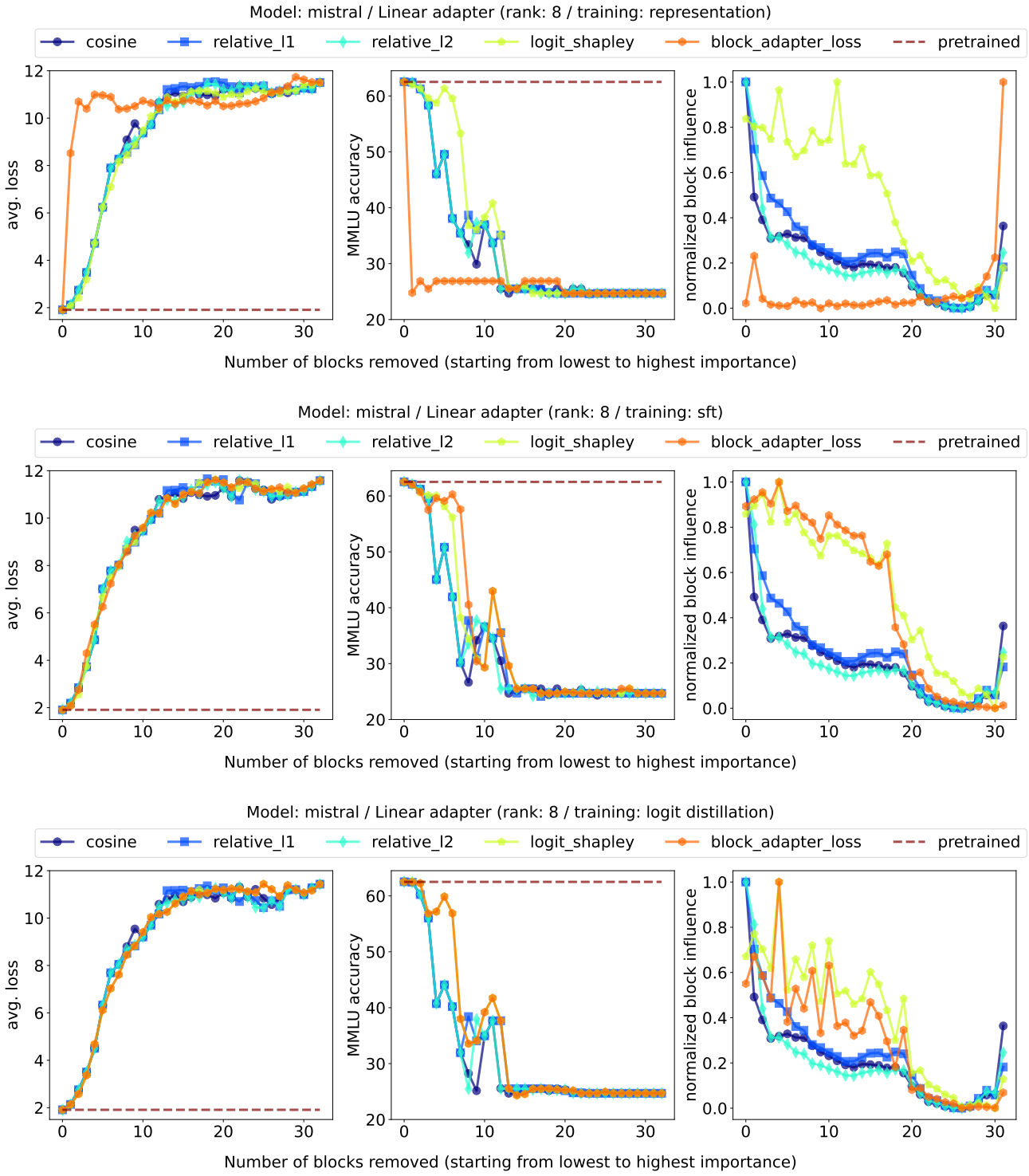

Figure 16: **Evaluating the impact of linear adapters for Mistral 7b with a rank of 8** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model.

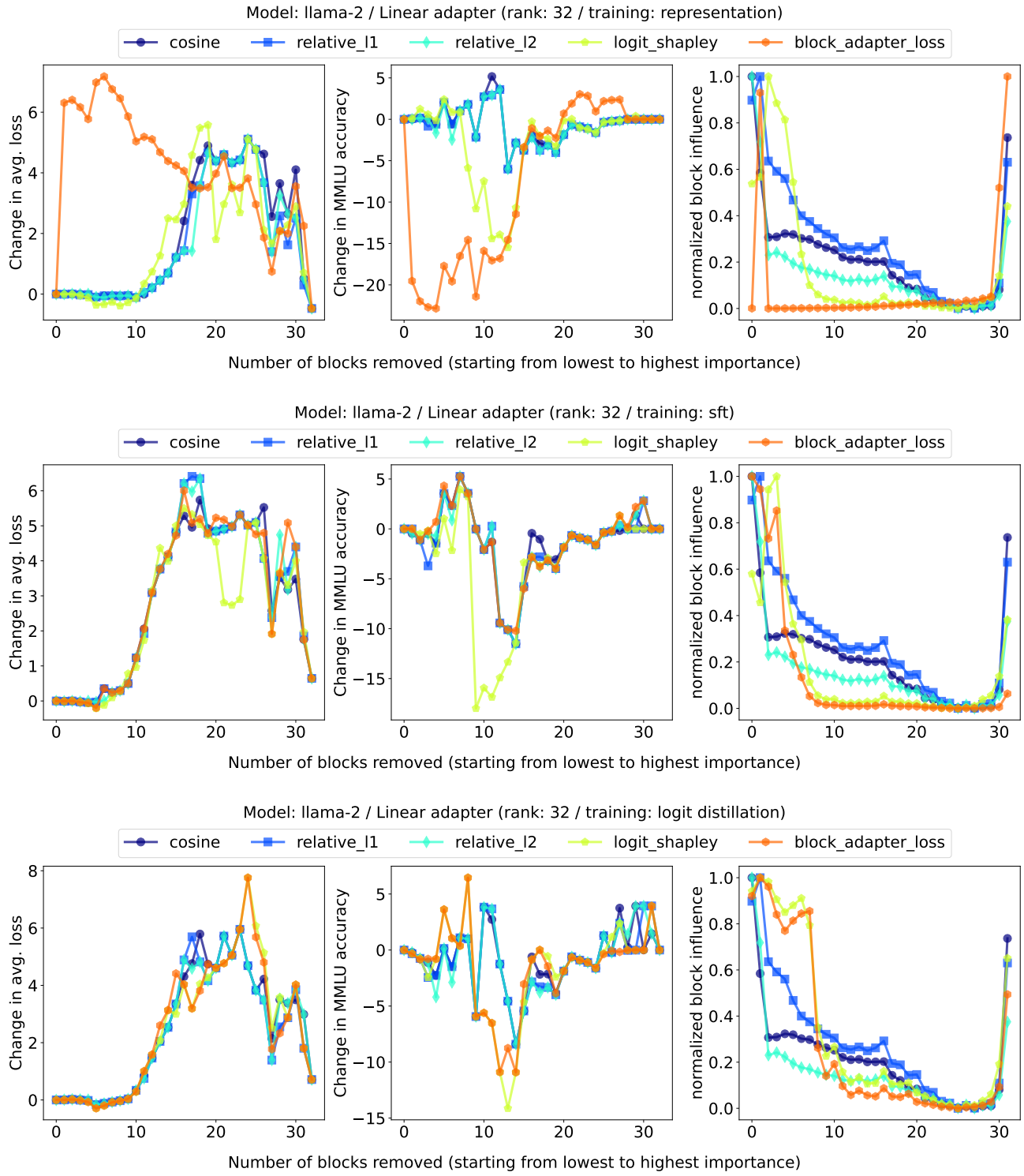

Figure 17: **Evaluating the relative impact of linear adapters for LLaMa-2 7b with a rank of 32** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model. The plot with the original values is presented in Fig. 19.

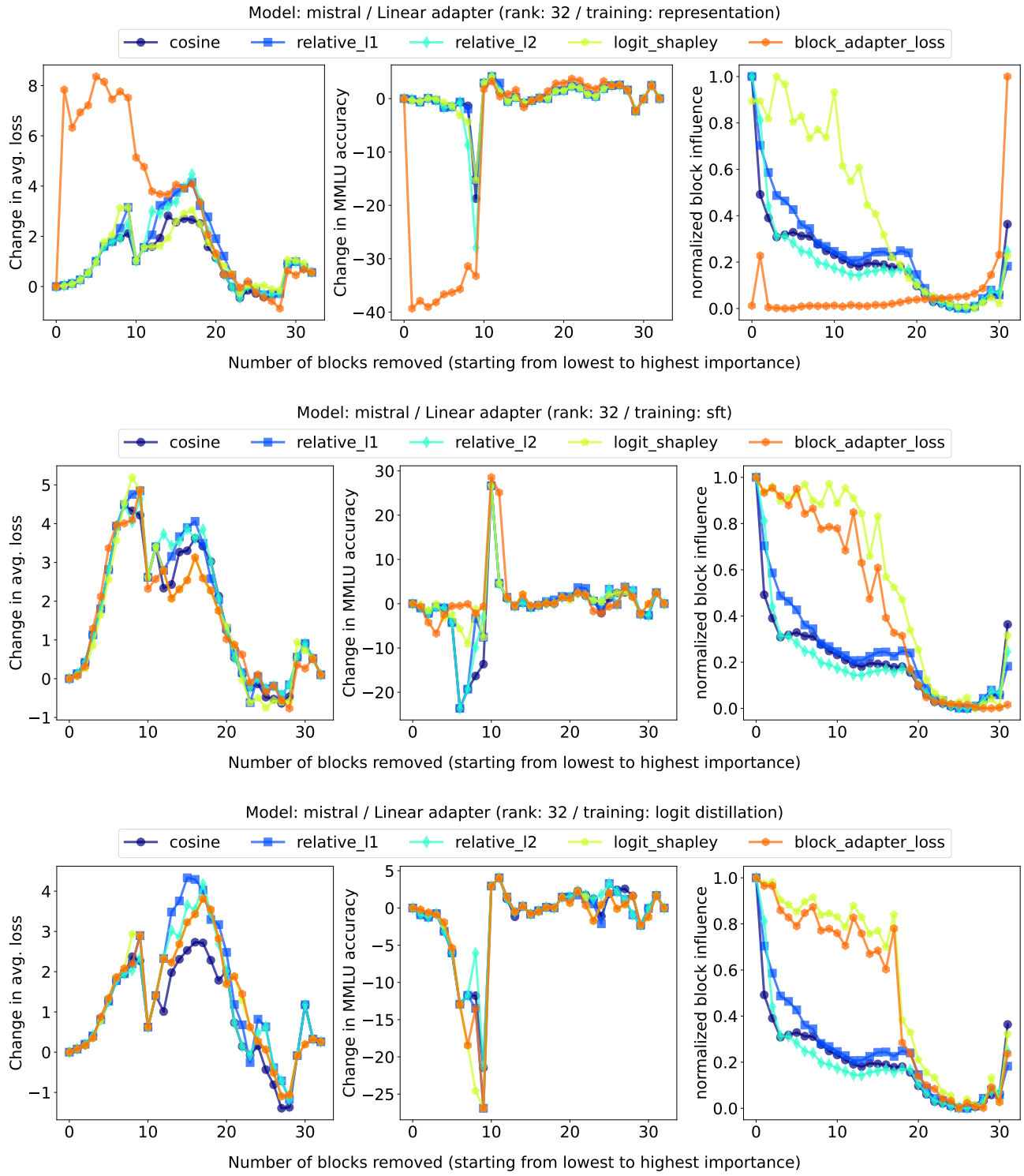

Figure 18: **Evaluating the relative impact of linear adapters for Mistral 7b with a rank of 32** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model. The plot with the original values is presented in Fig. 20.

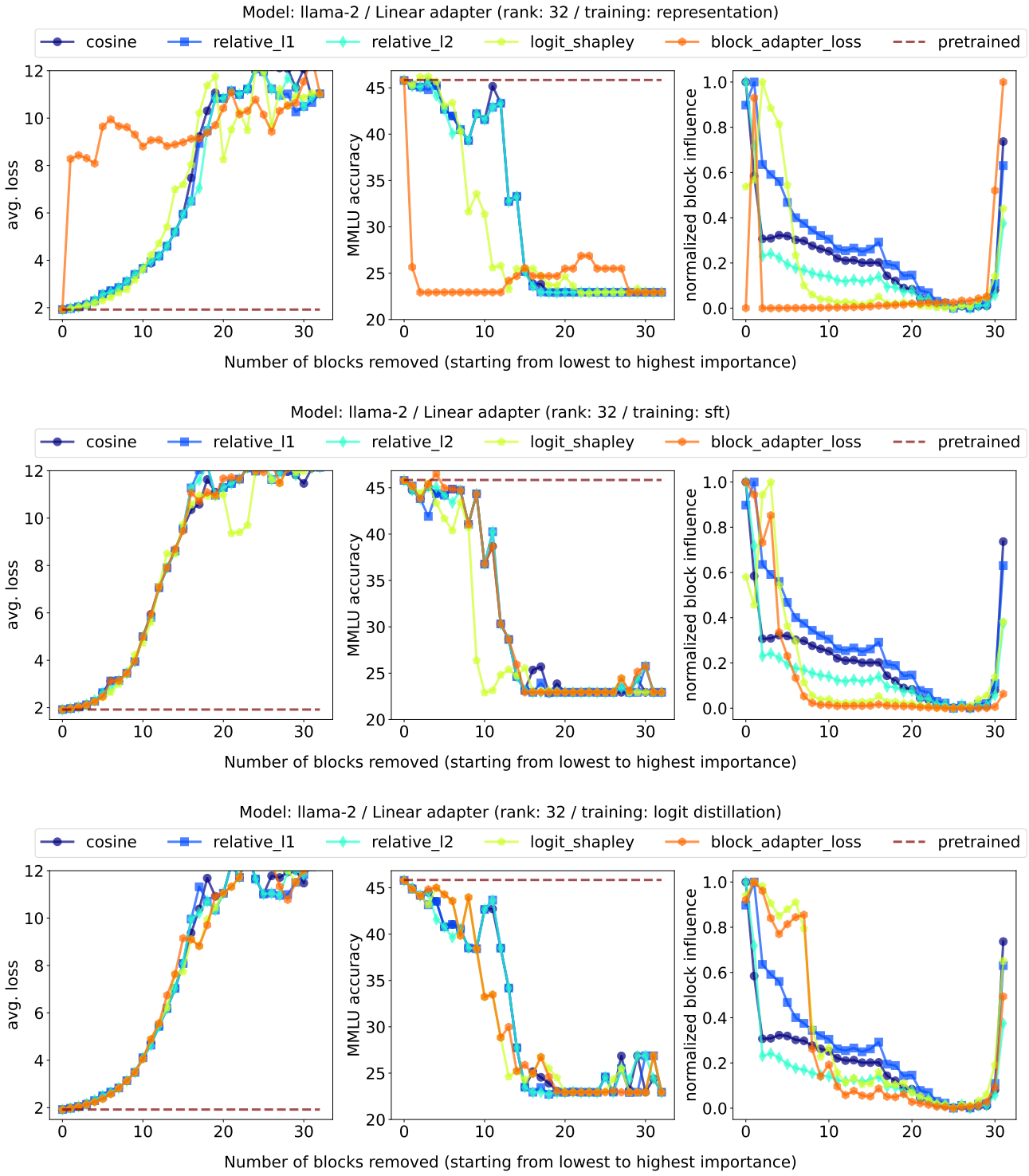

Figure 19: **Evaluating the impact of linear adapters for LLaMa-2 7b with a rank of 32** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model.

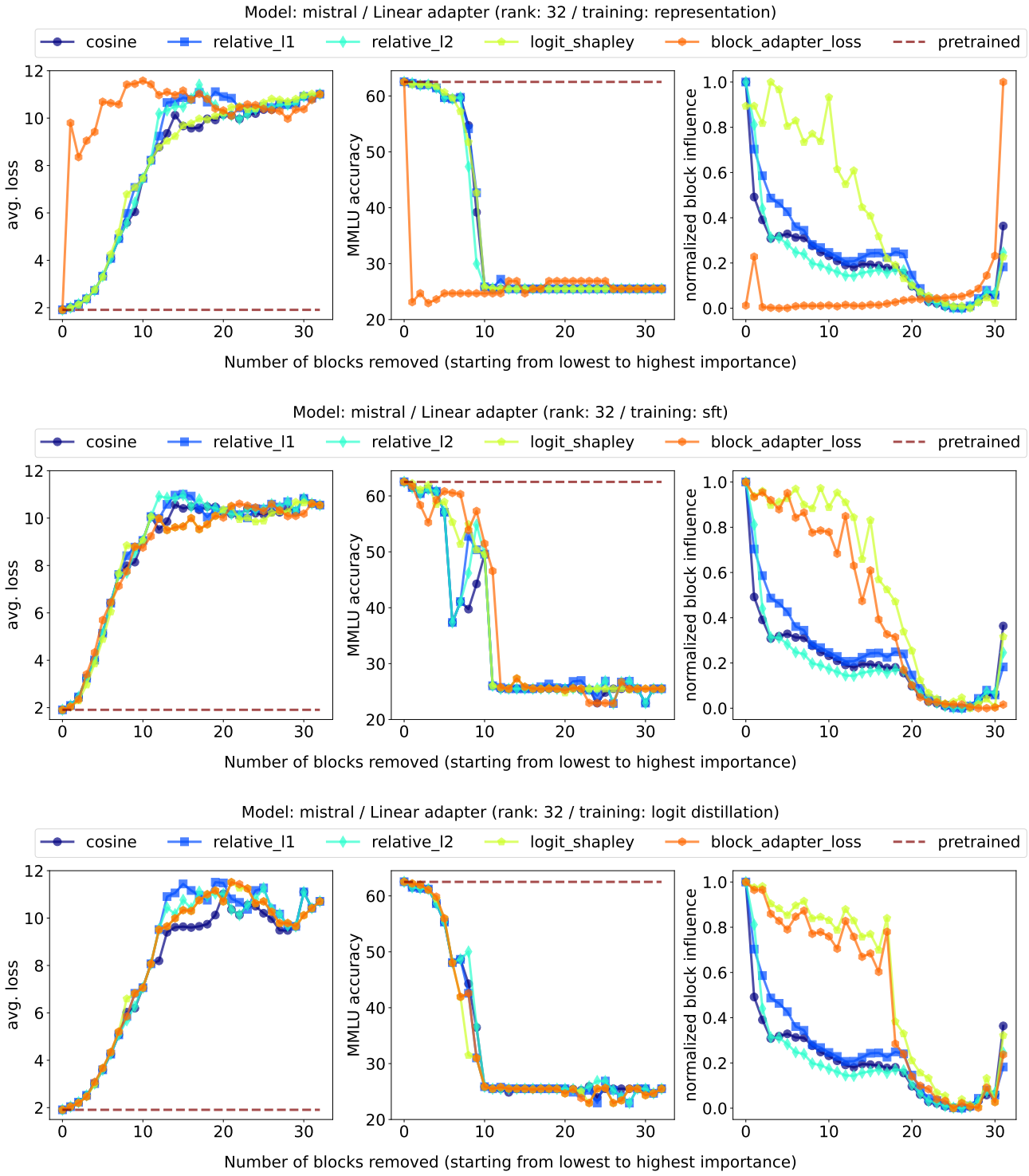

Figure 20: **Evaluating the impact of linear adapters for Mistral 7b with a rank of 32** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model.

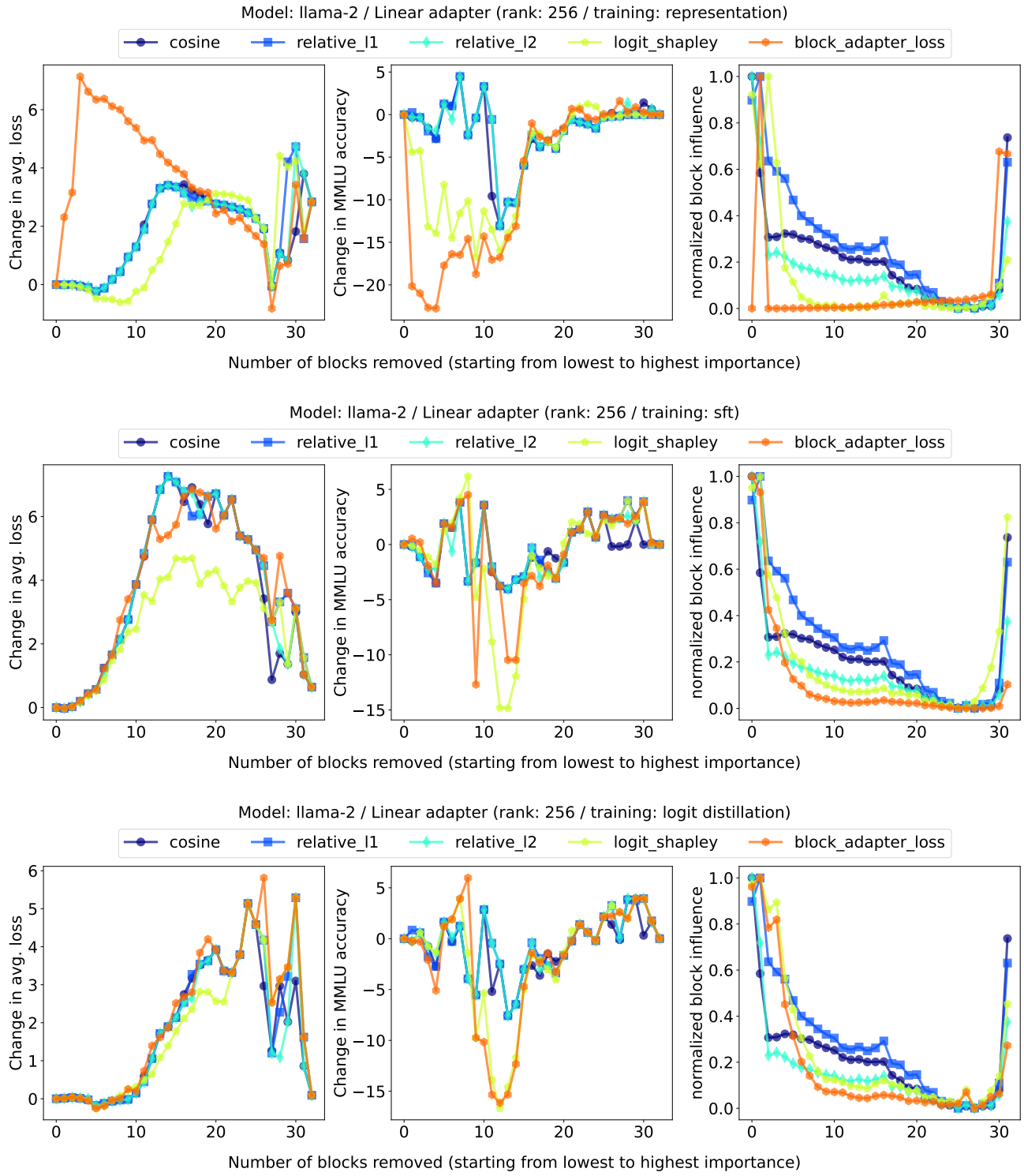

Figure 21: **Evaluating the relative impact of linear adapters for LLaMa-2 7b with a rank of 256** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model. The plot with the original values is presented in Fig. 23.

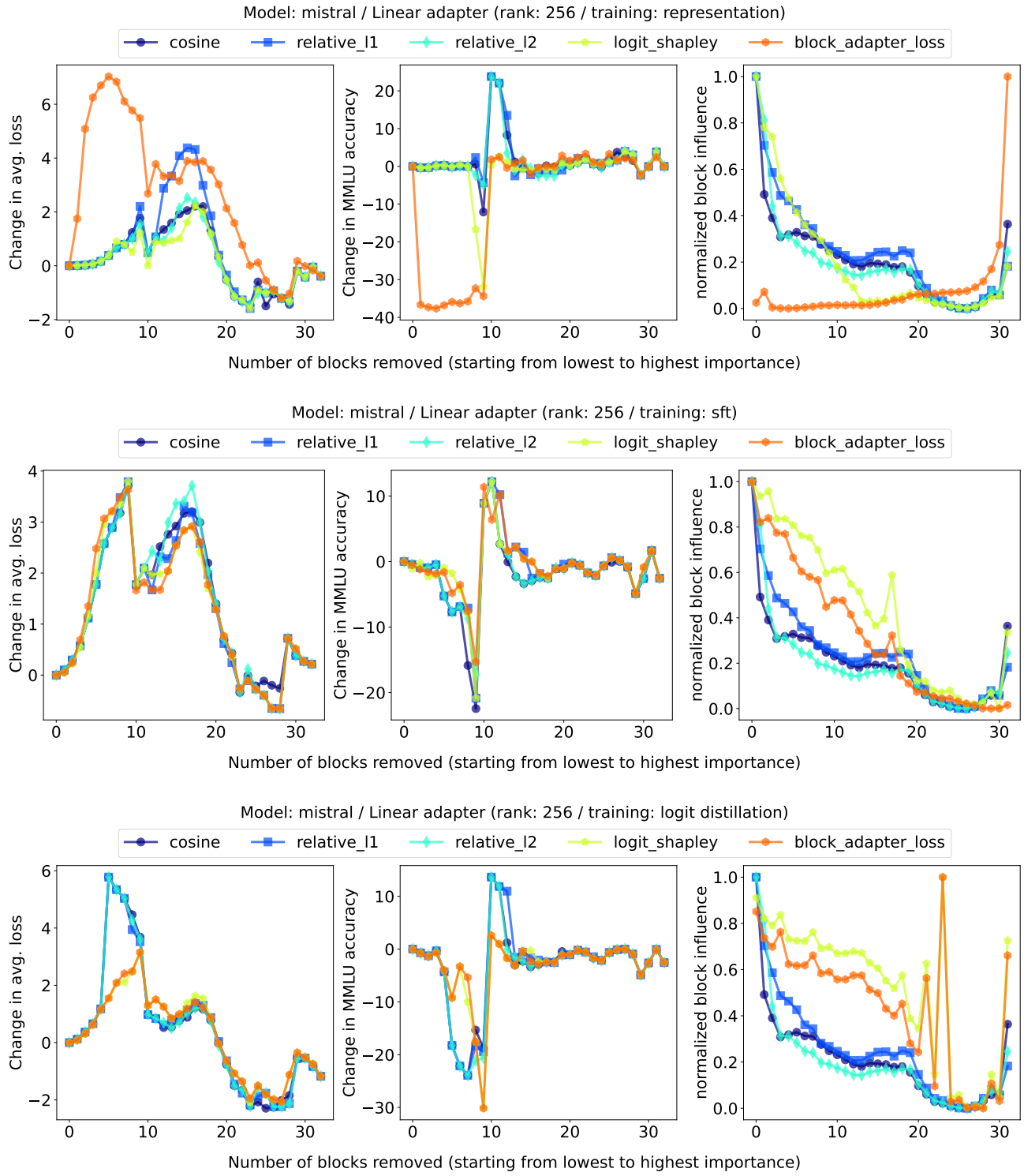

Figure 22: **Evaluating the relative impact of linear adapters for Mistral 7b with a rank of 256** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model. The plot with the original values is presented in Fig. 24.

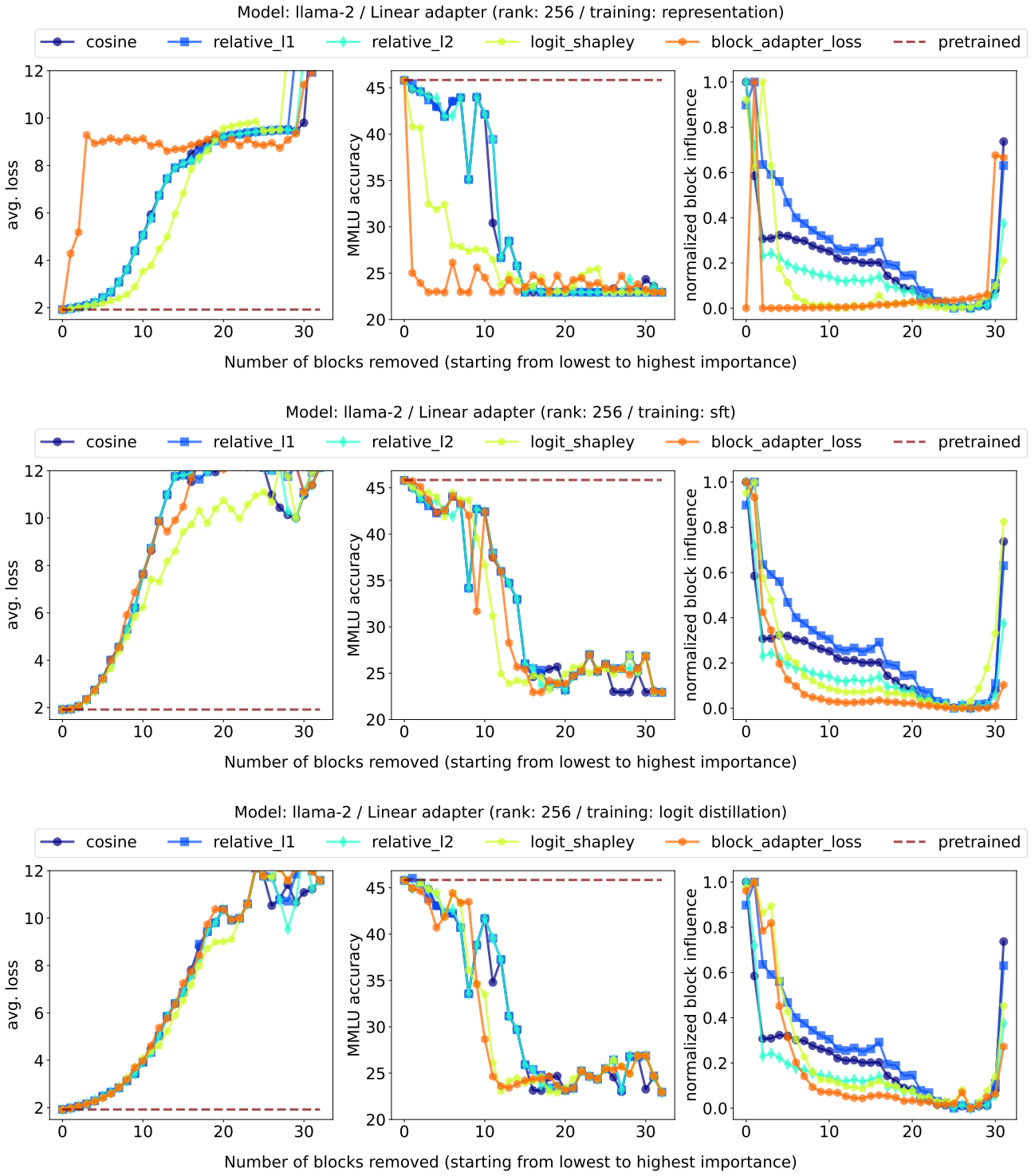

Figure 23: **Evaluating the impact of linear adapters for LLaMa-2 7b with a rank of 256** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model.

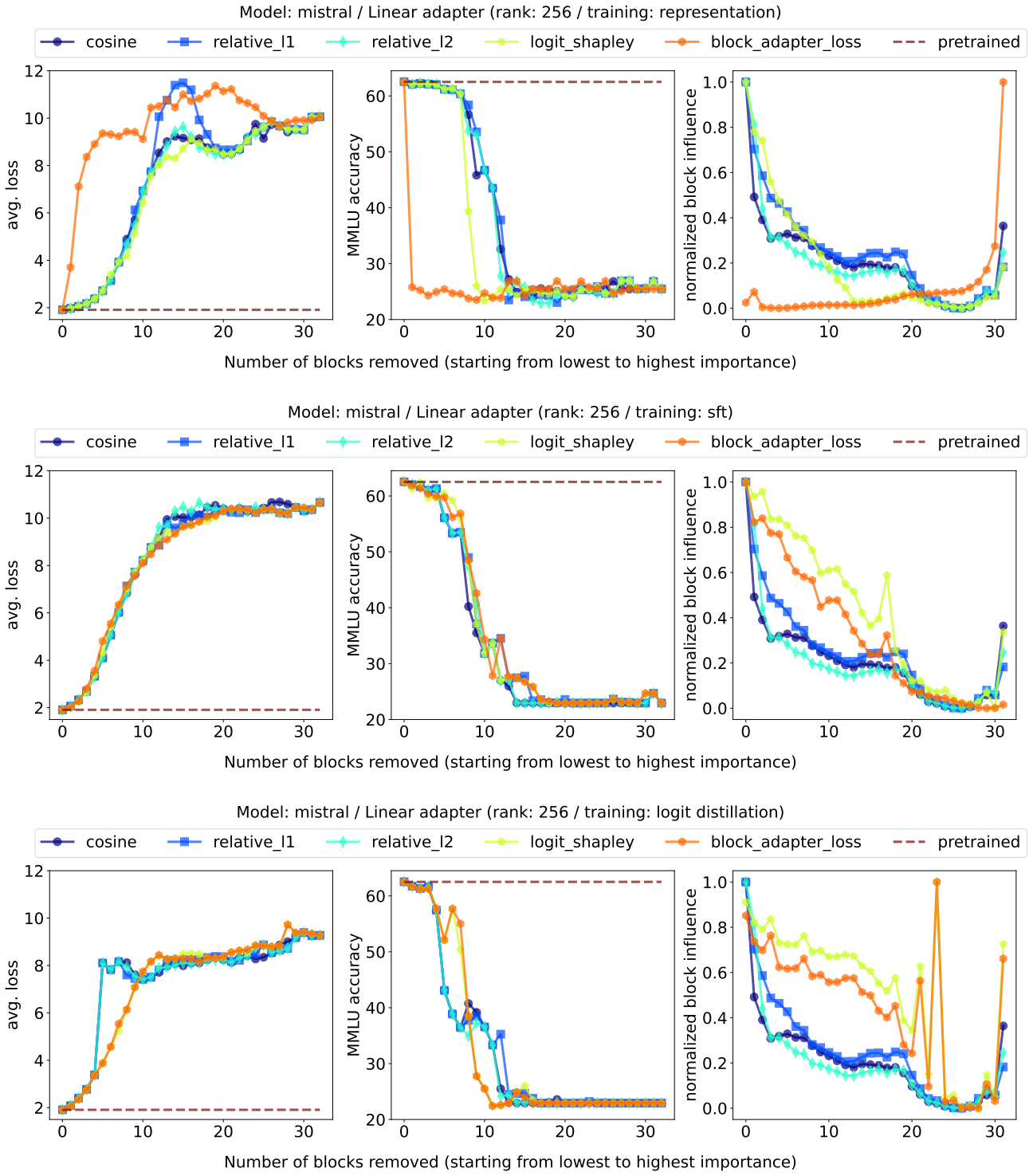

Figure 24: **Evaluating the impact of linear adapters for Mistral 7b with a rank of 256** trained using three different metrics including (a) MSE loss defined on the representation, (b) supervised fine-tuning (SFT), and (c) logit distillation where logits are distilled from the full model.