Iterative Structured Knowledge Distillation: Optimizing Language Models Through Layer-by-Layer Distillation

Malthe Have Musaeus
IT University of Copenhagen
malthe@musaeus.dk

Rob van der Goot
IT University of Copenhagen
robv@itu.dk

Abstract

Traditional language model compression techniques, like knowledge distillation, require a fixed architecture, limiting flexibility, while structured pruning methods often fail to preserve performance. This paper introduces Iterative Structured Knowledge Distillation (ISKD), which integrates knowledge distillation and structured pruning by progressively replacing transformer blocks with smaller, efficient versions during training. This study validates ISKD on two transformer-based language models: GPT-2 and Phi-1. ISKD outperforms L1 pruning and achieves similar performance to knowledge distillation while offering greater flexibility. ISKD reduces model parameters - 30.68% for GPT-2 and 30.16% for Phi-1 - while maintaining at least four-fifths of performance on both language modeling and commonsense reasoning tasks. These findings suggest that this method offers a promising balance between model efficiency and accuracy.

1 Introduction

Transformer-based language models, such as GPTs (Radford et al., 2018), Llama (Touvron et al., 2023), and Phi (Gunasekar et al., 2023), have become essential in natural language processing (Rahali and Akhloufi, 2023). These models, with parameter counts ranging from millions to billions, achieve high accuracy and versatility across various applications (Islam et al., 2023). However, their large size leads to scalability and deployment challenges due to high computational demands (Keles et al., 2023).

Efficient transformer model inference has been explored through various methods (Tay et al., 2022), with this paper focusing on pruning and knowledge distillation. Pruning removes less important weights, with structured pruning targeting entire layers (Ma et al., 2023), while unstructured pruning targets individual weights. Knowledge distillation trains a smaller "student" model to replicate the behavior of a larger "teacher" model (Sanh et al., 2019). Pruning reduces model size by removing weights from pre-trained models, while knowledge distillation transfers knowledge from larger to smaller models, often at the cost of performance. Pruning can cause performance degradation and requires extensive fine-tuning, while knowledge distillation uses a fixed architecture without gradual compression.

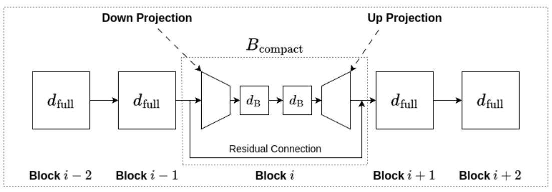

Figure 1: A transformer model where block i is substituted with a compact block, B_{compact}, containing two internal layers with hidden dimension d_{B}. Including down- and up-projection for compatability.

To address these issues, we propose *iterative structured knowledge distillation* (ISKD), which combines both methods. ISKD gradually replaces full transformer layers with smaller, retrained compact blocks, combining structured pruning with knowledge distillation. A visual representation of an ISKD pruned model is shown in Figure 1.

Our contributions are: 1) We propose ISKD [1] 2) We evaluate ISKD on language modeling and commonsense reasoning with GPT-2 and Phi-1. 3) We provide an in-depth analysis of the efficiency of ISKD.

2 Methodology

In this section, we present the compact block approach, describe the training of these smaller layers, and outline the setup, including model selection

[1] Code is available at https://github.com/Malthehave/ISKD-COLING-2025.

and pruning details.

2.1 Compact Block

Our goal is to compress transformer blocks by creating smaller, more efficient models. We introduce a compact block, B_{compact}, designed to mimic a compressed version of the transformer blocks in the pre-trained model (see Figure 1). For a transformer model with hidden dimension d_{full}, B_{compact} has a smaller hidden dimension, d_{B}, and fewer attention heads. To maintain compatibility, we down- and up-sample weights via linear projection. These compact blocks are introduced iteratively, requiring minimal calibration.

Although B_{compact} shares the same fundamental architecture of the pre-trained model, any model architecture with fewer parameters than the original block could be used. The decision to share architectures was driven by the goal of retaining the model's functionality, which we hypothesize is easier with similar architectures.

2.2 Block Training

After substituting a specific block with B_{compact}, the model is fine-tuned on a dataset similar to its original training dataset. We utilize 40M tokens from the TinyStories dataset (Eldan and Li, 2023), chosen for its high-quality, compact short stories, allowing efficient fine-tuning. In contrast, Phi-1 was pre-trained on 54B tokens. Each pruned model is calibrated on the TinyStories subset for 1 epoch using the AdamW optimizer, with weight decay of 0.01, cross-entropy loss, and a batch size of 16. A dual learning rate strategy is applied, prioritizing updates to the compact block while allowing the rest of the model to adapt. Learning rates are set as follows:

Compact Block The new compact block, with randomly initialized weights, uses a learning rate of $\alpha_{\text{new}} = 1e - 4$, consistent with the learning rate used in the original fine-tuning of Phi-1 (Gunasekar et al., 2023).

Existing Model Components To fine-tune pre-trained components without overwriting existing knowledge we experiment with a learning rate 5 times smaller at $\alpha_{\text{old}} = 2e - 5$.

2.3 Setup

We evaluate ISKD with two generative decoder-only language models: GPT-2 (117M parameters) and Phi-1 (1.4B parameters).

For GPT-2, we design B_{compact} with two layers of size 128, an intermediate MLP size of 512, and two attention heads, resulting in a 5.11% parameter reduction per block substitution and a total model reduction of 61.32% when all 12 blocks were substituted. For Phi-1, we design B_{compact} with four layers, a hidden size of 512, an intermediate MLP size of 2048, and one attention head. Each substitution reduces Phi-1's parameters by 2.5%, with a total possible model reduction of 60.32% after 24 block substitutions. These configurations are chosen to achieve a similar total parameter reduction of about 60% across both models. It should be noted that further exploration of these configurations could likely yield more efficient solutions.

3 Results

In this section, we introduce the datasets for evaluating ISKD on GPT-2 and Phi-1, followed by a comparison with two established pruning methods.

3.1 Data

We evaluate the pruned models in two ways. First, we use the Children's Book Test (CBT) (Hill et al., 2016) to test a model's ability to predict the correct sentence out of ten common noun options, given no prior context, which challenges the model's understanding of language structure and context-free prediction. Second, we use HellaSwag (Zellers et al., 2019) to evaluate commonsense reasoning by presenting the model with an incomplete sentence followed by four possible continuations, which requires the model to select the most likely continuation. We use the common zero-shot setup for both datasets.

3.2 Results of Pruning GPT-2

We apply ISKD to GPT-2 and evaluate performance on CBT and HellaSwag, using the random block sequence $6, 8, 4, 10, 2, 12, 3, 11, 5, 9, 7, 1$. As outlined in Section 4.1, random sequences work effectively as long as early layers are not pruned first. For the purpose of this study, we define early layers as the first third of the model's layers, following the approach outlined by Tenney et al. (2019). In their study of BERT-Large, which has 24 layers, they considered the first seven layers to be early layers, as most examples can be correctly classified at these layers, likely due to the presence of heuristic shortcuts. To evaluate the effectiveness of our approach, we compare our results to those of a baseline uniform classifier.

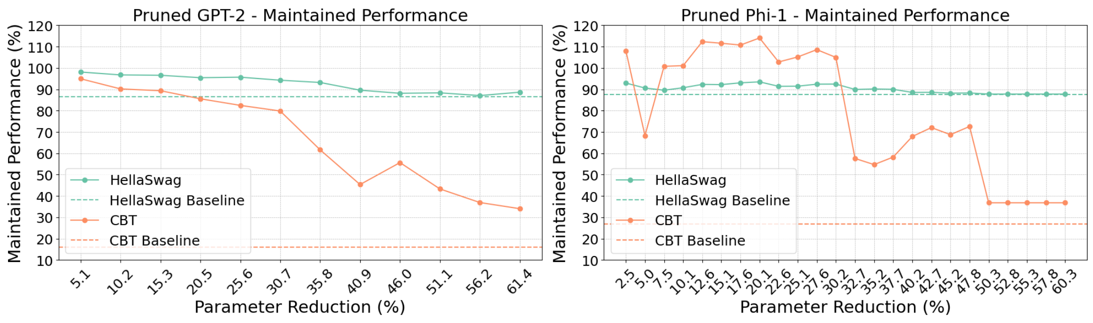

Figure 2: Test results (accuracy) from applying ISKD on 12 blocks of GPT-2 and 24 blocks of Phi-1.

As shown in Figure 2, substituting six blocks reduces parameters by 30.68%, maintaining 80.04% performance on CBT and 94.25% on HellaSwag. While performance declines gradually with each substitution, the impact is minimal across the first six block substitutions.

3.3 Results of Pruning Phi-1

We apply ISKD to Phi-1 with a random block sequence while avoiding early block pruning. Figure 2 shows that ISKD reduces parameters by 30.16% while improving CBT performance by 4.93% and maintaining 92.43% performance on HellaSwag. After the first block substitution, CBT performance improves, while HellaSwag performance initially drops by 7.08%, but subsequent substitutions have minimal impact. Degradation to baseline occurs only after a 40.21% reduction in parameters.

Notably, the trajectory of maintained performance for Phi-1 appears irregular, particularly compared to GPT-2. This irregularity may indicate that the fine-tuning process for Phi-1 has not fully converged, which could stem from the model's larger size and greater complexity. Additional research could examine the relationship between fine-tuning duration and performance retention to ensure more consistent results across substitution sequences.

3.4 Comparison to knowledge distillation and structured pruning

To compare the effectiveness of ISKD, we conduct a comparative evaluation on GPT-2 against two alternative compression techniques: knowledge distillation (Hinton, 2015) and structured L1 pruning (Yang et al., 2019). The L1 structured pruning approach involves removing entire rows from the weight matrices to reduce parameters. To ensure a fair comparison, both pruned models are trained for six epochs on the TinyStories dataset, matching the training conditions of ISKD. Six epochs because for ISKD to reach a 30.68% parameter reduction it will have been fine-tuned six times.

Algorithm	**Reduction**	**CBT**	**HellaSwag**
ISKD	30.68%	79.82%	94.25%
Knowledge Dis.	34.17%	85.97%	90.23%
L1 Pruning	30.73%	78.12%	91.48%

Table 1: Maintained performance on CBT and HellaSwag for ISKD, knowledge distillation, and structured L1 pruning algorithms with their reduction in parameter count.

As shown in Table 1, ISKD outperforms structured pruning on both CBT and HellaSwag, indicating superior performance retention. While ISKD also outperforms knowledge distillation on HellaSwag, it performs slightly lower on CBT. This result suggests that ISKD's current training setup and gradual compression strategy may not fully preserve the nuanced linguistic understanding required for CBT, which evaluates syntax and context-free predictions.

This highlights an opportunity to refine ISKD further. Future work could investigate whether alternative fine-tuning datasets with greater linguistic complexity or diversity could better support the retention of capabilities evaluated by CBT. Similarly, exploring adjustments to the architectural properties of the compact blocks, such as their hidden size or number of attention heads, may help preserve performance on tasks with higher linguistic demands.

4 Analysis and Discussion

In this section, we analyze the block substitution sequence and its impact, followed by an examination of each compact block's output sensitivity on the final model.

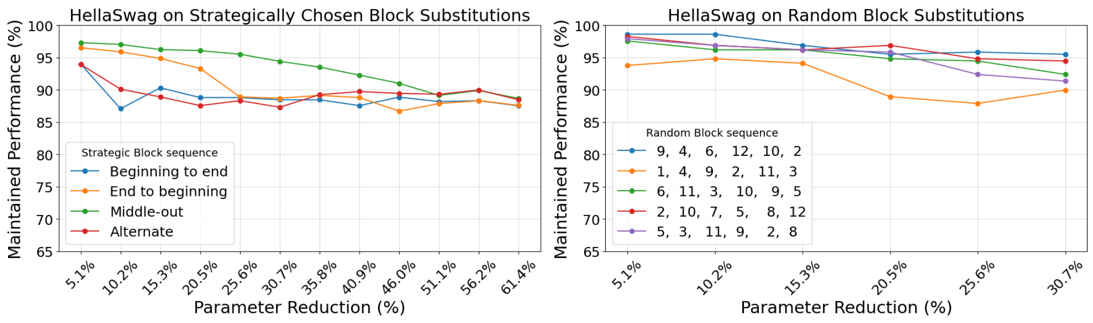

Figure 3: Results from pruning four strategically chosen and five randomly chosen block sequences in GPT-2. Both evaluated on test accuracy.

4.1 Block Substitution Sequence

While both GPT-2 and Phi-1 are evaluated, our analysis focuses on GPT-2 due to its lower computational requirements. We report results for GPT-2, with the assumption that similar trends likely apply to larger models like Phi-1.

To assess the robustness of ISKD regarding block substitution order, we experiment with four block sequences: beginning to end, end to beginning, middle-out, and alternating. As shown in Figure 3, sequences substituting early blocks first (beginning to end and alternating) perform worse, while middle-out performs best. This suggests the critical role of early blocks and the relatively lower importance of middle blocks. Prior work has shown that early transformer layers capture basic syntactic information (Tenney et al., 2019), which may explain the impact of substituting these blocks.

To further assess ISKD's robustness with respect to random block substitution sequences, we apply it to five randomly generated block sequences, pruning the first six blocks in each. Figure 3 shows that random sequences exhibit similar performance trajectories during the substitution process. These findings reinforce that ISKD's effectiveness is mainly affected by early block substitutions, while substitution order in middle blocks has minimal impact. The figure also shows that the cumulative effect of the number of blocks substituted remains a significant factor.

4.2 Sensitivity Analysis

Minimal impact on performance, despite alterations in the block substitution sequence, prompts an investigation into the functional contribution of individual blocks. Specifically, we examine whether substituted blocks significantly influence model performance or if they indicate overparameterization. The role of residual connections linking the input to the output of B_{compact} raises the question of potential redundancy in its computations.

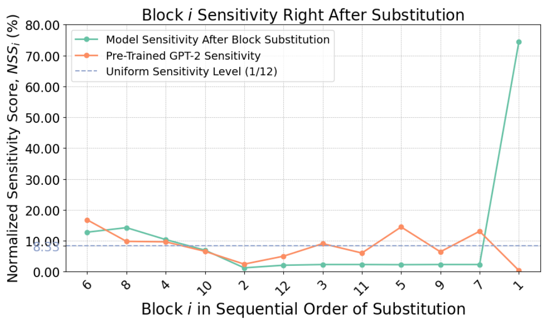

Figure 4: $NSS_i(x, y)$ values of blocks after ISKD substitution compared to $NSS_i(x, y)$ of blocks in unpruned GPT-2. Computed from the validation set.

To investigate this, we conduct a sensitivity analysis to quantify each block's impact on GPT-2's output using a gradient-based sensitivity score $SS_i(x, y)$ (Bansal et al., 2023):

$$SS_i(x, y) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{val}}}\left[\left|\mathbf{b}_i(x) \odot \frac{\partial \mathcal{L}(y, \hat{y}(x))}{\partial \mathbf{b}_i(x)}\right|\right] \quad (1)$$

Here, $\mathbf{b}_i$ represents the output of the i-th block, and $\mathcal{L}$ is the cross-entropy loss for sequences of T tokens x and labels y from the validation set. We normalize sensitivity scores and compare across n blocks, applying softmax to get $NSS_i(x, y)$ values for each of the $i = 12$ pruned models.

Figure 4 shows $NSS_i(x, y)$ values after applying ISKD on each block. No block has a value of zero, indicating that all compact blocks contributed to the model's output. Note that $\sum_{i=1}^{n} NSS_i(x, y)$ does not equal 1 because the scores reflect values after each block substitution.

Figure 4 also shows that the first five compact blocks align closely with their unsubstituted counterparts, suggesting that B_{compact} successfully approximates the functional roles of its larger counterpart. Substituting the final block in the sequence, which was the first block in the model, leads to a sensitivity spike of 74%. As discussed in Subsection 4.1, this also suggests that early blocks play a critical role, indicating further fine-tuning is needed for the final compact block.

5 Conclusion

This paper introduced ISKD as a novel approach to improving the efficiency of large language models. ISKD serves as an alternative to structured pruning and knowledge distillation by combining both methods. It avoids the performance loss associated with structured pruning and offers more flexibility than distillation, which requires a fixed architecture. ISKD reduces model parameters by 30.68% for GPT-2 and 30.16% for Phi-1. And it maintains over 80% of their original performance on language modeling and commonsense reasoning tasks.

Future work could explore optimizing ISKD for larger models, particularly reducing sensitivity in early layers during pruning. Additionally, applying ISKD to other transformer architectures and domains beyond language modeling may reveal broader applications for efficiently scaling deep learning models.

Limitations

The first main limitation of this study is the absence of hyperparameter optimization that potentially underestimates the full capabilities of ISKD. Tuning parameters such as learning rates, batch sizes, and compact block architectures could yield improved performance retention or higher compression rates. Choosing the right fine-tuning dataset could also play an important role in performance. Another limitation is that the evaluation relies on a limited set of benchmarks. While CBT and HellaSwag provide insights into language modeling and commonsense reasoning, they do not comprehensively represent the diverse landscape of NLP tasks. A more extensive evaluation across various task types would offer a more robust assessment of the compressed models' capabilities.

Acknowledgements

We thank the NLPnorth research unit at the IT University of Copenhagen (ITU) for their invaluable feedback throughout this work. We also thank the HPC team at ITU for providing access to the computational resources that made this research possible. Finally, we are grateful to the three anonymous reviewers for their insightful comments and suggestions.

References

Hritik Bansal, Karthik Gopalakrishnan, Saket Dingliwal, Sravan Bodapati, Katrin Kirchhoff, and Dan Roth. 2023. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11833–11856, Toronto, Canada. Association for Computational Linguistics.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. In *4th International Conference on Learning Representations, ICLR 2016*.

Geoffrey Hinton. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Saidul Islam, Hanae Elmekki, Ahmed Elsebai, Jamal Bentahar, Nagat Drawel, Gaith Rjoub, and Witold Pedrycz. 2023. A comprehensive survey on applications of transformers for deep learning tasks. *Expert Systems with Applications*, page 122666.

Feyza Duman Keles, Pruthuvi Mahesakya Wijewardena, and Chinmay Hegde. 2023. On the computational complexity of self-attention. In *International Conference on Algorithmic Learning Theory*, pages 597–619. PMLR.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Abir Rahali and Moulay A Akhloufi. 2023. End-to-end transformer-based models in textual-based nlp. *AI*, 4(1):54–110.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient transformers: A survey. *ACM Comput. Surv.*, 55(6).

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Chen Yang, Zhenghong Yang, Abdul Mateen Khattak, Liu Yang, Wenxin Zhang, Wanlin Gao, and Minjuan Wang. 2019. Structured pruning of convolutional neural networks via l1 regularization. *IEEE Access*, 7:106385–106394.

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.

A Appendix - Computational Efficiency Analysis of ISKD

In this appendix, we explain the computational efficiency analysis for ISKD, including the results of Multiply-Accumulate Operations (MACs) and Floating Point Operations (FLOPs) calculations. As shown in Figure 5, ISKD reduces the number of GMACs and GFLOPs required for a forward pass in GPT-2 linearly. For every block substituted, GMACs decreased by 0.85 and GFLOPs by 1.7. These reductions illustrate the computational benefits of ISKD in reducing overall model complexity.

The ISKD-pruned GPT-2 model with a 30.68% parameter reduction, from Section 3.4, has 10.53 GMACs. In contrast, the knowledge-distilled model achieves a 34.17% parameter reduction with 11.03 GMACs. Meanwhile, the unpruned GPT-2 model has a GMAC count of 16.11.

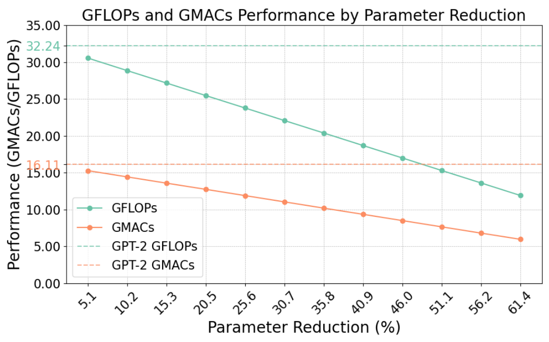

Figure 5: GMACS and GFLOPS VS. parameter reduction on GPT-2.