# Revisiting Intermediate-Layer Matching in Knowledge Distillation: Layer-Selection Strategy Doesn't Matter (Much)

**Zony Yu,  Yuqiao Wen,  Lili Mou**[*]
Dept. Computing Science & Alberta Machine Intelligence Institute (Amii), University of Alberta
[*]Canada CIFAR AI Chair, Amii
zony249@gmail.com, yq.when@gmail.com, doublepower.mou@gmail.com

## Abstract

Knowledge distillation (KD) is a popular method of transferring knowledge from a large "teacher" model to a small "student" model. Previous work has explored various layer-selection strategies (e.g., forward matching and in-order random matching) for intermediate-layer matching in KD, where a student layer is forced to resemble a certain teacher layer. In this work, we revisit such layer-selection strategies and observe an intriguing phenomenon that layer-selection strategy does not matter (much) in intermediate-layer matching—even seemingly nonsensical matching strategies such as *reverse matching* still result in surprisingly good student performance. We provide an interpretation for this phenomenon by examining the angles between teacher layers viewed from the student's perspective. Our work sheds light on KD practice, as layer-selection strategies may not be the main focus of KD system design, and vanilla forward matching works well in most setups.[1]

## 1 Introduction

Large language models have achieved impressive performance in various NLP tasks (Hurst et al., 2024; Guo et al., 2025). However, they need a large number of parameters, making the models cumbersome and difficult to run in resource-restricted scenarios. Knowledge distillation (KD; Hinton et al., 2015) is a widely adopted method to reduce model parameters by training a small "student" model from a large "teacher." With KD, the student is often able to retain most of the teacher's performance while using a fraction of its parameters (Gu et al., 2024; Guo et al., 2025; Wen et al., 2025; Yang et al., 2025).

Common KD approaches can be generally divided into two categories: prediction matching and intermediate-layer matching. Matching the prediction is usually mandatory, as it informs the student of the task to solve. This can be achieved by minimizing the divergence of predicted distributions (Hinton et al., 2015; Wen et al., 2023; Cui et al., 2025) or using reinforcement learning (Hao et al., 2022; Li et al., 2024).

Intermediate-layer matching distills the teacher's hidden states (i.e., intermediate layers) to the student (Sun et al., 2019; Jiao et al., 2020; Gromov et al., 2025). This approach often involves minimizing the distance between the student's and teacher's hidden states (usually with a linear mapping if the dimensions do not match). Also, a layer-selection strategy is required to specify which teacher layer is matched to which student layer.

Traditionally, researchers have explored various layer-selection strategies. Sun et al. (2019) match the student's layers to evenly spaced teacher layers; Passban et al. (2021) and Wu et al. (2021) learn an attention mechanism over the teacher's layers; Haidar et al. (2022) match the student's layers to randomly selected layers from the teacher, albeit in sorted order; and Wang et al. (2021) match the last student layer to a teacher layer close to the end.

Despite numerous previous efforts, we observe that the effectiveness of existing layer-selection strategies is not convincing, as previous studies often lack proper controlled comparisons. For example, Passban et al. (2021) only compare their work with the vanilla layer-matching baseline, and Wang et al. (2021) compare layer-selection strategies without controlling weight initializations. In addition, existing work tends to omit important distillation settings such as decoder-only model architectures. As a result, there is a lack of understanding about how different strategies compare against each other.

In this work, we aim to better understand how layer-selection strategies affect student performance in KD. We conduct systematic controlled

[1] The code is released at the following repo: https://github.com/MANGA-UOFA/Layer-Selection

*Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 1686–1694
December 20-24, 2025 ©2025 Association for Computational Linguistics

experiments across four models (BERT, BART, T5, and Qwen3) on eight tasks (including both classification and generation). We also consider two weight initialization strategies: random initialization and weight copying from the teacher. Our experiments reveal intriguing findings that

1. The layer-selection strategy doesn't matter (much), as different strategies perform similarly to each other, and
2. Regardless of layer-selection strategies, intermediate-layer matching itself is a highly effective KD method, outperforming the no-matching baseline.

We provide an interpretation for this phenomenon based on geometric analysis of the hidden states: from the student's point of view, the angles between two teacher layers are often acute; thus, matching any teacher layer pulls the student layer in a similar direction. As a result, intermediate-layer matching indeed benefits KD, although the matching strategy does not matter (much).

Our study offers a practical suggestion for KD: we recommend KD practitioners to focus on other aspects of KD systems (e.g., distillation temperature, choice of $f$-divergence loss functions), while simply using forward matching for intermediate-layer matching as a default strategy, if computing resources are limited.

## 2 Background and Related Work

Knowledge Distillation (KD) is a method of transferring rich knowledge contained in a teacher model to a student model. To inform the student of the task, it is essential to match the student's and teacher's predictions (Bucilă et al., 2006). Hinton et al. (2015) suggest minimizing the Kullback–Leibler (KL) divergence between the teacher and student distributions, denoted by $p$ and $q_{\boldsymbol{\theta}_s}$, respectively. The KL distillation loss is given by

$$\mathcal{L}_{\text{KL}}(\boldsymbol{\theta}_s) = \mathbb{E}_{\mathrm{y}\sim p(\mathrm{y}|\mathrm{x})}\left[\log \frac{p(\mathrm{y}|\mathrm{x})}{q_{\boldsymbol{\theta}_s}(\mathrm{y}|\mathrm{x})}\right] \qquad (1)$$

where x represents the input, and the output y (conditioned on x) is sampled from $p$. The student's parameters $\boldsymbol{\theta}_s$ are optimized, whereas the teacher's parameters are frozen.

Other than minimizing KL, different prediction matching approaches have been proposed. When the teacher distribution is diverse, for example, the reverse KL divergence (Tu et al., 2020; Gu et al., 2024) is used due to its mode-seeking behavior, i.e., the student only focuses on one of the high-probability regions in the teacher distribution (Bishop, 2006). Wen et al. (2023) propose an $f$-divergence KD framework, where symmetric divergences (such as Jensen–Shannon and total variation distance) provide a balance between mode averaging and mode seeking. Reinforcement learning can also be applied to KD (Hao et al., 2022; Li et al., 2024), which makes the student aware of its prefix and addresses the exposure bias problem (Bengio et al., 2015).

Regarding intermediate-layer matching, it distills the teacher's hidden states, thus providing additional supervisory signals to the student (Sun et al., 2019; Wen et al., 2025). Let $\mathcal{M} = \{(\varsigma_i, \tau_i)\}_i$ be the mapping between certain student and teacher layers, i.e., the $\varsigma_i$th layer of the student is mapped to the $\tau_i$th layer of the teacher. Intermediate-layer matching typically penalizes the distance between the matched layers, given by

$$\mathcal{L}_{\text{hid}}(\boldsymbol{\theta}_s, \{\boldsymbol{A}_i\}_i) = \sum_i \operatorname{dist}(\boldsymbol{A}_i \boldsymbol{h}^{(s)}_{\varsigma_i}, \boldsymbol{h}^{(t)}_{\tau_i}) \qquad (2)$$

where dist is a distance metric (such as mean squared error). The trainable linear operator $\boldsymbol{A}_i$ transforms the student's hidden state $\boldsymbol{h}^{(s)}_{\varsigma_i}$ to the space of the teacher's hidden state $\boldsymbol{h}^{(t)}_{\tau_i}$, if their dimensions do not match. Otherwise, $\boldsymbol{A}_i$ may be an identity matrix.

Intermediate-layer matching can be applied to different types of representations. Traditionally, this is achieved by matching the student's and teacher's activations (Sun et al., 2019; Sanh et al., 2019). Other studies match attention logits (Jiao et al., 2020), query–key–value relations (Wang et al., 2021), and cross-sample relations (Park et al., 2019; Huang et al., 2023). In our work, we focus on matching activations because it is the most fundamental approach in intermediate-layer matching.

Various layer-selection strategies have been proposed for matching a shallow student to a deep teacher. Sun et al. (2019) and Jiao et al. (2020) suggest mapping evenly spaced teacher layers to the student. Passban et al. (2021) and Wu et al. (2021) match each student layer to a weighted combination of all teacher layers to retain more knowledge. Haidar et al. (2022) randomly reselect a sequence of teacher layers to match with the student (in order) after each epoch, so that the student is exposed to different teacher layers. Wang et al. (2021) and Ko et al. (2023) suggest mapping the last non-prediction student layer to a teacher layer close to the end.

Overall, although previously proposed methods work well within their environments, it remains unclear how various layer-selection strategies compare under a controlled setup, or the extent they contribute to KD. Thus, we address this by conducting systematic investigations across numerous tasks, models, and initialization strategies, and provide an interpretation for our observations.

## 3 Approaches

Intermediate-layer matching requires a strategy to select which teacher layers are matched with which student layers. In this study, we explore both standard and seemingly nonsensical layer-selection strategies to uncover its effect on intermediate-layer matching KD.

**Forward Matching.** In this variant, lower student layers are matched to lower teacher layers. In particular, we follow Sun et al. (2019) and select evenly spaced teacher layers for matching.

**All-to-One Matching.** In this variant, all student layers are matched to the middle teacher layer. While the idea of matching to one layer is proposed by previous studies (Wang et al., 2020, 2021), we slightly modify their approaches (i.e., matching all student layers instead of one), for fair comparison with the rest of our settings.

**Reverse Matching.** We experiment with a counterintuitive strategy, where matching is in reverse order (i.e., lower student layers matched to upper teacher layers). This seemingly nonsensical strategy sheds light on the mechanism of intermediate-layer matching.

**Out-of-Order Random Matching.** We choose the same teacher layers as forward matching, then randomly shuffle the order. The order is maintained during distillation. We average the performance across five seeds to evaluate the effect of different random mappings.

Note that the intermediate-layer matching loss is combined with the predictor's KL loss by $\mathcal{L} = \mathcal{L}_{\text{KL}} + \lambda\mathcal{L}_{\text{hid}}$, where $\lambda$ is a hyperparameter to balance the losses. We experimentally determined $\lambda = 3$ to be a good balance between KL and intermediate-layer matching losses, since it produces improved performance over the No Matching ($\lambda = 0$) baseline, while higher values of $\lambda$ may negatively impact performance.

We emphasize that across all layer-selection strategies, we fix the set of teacher layers that are available for the student to learn from. In other words, the only difference among the layer-selection strategies is the **order** of matching.

## 4 Results and Analysis

**Setups.** We analyzed different layer-selection strategies on both classification and generation tasks. For classification, we adopt the widely used MNLI (Williams et al., 2018), QQP,[2] QNLI (Rajpurkar et al., 2016), and SST-2 (Socher et al., 2013) using the BERT model (Devlin et al., 2019). We also include more challenging tasks, namely HellaSwag (Zellers et al., 2019) and CommonsenseQA (CoQA; Talmor et al., 2019), using the more recent Qwen3 models (Yang et al., 2025). For generation, we use the popular DART (Nan et al., 2021) and WMT16 En–Ro (Bojar et al., 2016) datasets. We evaluate BART (Lewis et al., 2020) on the DART benchmark and T5 (Raffel et al., 2020) on WMT16, respectively. Our selection of tasks and models give a comprehensive coverage of different tasks and model architectures.

For moderate-sized student models (BERT, BART, and T5), we explore two parameter initialization strategies: copying the weights from select teacher layers and random initialization. The former is a practical method used to quickly transfer knowledge from select teacher layers to the student (Sanh et al., 2019; Shleifer and Rush, 2020), and the latter is used to study the effects of layer-selection strategy in isolation. For the Qwen3 large language model, we only perform weight-copying because it is often not feasible to directly finetune large models on downstream tasks without warming up the weights (commonly achieved through pretraining).

More experimental details can be found in Appendix A.

**Main Results.** In Table 1, we present the main results of our layer-selection experiments. In Rows 1–2, our finetuned teachers perform similarly to previous work (Devlin et al., 2019; Nan et al., 2021; Wen et al., 2023), showing that we have successfully set up the environment for KD experiments.

We examine different layer-selection strategies. As shown in Rows 9–12, the student model

[2] https://www.kaggle.com/c/quora-question-pairs

| Model | | Layer Matching | # | Classification Tasks | | | | Generation Tasks | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | MNLI-m/mm | QQP | QNLI | SST-2 | DART | WMT16 |
| | | | | Acc | Acc / F1 | Acc | Acc | BLEU | BLEU |
| Teacher | Previous work | – | 1 | 84.6 / 83.4 | – / 71.2 | 90.5 | 93.5 | 48.56 | 25.82 |
| | Our replication | – | 2 | 84.5 / 84.1 | 89.0 / 71.4 | 90.8 | 93.1 | 48.80 | 25.90 |
| Student | Randomly initialized | None | 3 | 63.2 / 63.6 | 81.5 / 56.4 | 61.2 | 81.1 | **38.76** | 8.02 |
| | | Forward | 4 | 72.5 / 72.0 | 83.9 / 61.3 | 64.7 | 85.1 | 32.64 | **18.13** |
| | | Reverse | 5 | 69.3 / 68.9 | **84.3 / 61.8** | **65.2** | 83.3 | 33.12 | 17.15 |
| | | All-to-one | 6 | **74.0 / 73.8** | 83.4 / 60.2 | 65.0 | **85.4** | 33.86 | 17.16 |
| | | Out-of-order rand | 7 | 70.5 / 70.5 | 82.4 / 58.8 | 64.4 | 82.9 | 32.73 | 16.71 |
| | Weights copied | None | 8 | 77.4 / 76.5 | 87.6 / 67.1 | 81.2 | 88.7 | 46.32 | 22.36 |
| | | Forward | 9 | **79.7 / 78.8** | **88.2 / 69.1** | **83.8** | **92.3** | 47.94 | **22.65** |
| | | Reverse | 10 | 79.2 / 78.2 | 88.1 / 68.3 | 83.2 | 89.6 | **48.45** | 21.57 |
| | | All-to-one | 11 | 79.4 / 78.7 | 87.6 / 68.6 | 82.8 | 91.4 | 47.10 | 21.89 |
| | | Out-of-order rand | 12 | 79.0 / 78.0 | 87.5 / 67.2 | 82.6 | 90.7 | 47.99 | 22.01 |

Table 1: Main results. We use BERT on classification tasks, BART on DART, and T5 on WMT16.

| Model | Layer Matching | # | HellaSwag | CoQA |
|---|---|---|---|---|
| | | | Acc | Exact Match |
| Teacher | – | 1 | 63.11 | 75.02 |
| Student | None | 2 | 35.16 | 33.73 |
| | Forward | 3 | **37.85** | 35.40 |
| | Reverse | 4 | 35.01 | **37.67** |
| | All-to-one | 5 | 34.99 | 37.35 |
| | Out-of-order rand | 6 | 35.47 | 34.80 |

Table 2: KD results on more challenging tasks using the recent Qwen3 model.

achieves similar results across different strategies, with only 2–3 points difference in accuracy for classification tasks and 1–2 points difference in BLEU for generation tasks. Notice that Reverse Matching and Out-of-Order Random Matching appear nonsensical, when in fact they still achieve close performance to Forward Matching, largely outperforming No Matching. The results show that layer-selection strategy has an unexpectedly small effect on student performance.

It should be emphasized that intermediate-layer matching indeed helps KD compared with No Matching,[3] even though the matching strategy does not play a significant role. On MNLI, for example, all strategies improve upon No Matching by six to ten points with random weight initialization and two points with weight-copying.

Next, we take a closer look at how different layer-selection strategies behave under the two parameter initialization settings. To reiterate, weight-copying is a simple and practical method of transferring the teacher's knowledge to the student (Sanh et al., 2019; Shleifer and Rush, 2020). We also experiment with randomly initialized students in order to disentangle the effects of layer-matching. In Rows 4–7, we see that layer-selection strategies perform similarly to one another, and for the most part better than No Matching.

**Results on Challenging Tasks.** We further compare different layer-selection strategies on the more challenging HellaSwag and CoQA tasks, which involve more reasoning. As such, we employ the newer Qwen3 models. As shown in Table 2, we observe a similar trend as in our main results, i.e., different strategies generally produce similar results. This suggests the generality of our findings.

**An Interpretation Based on Vector Angles.** A curious question arises from these observations: why does intermediate-layer matching help KD, but different layer-selection strategies perform similarly? To answer this, we measure the angles between the teacher's layers, viewed from the student. Specifically, we measure the angles formed by two teacher layers' and one student layer's vector representations, depicted in Figure 1a. We show the phenomenon in the MNLI and WMT16 En–Ro datasets in Figures. 1b, 1c and 1d. We see that in both randomly initialized and weight-copied settings, the cosine similarity is positive, suggesting that the angles are mostly acute. In other words, the student layer is pulled to the same general direction regardless of which teacher layer it is matched to. This finding is consistent with Men et al. (2025) and Gromov et al. (2025), where they show that different layers may contain similar knowledge.

**Additional Results.** We further analyze the student depth in Appendix B and experimental stability in Appendix C.

[3] One exception is the DART experiment with randomly initialized weights, for which we suspect intermediate-layer matching causes the student to overfit. That said, different strategies still perform similarly to conventional Forward Matching, and thus, it does not contradict our general finding.

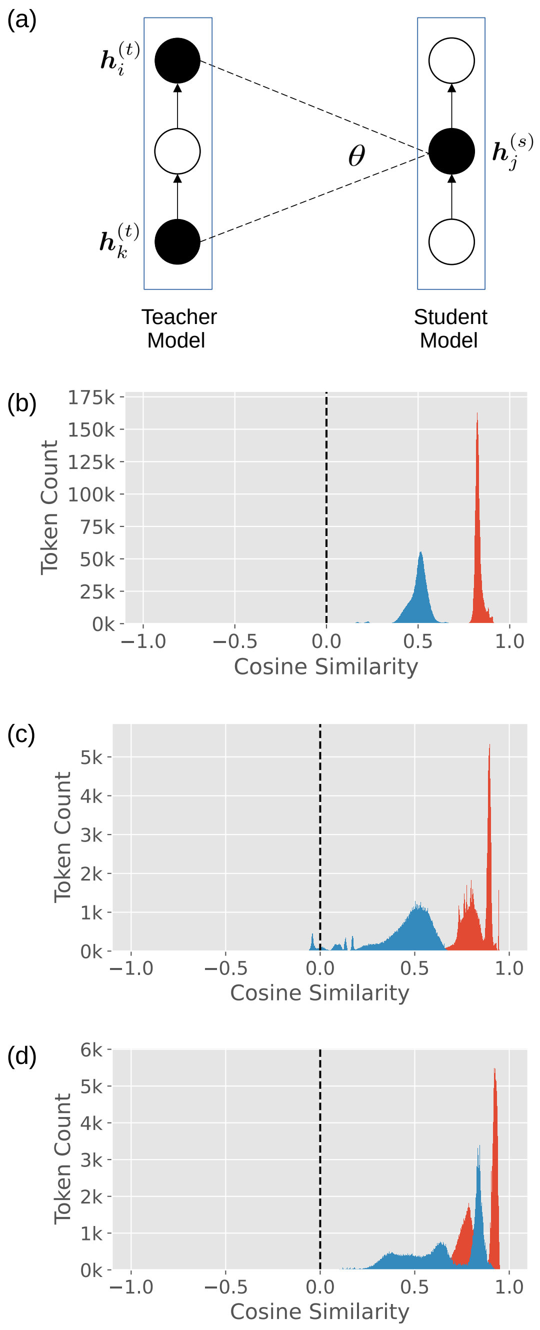

Figure 1: (a) Illustration of the angle calculation. Cosine similarities are shown for (b) MNLI classification, (c) Encoder in the WMT task, (d) and Decoder in WMT (bottom). Orange refers to the setup of random parameter initialization and blue refers to student weights initialized by the teacher.

## 5 Concluding Remarks

In this paper, we observe an intriguing phenomenon that, although intermediate-layer matching helps knowledge distillation, the layer-selection strategy does not matter (much); we also provide an interpretation based on the angles of teacher and student layers. Our work suggests potential limitations and oversights in previous work, where researchers present various heuristic layer matching methods when training their distilled systems, but their effect is not comprehensively studied. We advise the KD practitioners to focus their efforts on other areas of KD, for example, loss functions, initializations, and representation learning.

## 6 Limitations

In our work, we have experimented with various setups, including eight tasks (six classification and two generation), four model architectures, and two parameter initialization methods. Although the results are generally consistent, there is one exception that intermediate-layer matching does not help in the DART setup. This is understandable as empirical findings are often noisy. We suspect that it is due to the student model overfitting to the teacher's representations, since we are training a wider student model (compared to T5 and BERT students) from randomly initialized weights on a small dataset. That said, this does not contradict our conclusions, as all layer-selection strategies still perform equally bad.

Additionally, we clarify that our work focuses on KD in the fine-tuning regime (for a certain task) instead of pretraining, due to the limited resources. It is noticed that fine-tuning is the typical setting of how most practitioners perform KD. We also mention that this paper is intended for an NLP audience, therefore non-NLP experiments (such as computer vision) are outside the scope of this work.

It is also worth mentioning that our work does not suggest intermediate-layer matching is unhelpful for KD. Rather, we present an interesting phenomenon that the layer-selection strategy plays an insignificant role in the process. We argue that future studies on layer selection should have a more rigorous comparison of its effect.

## 7 Acknowledgments

We thank the reviewers and area chairs for their efforts. The research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Amii Fellow Program, the Canada CIFAR AI Chair Program, a donation from DeepMind, and the Digital Research Alliance of Canada (alliancecan.ca).

## References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, page 1171–1179.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, ···, and Andy Zou. 2024. Lessons from the trenches on reproducible evaluation of language models. *arXiv preprint arXiv:2405.14782*.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer-Verlag.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the Conference on Machine Translation: Shared Task Papers*, volume 2, pages 131–198.

Cristian Bucilă, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 535–541.

Xiao Cui, Yulei Qin, Yuting Gao, Enwei Zhang, Zihan Xu, Tong Wu, Ke Li, Xing Sun, Wengang Zhou, and Houqiang Li. 2025. SinKD: Sinkhorn distance minimization for knowledge distillation. *IEEE Transactions on Neural Networks and Learning Systems*, 36(7):11887–11901.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Dan Roberts. 2025. The unreasonable ineffectiveness of the deeper layers. In *The Thirteenth International Conference on Learning Representations*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, ···, and Zhen Zhang. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Md Akmal Haidar, Nithin Anchuri, Mehdi Rezagholizadeh, Abbas Ghaddar, Philippe Langlais, and Pascal Poupart. 2022. RAIL-KD: Random intermediate layer mapping for knowledge distillation. In *Findings of the Association for Computational Linguistics: NAACL*, pages 1389–1400.

Yongchang Hao, Yuxin Liu, and Lili Mou. 2022. Teacher forcing recovers reward functions for text generation. In *Advances in Neural Information Processing Systems*, pages 12594–12607.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Kun Huang, Xin Guo, and Meng Wang. 2023. Towards efficient pre-trained language model via feature correlation distillation. In *Advances in Neural Information Processing Systems*.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, ···, and Yury Malkov. 2024. GPT-4o system card. *arXiv preprint arXiv:2410.21276*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 4163–4174.

Jongwoo Ko, Seungjoon Park, Minchan Jeong, Sukjin Hong, Euijai Ahn, Du-Seong Chang, and Se-Young Yun. 2023. Revisiting intermediate layer distillation for compressing language models: An overfitting perspective. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 158–175.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Dongheng Li, Yongchang Hao, and Lili Mou. 2024. LLMR: Knowledge distillation with a large language model-induced reward. In *Proceedings of the Joint International Conference on Computational Linguistics, Language Resources and Evaluation*, pages 10657–10664.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2025. ShortGPT: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics*, pages 20192–20204.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, · · · , and Nazneen Fatema Rajani. 2021. DART: Open-domain structured data record to text generation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. 2019. Relational knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3967–3976.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. ALP-KD: Attention-based layer projection for knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13657–13665.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

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*.

Sam Shleifer and Alexander M. Rush. 2020. Pre-trained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 4323–4332.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4149–4158.

Lifu Tu, Richard Yuanzhe Pang, Sam Wiseman, and Kevin Gimpel. 2020. ENGINE: Energy-based inference networks for non-autoregressive machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 2819–2826.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pre-trained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, pages 2140–2151.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, pages 5776–5788.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. F-divergence minimization for sequence-level knowledge distillation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 10817–10834.

Yuqiao Wen, Freda Shi, and Lili Mou. 2025. Knowledge distillation for language models. In *Proceedings of the 2025 Annual Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 5: Tutorial Abstracts)*, pages 25–29.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1112–1122.

Yimeng Wu, Mehdi Rezagholizadeh, Abbas Ghaddar, Md Akmal Haidar, and Ali Ghodsi. 2021. Universal-KD: Attention-based output-grounded intermediate layer knowledge distillation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 7649–7661.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, · · · , and Zihan Qiu. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

## A Datasets and Models

We evaluate our layer-selection strategies on a variety of classification and generation tasks.

**GLUE.** The General Language Understanding Evaluation (GLUE) benchmark is a popular suite for natural language classification. From GLUE, we chose MNLI (Williams et al., 2018), QQP,[4] QNLI (Rajpurkar et al., 2016), and SST-2 (Socher et al., 2013), as these tasks have large training sets and produce robust model performance. For each task, we finetune the 12-layer $\text{BERT}_{\text{Base}}$ (Devlin et al., 2019) as the teacher. We adopted standard evaluation metrics, namely, accuracy for all tasks and $F_1$ as an additional metric for QQP.

**DART.** The DART dataset (Nan et al., 2021) is a popular data-to-text generation task. We followed Nan et al. (2021) and finetuned $\text{BART}_{\text{Large}}$ (Lewis et al., 2020) with 12 encoder and 12 decoder layers, which is the teacher model in the experiment. We report BLEU scores measuring textual overlap (Papineni et al., 2002).

**WMT16 En–Ro.** The WMT16 dataset (Bojar et al., 2016) provides parallel text between six different language pairs. For our experiments, we followed the setups in Wen et al. (2023), who chose the English–Romanian translation direction and used 100K from the 614K total samples for efficiency considerations. We also followed Wen et al. (2023) and finetuned 12-layer $\text{T5}_{\text{Base}}$ (Raffel et al., 2020) as the teacher, which has the same number of layers as the DART experiment. We also report BLEU scores as the evaluation metric.

**CommonsenseQA.** The CommonsenseQA dataset (Talmor et al., 2019) is a question-answering task where each question is followed by five possible answers. We concatenated the correct answer with the question, then finetuned the 40-layer Qwen3-8B for next-token prediction to use as the teacher. Following Biderman et al. (2024), we use zero-shot prompting to obtain responses from the language model and report exact match with the ground truth.

**HellaSwag.** The HellaSwag dataset (Zellers et al., 2019) is a difficult sentence completion task that presents prefix sequence along with four possible endings. Like the previous task, we finetuned a 40-layer Qwen3-8B model for next-token prediction. We follow Biderman et al. (2024) and use the language model to rank each ending according to perplexity, conditioned on the prefix.

For the student, we adopted the teacher's architecture but reduced the number of layers to three in the main experiments. Specifically, we use teacher Layers 4, 8, and 12 for matching. Note that, for BART and T5 models, this means three layers for the encoder and another three layers for the decoder. Moreover, we employed two parameter initialization strategies for the student: randomly initializing the weights and copying the weights from the corresponding teacher layer. The former isolates the effects of intermediate-layer matching from weight copying, whereas the latter is a more practical method that yields higher performance (Sanh et al., 2019; Shleifer and Rush, 2020).

Regarding the more challenging experiments, namely HellaSwag and CommonsenseQA, we shrunk the student by the same proportions as in the main setup. Specifically, we distilled the 40-layer teacher to a 10-layer student, using Layers 4, 8, · · · , and 40 in the teacher model for matching.

## B Analysis of Student Depths

We validate our intriguing phenomenon across students with different depths. Due to the limit of computing resources, we selected MNLI as the representative classification task, but include both DART and WMT16 En–Ro generation tasks. Specifically, we experimented with student models containing three, six, and nine layers, initialized by copying the teacher's weights. As seen in Table 3, different layer-selection strategies show similar performances, confirming that the layer-selection strategies do not matter (much) across student models with various depths.

## C Experimental Stability

In the main results (Table 1), excluding the Out-of-Order Random Matching setup, we ran every experiment only once due to the large number of models, tasks, and setups.

We show the stability of our results in Table 4

[4] https://www.kaggle.com/c/quora-question-pairs

| Model | Depth | Layer Matching | # | MNLI-m/mm | DART | WMT16 |
|---|---|---|---|---|---|---|
| | | | | Acc | BLEU | BLEU |
| Teacher | 12-layer | – | 1 | 84.5 / 84.1 | 48.80 | 25.90 |
| Student | 3-layer | None | 2 | 77.4 / 76.5 | 46.32 | 22.36 |
| | | Forward | 3 | **79.7** / **78.8** | 47.94 | **22.65** |
| | | Reverse | 4 | 79.2 / 78.2 | **48.45** | 21.57 |
| | | All-to-one | 5 | 79.4 / 78.7 | 47.10 | 21.89 |
| | | Out-of-order rand | 6 | 79.0 / 78.0 | 48.18 | 22.04 |
| | 6-layer | None | 7 | 82.1 / 81.3 | 46.88 | 24.91 |
| | | Forward | 8 | **83.5** / **82.9** | **48.45** | **25.00** |
| | | Reverse | 9 | 82.1 / 80.9 | **48.45** | 24.30 |
| | | All-to-one | 10 | 82.3 / 81.8 | 48.39 | 24.44 |
| | | Out-of-order rand | 11 | 82.3 / 81.5 | 48.03 | 24.38 |
| | 9-layer | None | 12 | 84.2 / 83.3 | 46.05 | **25.88** |
| | | Forward | 13 | 84.1 / **83.4** | 47.66 | 25.67 |
| | | Reverse | 14 | 83.2 / 82.4 | 47.01 | 25.11 |
| | | All-to-one | 15 | 83.2 / 82.5 | 46.95 | 25.43 |
| | | Out-of-order rand | 16 | **84.4** / 83.3 | **47.37** | 25.41 |

Table 3: Performance of different layer-selection strategies on students of different depths. Student's parameters are initialized by copying the weights of the teacher.

| Model | | Run | MNLI-m/mm | DART | WMT16 |
|---|---|---|---|---|---|
| | | | Acc | BLEU | BLEU |
| 3-layer Student | Randomly Initialized | 1 | 71.2 / 71.2 | **32.44** | 16.05 |
| | | 2 | **72.2** / **71.8** | 32.41 | 16.90 |
| | | 3 | 70.8 / 71.1 | 33.33 | 16.95 |
| | | 4 | 70.5 / 70.8 | 33.13 | **17.01** |
| | | 5 | 67.9 / 67.8 | 32.35 | 16.65 |
| | | Mean | 70.5±1.4<br>/70.5±1.4 | 32.73±0.41 | 16.71±0.35 |
| | Weights Copied | 1 | 79.3 / 78.3 | 48.18 | 21.79 |
| | | 2 | 78.5 / 77.4 | **48.49** | 21.93 |
| | | 3 | **79.7** / **78.6** | 47.65 | 21.86 |
| | | 4 | 79.2 / 78.5 | 48.08 | **22.53** |
| | | 5 | 78.5 / 77.3 | 47.54 | 21.95 |
| | | Mean | 79.0±0.47<br>/78.0±0.56 | 47.99±0.35 | 22.01±0.27 |

Table 4: Out-of-Order Random Matching experiments on MNLI, DART, and WMT16 En–Ro. For each task and parameter initialization strategy, we computed the mean and standard deviation of five runs.

by computing the standard deviation of five runs. Here, we chose Out-of-Order Random Matching because this is in theory the most noisy setup due to the stochasticity of layer matching.

In Table 4, we see that random initialization yields higher standard deviation than the weight-copied setting. This is understandable, as the former setup involves more randomness. Nevertheless, the model performs stably in both settings, showing that our results and findings are reliable.