Extended Abstract Track

On Layer-wise Representation Similarity: Application for Multi-Exit Models with a Single Classifier

Editors: List of editors' names

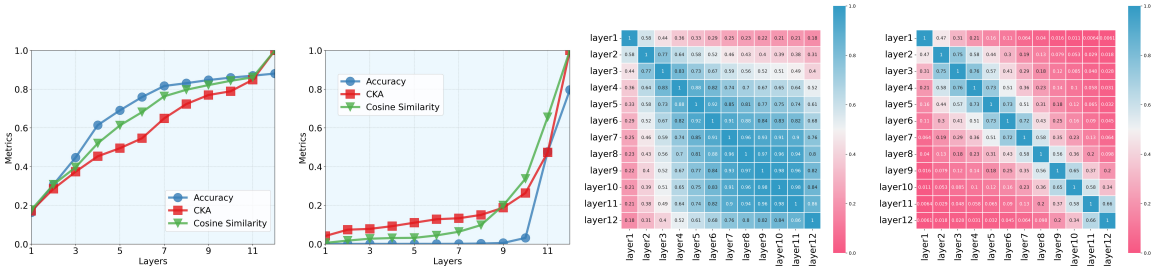
Abstract

Analyzing the similarity of internal representations within and across different models has been an important technique for understanding the behavior of deep neural networks. Most existing methods for analyzing the similarity between representations of high dimensions, such as those based on Canonical Correlation Analysis (CCA) and widely used Centered Kernel Alignment (CKA), rely on statistical properties of the representations for a set of data points. In this paper, we focus on transformer models and study the similarity of representations between the hidden layers of individual transformers. In this context, we show that a simple sample-wise cosine similarity metric is capable of capturing the similarity and aligns with the complicated CKA. Our experimental results on common transformers reveal that representations across layers are positively correlated, albeit the similarity decreases when layers are far apart. We then propose an aligned training approach to enhance the similarity between internal representations, with trained models that enjoy the following properties: (1) the last-layer classifier can be directly applied right after any hidden layers, yielding intermediate layer accuracies much higher than those under standard training, (2) the layer-wise accuracies monotonically increase and reveal the minimal depth needed for the given task, (3) when served as multi-exit models, they achieve on-par performance with standard multi-exit architectures which consist of additional classifiers designed for early exiting in shallow layers. To our knowledge, our work is the first to show that one common classifier is sufficient for multi-exit models. We conduct experiments on both vision and NLP tasks to demonstrate the performance of the proposed aligned training.

Keywords: Representation Similarity, Transformer, Early Exit

1. Introduction

Transformer models (26) have revolutionized vision and NLP tasks, including image classification (4), image generation (30; 17; 31), and language understanding (3; 16; 32). While larger models improve performance, they pose challenges in understanding and deployment (1). A promising direction for understanding these models is to study the representations across layers. Recent work has uncovered Neural Collapse (NC) (15; 5; 33; 22; 23) and progressive NC (10; 18; 27), where last layer classifiers progressively compress within-class features while enhancing the discrimination of between-class features from shallow to deep layers. Another line of work attempts to compare the similarity between representations, including CCA (21), CKA (13), OPT (9), and PNKA (12). These methods rely on statistical properties to support features with different dimensions for various architecture. However, these approaches are computational expensive. This work focus on transformer models, which stack several identical blocks with residual connections, resulting in consistent feature dimensions across layers. This property motivates evaluating the layer-wise representation similarity on a per-sample basis. Thus, we propose COsine Similarity (COS) as a more efficient metric for transformers and verify that it also aligns well with statistical methods like Centered Kernel Alignment (CKA). Base on it, we demonstrate that improving



(a) CIFAR10 (b) ImageNet1K (c) CIFAR10 (d) ImageNet1K

Figure 1: COsine Similarity (COS) of each layer with last layer(a,b) and across all layer pairs(c,d) on DeiT-S model.

layerwise COS can enhance layerwise accuracy and promote more early saturation events (8) , potentially boosting efficiency through early exit strategies. Finally, to achieve improved layerwise COS, we introduce an aligned training method that applies the final-layer classifier after any hidden layer for both classification and text generation tasks.

Contribution In summary, our contributions include:

- Introduce the layer-wise COsine Similarity (COS) metric, calculated per sample, to measure representation similarity across layers.
- Develop aligned training to enhance the layer-wise COS. Specifically, we use a single classifier after any hidden layer to improve the effectiveness of shallow layers.
- Conduct experiments using aligned training in both vision and NLP domains, like image/text classification and text generation tasks. We show that this method improves the inference efficiency while maintaining performance.

2. Measuring Layer-wise Representational Similarity in Transformer

A standard transformer consists of L identical layers. For image classification tasks, the predictions typically rely on the last layer hidden states $\boldsymbol{h}^L$ corresponding to the [CLS] token after L transformations. To analyze how features evolve across layers, we study layerwise similarity between the hidden states $\{\boldsymbol{h}^1, \boldsymbol{h}^2, ..., \boldsymbol{h}^L\}$ of the [CLS] token.

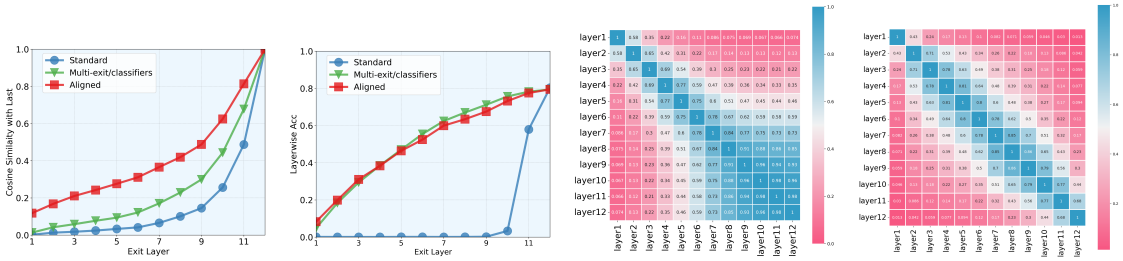
A simple method for measuring layer-wise representational similarity in transformers We propose measuring cosine similarity between features $\boldsymbol{h}^\ell$ and $\boldsymbol{h}^{\ell'}$ at layers ℓ and ℓ':

$$\text{COS} = \langle \boldsymbol{h}^\ell, \boldsymbol{h}^{\ell'} \rangle / \|\boldsymbol{h}^\ell\|_2 \|\boldsymbol{h}^{\ell'}\|_2.$$

This COsine Similarity (COS) metric provides a clear geometric interpretation of feature alignment. Unlike CKA (13), COS is not invariant to all transformations except isotropic scaling. COS is computed per sample and doesn't rely on inter-example structures. We average COS over all training samples in our experiments.

To verify the effectiveness of the sample-wise COS metric, we train the DeiT-S model (24) on CIFAR-10 and ImageNet1K dataset from scratch. We computed COS as well as the CKA between features in each layer and the last layer(Figure 1(a,b)), and plotted COS between all layer pairs as heatmap(Figure 1(c,d)). We get the following key observations:

- **COS aligns with CKA** The COS effectively reflects layer-wise representation similarity as it has the same trend as CKA. In appendix C.2, we show that COS is effective due to residual connections between transformer blocks has eliminated the rotation ambiguity.



(a) Cos Similarity (b) Layerwise Acc (c) Aligned (d) Multi classifiers

Figure 2: Comparison of ViT model for ImageNet by standard training, proposed aligned training, and the multi-exit/classifiers, in terms of (a) cosine similarity, (b) layer-wise testing accuracy, layerwise NC1 and linear probing accuracy at each layer, and (c-d) cosine similarities between all pairs of layers.

- **Positive layer-wise representation similarity** The COS for all pair of layers is evaluated on the same model for small and large datasets. Small datasets exhibit a ridge-to-plateau pattern, indicating that shallow layers drastically alter features while deep layers maintain relatively stable representations. In contrast, large datasets display a consistent ridge pattern, suggesting continuous feature refinement throughout the network.
- **Correlation between cosine similarity and accuracy** Applying the classifier to each hidden layer reveals a high correlation between COS and layer-wise accuracy.

3. Aligned Training for Enhancing Layer-wise Representational Similarity: Application for Multi-Exit Models with a Single Classifier

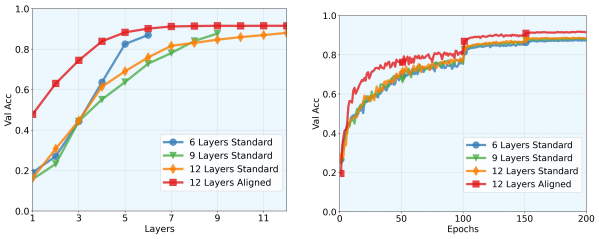
In the previous section, we observed that layer-wise similarity can be captured by a simple sample-wise COsine Similarity (COS) metric. In this section, we propose an aligned training method to enhance the COS across layers. Specifically, the last-layer classifier can be directly applied after any hidden layers, enabling a multi-exit model that use a single classifier. To the best of our knowledge, our work is the first to show that one common classifier is sufficient for multi-exit models. As shown in appendix(see Table 1), a simple classifier can significantly reduce the number of parameters for multi-exit models.

Aligned Training We enhance the layer-wise similarity by jointly optimizing the following aligned loss that is the weighted average of the CE loss from all the layers,

$$\mathcal{L}_{\text{aligned}}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\ell=1}^{L} \mathcal{L}_{\text{CE}}(\boldsymbol{W}\boldsymbol{h}^{\ell} + \boldsymbol{b}, \boldsymbol{y}), \tag{1}$$

Roughly speaking, the aligned loss (1) introduces CE loss for intermediate layers and would encourage each layer features $\boldsymbol{h}^{\ell}$ to align with the common classifier $\boldsymbol{W}$—as implied by the NC phenomenon—hence improving the representation similarity across layers.

Improved layer-wise representation similarity and accuracy Figure 2(a,c) show increased layer-wise representation similarity from aligned training. This method aligns features to a common classifier, significantly improving pair-wise feature similarity across layers. As a result, Figure 2(b) demonstrates substantially higher layer-wise accuracies when applying the last-layer classifier after each hidden layer, compared to standard training.



(a) Layerwise Acc (b) Training Curve

Figure 3: Layer-wise accuracy between standard model and aligned model.

Determine minimal number of layers Determining the optimal number of layers for a task can be challenging. Aligned training helps identify the minimal layers needed by enhancing shallow layer performance and reducing redundancy. As shown in Figure 3 for a DeiT-small model on CIFAR10, aligned training produces rapidly increasing layer-wise accuracy then saturate. The minimum layer count is the smallest layer achieving near-highest accuracy. No retraining is needed; selected layers can be used with the last-layer classifier. In contrast, standard training models show increasing accuracy without saturating, even with 12 layers, requiring multiple model sizes to determine minimal layer count. Interestingly, aligned training models truncated to 6 or 9 layers slightly outperform standard models of the same size and converge faster(Figure 3 (b)).

More early saturate events A "saturate event" (8) occurs when the model's final predicted token becomes the top candidate and remains unchanged across all subsequent layers. We validate the saturate events on vision models. Figure 4 illustrates the COS metric with last hidden states and the number of saturate events at each layer. We observe that aligned training encourages more early saturate events by increasing the cosine similarity with last hidden states. This demonstrates that the aligned model has a stronger potential for supporting early exit.

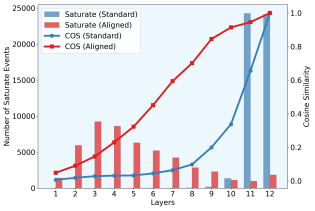


Figure 4: Aligned/ standard training on COS and saturation events.

Multi-exit model with a single classifier Multi-exit models (28; 7; 29) use different classifiers for each layer. Our approach uses a single classifier for all layers, decreasing the model size and exploiting the representation similarity. Compared to multi-exit training with multiple classifiers (29), our aligned training achieves higher COS and comparable layer-wise accuracy, as shown in Figure 2. For early exits, we use a confidence threshold. Figure 5 shows that in aligned training, most samples exit at early layers, unlike standard training where most exit at the last layer.

Applications on Language Models We extend our aligned training approach to NLP tasks, demonstrating its effectiveness in fine-tuning Large Language Models (LLMs). For text classification tasks, we evaluate **AlignedBERT** (BERT-Base with aligned loss) on GLUE benchmark tasks. Results show improved performance over the baseline in early layers. For text generation task, we evaluate **AlignedGPT** (GPT2 with aligned training)on Wikitext-103 dataset. Result shows it can maintain text quality while enable efficient generation using shallower layers. More results can be found in appendix B.

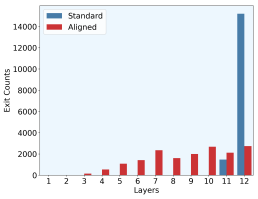


Figure 5: Number of samples exit at each layer during inference.

References

[1] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[5] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences*, 118(43):e2103091118, 2021.

[6] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

[7] Shijie Geng, Peng Gao, Zuohui Fu, and Yongfeng Zhang. Romebert: Robust training of multi-exit bert. *arXiv preprint arXiv:2101.09755*, 2021.

[8] Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, 2022.

[9] William L Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*, 2016.

[10] Hangfeng He and Weijie J Su. A law of data separation in deep learning. *Proceedings of the National Academy of Sciences*, 120(36):e2221704120, 2023.

[11] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[12] Camila Kolling, Till Speicher, Vedant Nanda, Mariya Toneva, and Krishna P Gummadi. Pointwise representational similarity. *arXiv preprint arXiv:2305.19294*, 2023.

[13] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[14] Shangyun Lu, Bradley Nott, Aaron Olson, Alberto Todeschini, Hossein Vahabi, Yair Carmon, and Ludwig Schmidt. Harder or different? a closer look at distribution shift in dataset reproduction. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, volume 5, page 15, 2020.

[15] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

[16] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[17] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021.

[18] Akshay Rangamani, Marius Lindegaard, Tomer Galanti, and Tomaso A Poggio. Feature learning in deep classifiers through intermediate neural collapse. In *International Conference on Machine Learning*, pages 28729–28745. PMLR, 2023.

[19] Yixuan Su and Nigel Collier. Contrastive search is what you need for neural text generation. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=GbkWw3jwL9.

[20] Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. A contrastive framework for neural text generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=V88BafmH9Pj.

[21] Bruce Thompson. Canonical correlation analysis. 2000.

[22] Christos Thrampoulidis, Ganesh Ramachandra Kini, Vala Vakilian, and Tina Behnia. Imbalance trouble: Revisiting neural-collapse geometry. *Advances in Neural Information Processing Systems*, 35:27225–27238, 2022.

[23] Tom Tirer, Haoxiang Huang, and Jonathan Niles-Weed. Perturbation analysis of neural collapse. In *International Conference on Machine Learning*, pages 34301–34329. PMLR, 2023.

[24] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[25] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[27] Peng Wang, Xiao Li, Can Yaras, Zhihui Zhu, Laura Balzano, Wei Hu, and Qing Qu. Understanding deep representation learning via layerwise feature compression and discrimination. *arXiv preprint arXiv:2311.02960*, 2023.

[28] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. Deebert: Dynamic early exiting for accelerating bert inference. *arXiv preprint arXiv:2004.12993*, 2020.

[29] Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. Berxit: Early exiting for bert with better fine-tuning and extension to regression. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: Main Volume*, pages 91–104, 2021.

[30] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

[31] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022.

[32] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[33] Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. *Advances in Neural Information Processing Systems*, 34:29820–29834, 2021.

Appendix

Notations and Organizations. The appendix provides additional experimental results and detailed information about the setup. Here, $\boldsymbol{h}^{\ell}$ represents the feature of an individual sample from layer ℓ of transformers. The appendix structure includes: parameter savings using a single classifier (Appendix A), additional language model experiments (Appendix B), sample-wise COS results (Appendix C.1), confirmation that residual connections in transformers eliminate rotation ambiguity of features (Appendix C.2), and the setup for the aligned training method (Appendix C.3).

Appendix A. Parameters Saving using Single Classifier

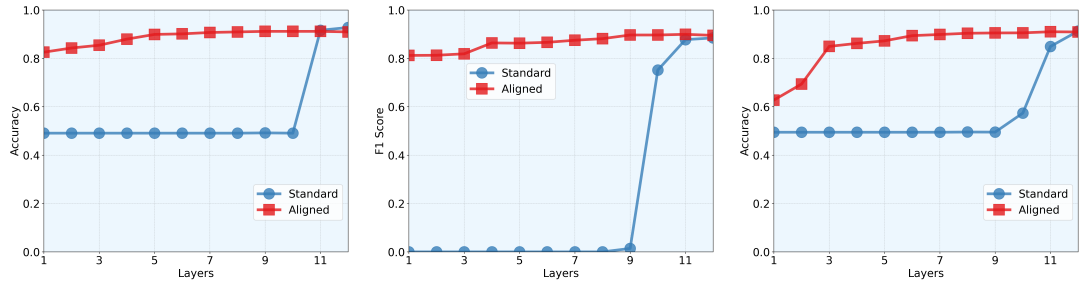
Table 1: Comparison of number of parameters across different architectures between multiple classifiers and single classifier(ours). Multiple classifiers refers to a multi-exit model with a different classifier at each layer. The last column shows that the percentage of saved parameters using single classifier.

Models	Hidden Dim	# of Classes	# of Layers	Multiple Classifiers (#Params)	Single Classifier (#Params)	#Param Saving
DeiT-S(24)	384	1,000	12	26.27M	22.05M	16.07%
DeiT-B(24)	768	1,000	12	95.02M	86.57M	8.89%
GPT-2(16)	768	50,257	12	541.57M	117.35M	78.39%
GPT-3(2)	12,288	50,257	96	233.67B	175.63B	25.10%
LLAMA-2 (25)	4,096	32,000	40	75.11B	70.35B	6.81%

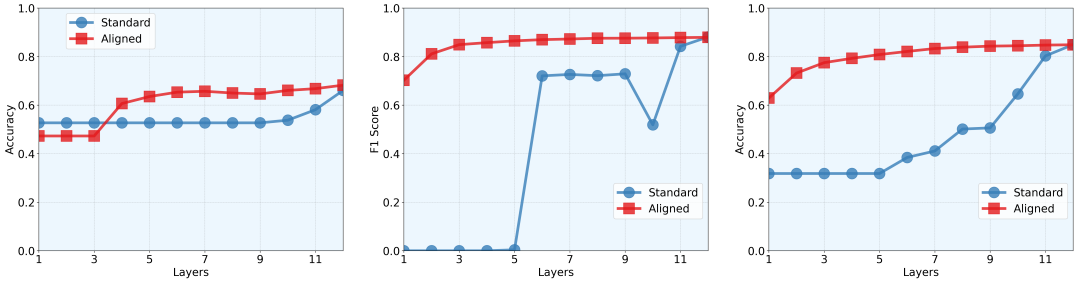
Appendix B. Applications on Language Models

All the previous experiments mainly focus on ViT for vision tasks. In this section, we will present additional experiments to demonstrate the performance of the aligned training for fine-tuning LLMs for NLP tasks. Specifically, we empirically evaluate the approach on text classification by fine-tuning the pretrained BERT models and generation tasks by fine-tuning the GTP2 model. For comparison, we also independently finetune the baseline models for both tasks using standard training.

AlignedBERT: text classification tasks We first study the text classification problem using the BERT model, which is similar to ViT for image classification as BERT also employs the [CLS] token for classification. We take a pretrained 12-layer BERT-Base model, extract the feature tied to the [CLS] token from each layer, and apply the same classification head to derive logits from each layer. Subsequently, we employ the aligned loss for fine-tuning to obtain the so-called AlignedBERT. We test AlignedBERT on the General Language Understanding Evaluation (GLUE) benchmark, which encompasses nine tasks that gauge understanding of natural language. This includes single-sentence tasks such as CoLA and SST-2, similarity and paraphrasing tasks like MRPC, STS-B, and QQP, as well as natural language inference tasks - MNLI, QNLI, RTE and WNLI. The accuracy or F-1 score per layer is depicted in Figure 6. AlignedBERT demonstrates better performance than the standard baseline for layer-wise accuracy. This indicates that most layers are redundant and that using only the first few layers of AlignedBERT can achieve good text classification performance.

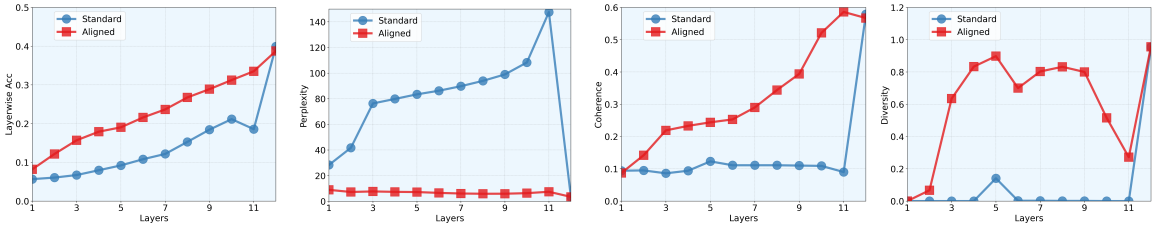


(a) SST-2 (b) MRPC (c) QNLI



(d) RTE (e) QQP (f) MNLI

Figure 6: Comparison of layer-wise scores for models trained with standard training and the proposed aligned training strategy, with $\text{Bert}_{\text{Base}}$ as the backbone.



(a) Prediction Accuracy (b) Perplexity (c) Coherence (d) Diversity

Figure 7: Evaluation of standard training and aligned training for GPT2 model on Wikitext-103 dataset in terms of (a) prediction accuracy, (b) perplexity, (c) coherence, and (d) diversity. See definitions of these metrics in Appendix C.3

.

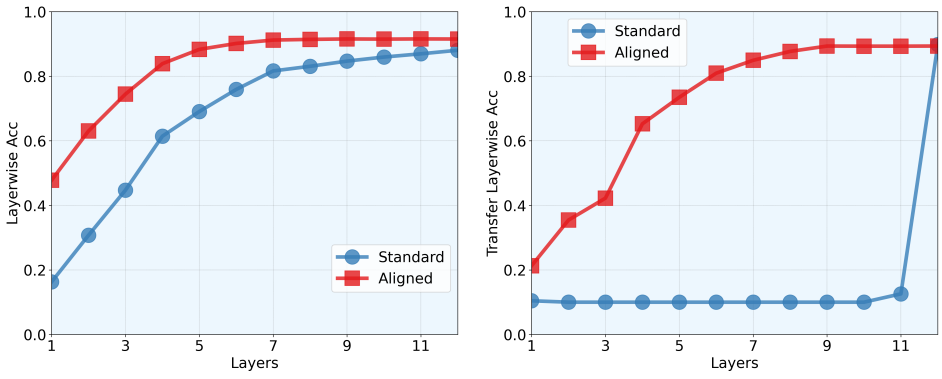
AlignedGPT: text generation tasks We then study open-ended text generation due to its widespread applicability in various areas. In formal terms, given a human-written prefix or context $\boldsymbol{x}$, the task involves decoding a continuation from the language model using the auto-regressive approach that predicts one token each time. The prediction of the next token is the same to a classification problem: given current tokens $\boldsymbol{x}$ as input, the transformer makes the prediction based on the feature $\boldsymbol{h}^L$ of the last token. The transformer is also trained with CE loss where the expected next token is the label and the last-layer linear classifier (generation head) $\boldsymbol{W}$ represents the embeddings for all the possible tokens. To improve representation similarity across layers so that features from early layers can also predict the next token, we use aligned training to finetune the auto-regressive model as in (1) that appends CE loss across shallow layers.

For both the standard model and the model fine-tuned with aligned training, we can generate text from the intermediate layers by extracting hidden states from these layers and passing them to the last-layer classifier to get the next token logits. These logits are then used by certain decoding methods to generate the new token. Following (20; 19), we evaluate the generated text from two perspectives: (1) language modeling quality, which assesses the intrinsic quality of the model and is measured by prediction accuracy and perplexity, and (2) generation quality, which measures the quality of the text produced by the model using coherence and diversity. Coherence is a measurement of relevance between prefix text and generated text, while diversity considers the recurrence of generation at varying n-gram levels. See the Appendix for the details.

In this experiment, we fine-tune GPT-2 on the Wikitext-103 dataset and show the results in Figure 7. In terms of model quality, AlignedGPT outperforms the standard GPT-2 in prediction accuracy and exhibits lower perplexity across intermediate layers, excluding the last layer where the two models achieve comparable performance. Regarding the quality of text generated from intermediate layers, AlignedGPT also excels in maintaining higher coherence and diversity. This suggests that we can utilize the shallower layers for text generation to improve inference efficiency without significantly reducing the quality of the generated text.

Effects on transferability It is often claimed that shallow layers learn universal patterns while deep layers fit to class labels. Questions arise about whether the proposed aligned training approach is that aligning shallow layer features with deep layer features could cause the shallow layers to lose their transferability. To resolve this question, we conduct two sets of experiments:

- **Distribution shift**: we first train a DeiT on CIFAR10 with standard training and align training, and then evaluate the layer-wise accuracy on CIFAR10.2 (14),
- **Transfer to different tasks**: we first train a DeiT on ImageNet with standard training and align training, and then evaluate the layer-wise accuracy on CIFAR10 by *only* fine-tune a linear classifier, with the feature mapping fixed.



(a) CIFAR10 to CIFAR10.2 (b) ImageNet to CIFAR10

Figure 8: The comparison of layer-wise accuracy between a standard model and an aligned model.

The results are plotted in Figure 8. We observe that for both cases, the distribution shift and transferring to different tasks, layer-wise accuracy curves resemble those on the pre-trained datasets shown in Figure 3 and Figure 2, demonstrating that aligned training not only improves layer-wise accuracy for the pre-trained datasets but also for the downstream datasets. In other words, the aligned training methods maintain transferability, ensuring that the trained model can be effectively transferred.

Appendix C. Additional experiments

In this section, we first describe more details about the datasets and the computational resource used in the paper. Particularly, CIFAR10, CIFAR10.2, ImageNet1K, GLUE, and Wikitext-103 are publicly available for academic purpose under the MIT license. For experiments on vision tasks, we run all experiments on 4 RTX A5000 GPUs with 24GB memory. For experiments on NLP tasks, we run all experiments on single RTX A5000 GPU with 24G memory.

Implementation details for Vision Experiments. We conduct experiments on both the CIFAR10 and ImageNet1K datasets. The CIFAR10 dataset includes 60,000 color images in 10 classes, each measuring 32×32 pixels. ImageNet1K contains 1.2 million color images distributed in 1000 classes. To increase the diversity of our training data, we use a data augmentation strategy. This includes random crop and padding, random horizontal flip with a probability of 0.5, and random rotation within 15 degrees. For optimization, we employ AdamW with an initial learning rate of 0.1. This rate decays according to the MultiStepLR at the 100th and 150th epochs, over a total of 200 epochs. We set the weight decay at 1e-4. The global batch size for both datasets is set at 256.

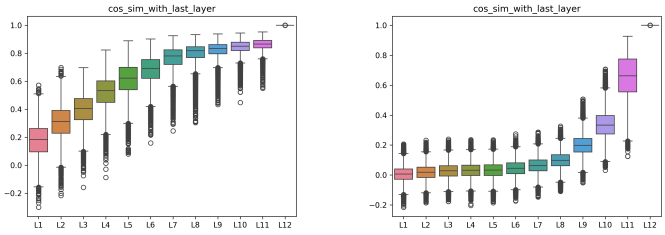
Implementation details for NLP Experiments. The General Language Understanding Evaluation (GLUE) benchmark comprises nine tasks for assessing natural language understanding. In our AlignedBERT experiments on the GLUE dataset, we used a sequence length of 256. We employed AdamW for optimization with an initial learning rate of 2e-5, and a batch size of 32. Each task underwent fine-tuning for three epochs. The WikiText-103 language modeling dataset consists of over 100 million tokens extracted from Wikipedia's verified Good and Featured articles. For AlignedGPT experiments on the WikiText-103 dataset, we maintained the sequence length at 256 and used AdamW with an initial learning rate of 2e-5. In this case, we set the batch size to 8.

C.1. Box Plots of Sample-wise Cosine Similarity

There may be some rare samples with negative sample-wise cosine similarity between features from layers that are far apart.

C.2. Residual Connections Eliminate Rotation Ambiguity

Section 2 demonstrates a consistent trend between COS and CKA. Additionally, when we compute the cosine similarity of features from adjacent layers in Figure 10, most samples exhibit high similarity. These findings suggest that Transformers do not have orthogonal



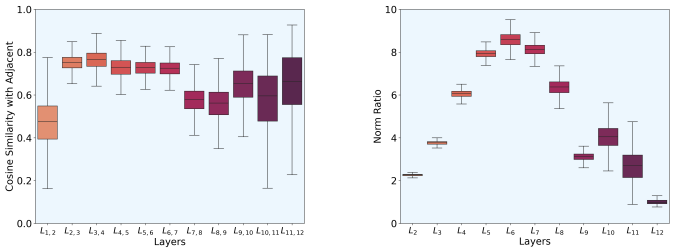
(*a*) CIFAR10 (*b*) IN1K

Figure 9: Sample-wise cosine similarity of features from shallow layers and the last-hidden layer. The DeiT-S model is trained with standard training on CIFAR-10 and ImageNet. It shows there are rare samples with negative sample-wise cosine similarity.

transformations across layers. But why does this occur? In this section, we examine the role of skip connections in preventing orthogonal transformations.

Most transformer architectures include skip connections, which are added after the (i) self-attention layer and (ii) MLP layer. By combining (**??**) as a long branch part with the identity part, we obtain:

$$\begin{aligned} \boldsymbol{h}^{\ell+1} &= \mathrm{MLP}(\mathrm{LN}(\mathrm{MSA}(\mathrm{LN}(\boldsymbol{h}^{\ell}) + \boldsymbol{h}^{\ell})) + \mathrm{MSA}(\mathrm{LN}(\boldsymbol{h}^{\ell})) + \boldsymbol{h}^{\ell} \\ &= f(\boldsymbol{h}^{\ell}) + \boldsymbol{h}^{\ell} \end{aligned}$$

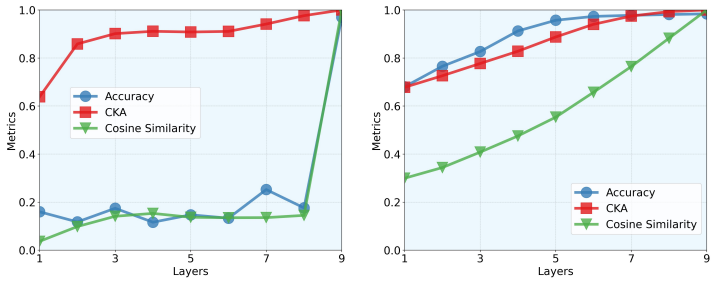


(*a*) COS of adjacent layers (*b*) Norm Ratio

Figure 10: Cosine similarity of features from adjacent layers $\mathrm{COS}(\boldsymbol{h}^{\ell-1}, \boldsymbol{h}^{\ell})$ and norm ratios $\|\boldsymbol{h}^{\ell}\|/||f(\boldsymbol{h}^{\ell})||$ distributions. The DeiT-Small model is trained on Imagenet-1K and evaluated on its validation dataset.

To investigate the effect of residual connections, we calculate the norm ratio $\|\boldsymbol{h}^{\ell}\|/||f(\boldsymbol{h}^{\ell})||$. Here, $\boldsymbol{h}^{\ell}$ represents the hidden output from the (ℓ)-th layer using the skip connection, and $f(\boldsymbol{h}^{\ell})$ is the transformation of $\boldsymbol{h}^{\ell}$ from the long branch. The results are displayed in Figure 10. High norm ratios suggest that skip connections significantly influence the representational structure of ViT.

To provide further evidence that residual connections resolve the rotation ambiguity, we compared the MLP model with and without these connections and computed their



(a) Without residual connections (b) With residual connections

Figure 11: Comparison of layerwise accuracy, COS(COsine Similarity), and CKA (Centered Kernel Alignment) with the last layer of the 9-layer MLP models with and without residual connection on the MNIST validation dataset. The models are trained from scratch using standard training. In the left figure, CKA fails to accurately reflect the change in layerwise accuracy for the MLP without residual connection. In the right figure, the presence of a residual connection is the reason why CKA works well, as it helps eliminate rotation ambiguity.

COS and CKA values. For the MLP model without residual connections, as shown in Figure 11(a), the CKA value is not consistent with accuracy and cosine similarity. A high CKA value might indicate significant similarity between features across layers, but it does not necessarily correlate with high classification accuracy. This inconsistency primarily results from the fact that CKA does not account for rotation in the feature space, suggesting that features could rotate without the residual connections. In contrast, for the MLP model with residual connections, as depicted in Figure 11(b), the CKA value aligns with layerwise accuracy, indicating that residual connections effectively eliminate the rotation ambiguity of features.

C.3. Aligned Training

Illustration of Train Once and Fit all devices. Figure 12 illustrate how aligned training support train once and fit all devices. After aligned training, one can directly fetch from shallow to deep layers of transformer according to the device computational resources and memory constrains.

Alternative Approach for Enhancing Layer-wise Representation Similarity. In addition to using aligned training loss to enhance similarity, another method is to add the cosine similarity as a regularization term to the loss function.

$$\mathcal{L}_{\text{sim}}(\boldsymbol{h}^{\ell}, \boldsymbol{h}^{L}) = \sum_{l=1}^{L} \lambda_{\ell}(1 - \cos(\boldsymbol{h}^{\ell}, \boldsymbol{h}^{L}))$$

And the total loss is the sum of this two term:

$$\mathcal{L}_{\text{CE-reg}}(\boldsymbol{x}, y) = \mathcal{L}_{\text{CE}}(\mathcal{C}(\boldsymbol{h}^{L}), y) + \beta \mathcal{L}_{\text{sim}}(\boldsymbol{h}^{\ell}, \boldsymbol{h}^{L})$$

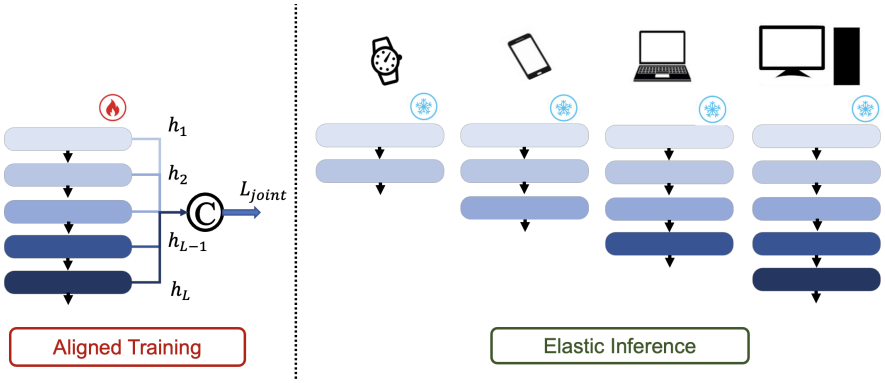
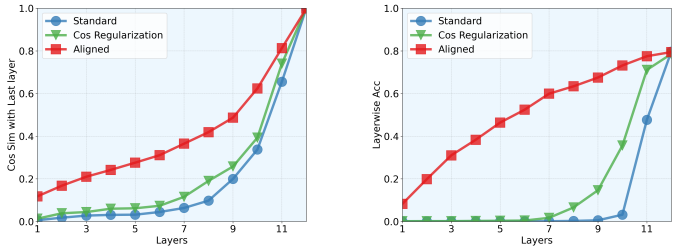


Figure 12: Aligned training of transformer using joint CE loss of all layer features with common classifier and elastic inference for different memory constrains. Once the model is trained using the aligned method, it can fit all devices. Features from darker layers indicate better performance.



(*a*) Cos Similarity (*b*) Layerwise Acc

Figure 13: Cosine similarity with last layer and layerwise accuracy of standard training using $\mathcal{L}_{\text{CE}}(\mathcal{C}(\boldsymbol{h}_\ell), y))$, standard-Reg training using $\mathcal{L}_{\text{CE-reg}}(\boldsymbol{x}, y)$ and Aligned training using $\mathcal{L}_{\text{aligned}}(\boldsymbol{x}, y)$. The 12 layers DeiT model is trained on ImageNet1K dataset. Regularization term helps little for improving the cosine similarity and layerwise accuracy. But our aligned training improves both a lot.

where $\beta > 0$ is the regularization coefficient. According to Figure 13, the regularization term contributes minimally to the improvement of cosine similarity and layer-wise accuracy, compared to aligned training methods.

Using the CE-reg loss results in poor layerwise accuracy and lower cosine similarity compared to the aligned loss. The likely reason for this is an imbalance between the COS alignment objective and the primary classification objective. Our intuition is that directly optimizing for high COS alignment may fail because the COS alignment loss primarily focuses on aligning features across layers, without necessarily making the features discriminative enough for the classification task. In contrast, the cross-entropy (CE) loss directly optimizes for classification, and as a consequence, it naturally improves COS alignment. This suggests that while COS alignment is important, it may not be sufficient on its own without the robust guidance provided by the CE loss.

Another approach involves adding the CKA term as a regularization term to the CE loss. However, this approach may not be effective and has several drawbacks. First, CKA is not always reliable in representing layer-wise similarity across all settings, particularly in scenarios with rotation ambiguity or when residual connections are absent. Second, it is computationally expensive, as it requires computing the Gram matrix to evaluate relationships between features. Lastly, CKA might not perform better than the COS regularization term and may yield similar results, falling short compared to the aligned loss. In transformers, both COS and CKA measure feature similarity and tend to exhibit similar trends. As shown in Figure 13, the COS regularization term contributes minimally to improving cosine similarity and layer-wise accuracy. Based on this, we infer that using CKA as a regularization term would similarly have a minimal impact on enhancing these metrics. Therefore, aligned training approaches may be more effective than relying solely on regularization terms.

Setup for Aligned Training. It's reported (29) that training only with this aligned loss would cause the performance drop in the last layer. So following (29), we choose the "alternating" training approach, which alternates objectives based on the iteration number. During odd-numbered iterations, we use the CE loss of the final layer $\mathcal{L}_{\text{CE}}(\mathcal{C}(\boldsymbol{h}_L), y)$. For even-numbered iterations, the strategy involves using the aligned loss $\mathcal{L}_{\text{aligned}}(\boldsymbol{x}, y)$.

Note that this training strategy, which uses a common classifier, no longer requires the KL-divergence term that is commonly used in mutli-exit/classifeirs training. This is because the deep layers have been trained to capture the abstract and discriminative features of the input data, effectively serving as the teacher model. The KL-divergence term is typically used to guide the shallower layers. However, when we use a common classifier, our aligned training method becomes a latent knowledge self-distillation method. The shallow layers can mimic or align their feature representations with those of the deep layers by aligning with the common classifier. As such, the deep layers, with their advanced feature representations, act as the teachers, while the shallow layers, in their quest to improve their feature extraction capabilities, assume the role of students. Therefore, the KL-divergence term is no longer necessary.

Setup in AlignGPT. Our aligned training method can be used with any transformer-based language models. In this study, we evaluated our method using the GPT-2 model. We finetune the GPT2 models using aligned training methods and then use intermediate layers of GPT2 to generate the texts.

- **Model and Baselines** We finetune GPT-2 on the Wikitext-103 dataset with the proposed objective $\mathcal{L}_{\text{aligned}}$ for $40k$ training steps and generate the text continuation with nucleus sampling (11) with $p = 0.95$ decoding methods. For the standard baseline model, we finetune the model with CE loss $\mathcal{L}_{\text{MLE}}$. The model is finetuned using a single 24G RTX A5000 GPU for 70 hours.
- **Evaluations** Following (19), we evaluate the model from two perspectives: (1) language modeling quality, assessing the inherent quality of the model, and (2) generation quality, measuring the quality of the text the model produces. In assessing language modeling quality, we calculate the prediction accuracy and perplexity of each layer. When evaluating generation quality, we measure the similarity between the prompt text and generated

text using coherence. We employ generation repetition to gauge the diversity of the generated text. The metrics are defined as follows

– **Prediction Accuracy** The accuracy is computed on the Wikitext-103 test set as,

$$\textbf{Acc} = \frac{1}{D}\sum_{i=1}^{D}\sum_{i=1}^{n}\mathbb{1}[\arg\max p_\theta(x|\boldsymbol{x}_{<i}) = x_i] \tag{2}$$

where the D is the number of samples in the test dataset.

– **Perplexity** The perplexity is computed on the test set of Wikitext-103. It's computed as the exponential of the test loss.

– **Coherence** Coherence measures the relevance between the prefix text and the generated text. We apply the advanced sentence embedding method, SimCSE (6), to measure the semantic coherence or consistency between the prefix and the generated text. The coherence score is defined as follows,

$$\textbf{Coherence} = \boldsymbol{h}_{\boldsymbol{x}}^T\boldsymbol{h}_{\hat{\boldsymbol{x}}}/\|\boldsymbol{h}_{\boldsymbol{x}}\|\|\boldsymbol{h}_{\hat{\boldsymbol{x}}}\| \tag{3}$$

where $\boldsymbol{x}$ is the prefix text and $\hat{\boldsymbol{x}}$ is the generated text and $\boldsymbol{h}_{\boldsymbol{x}} = \text{SimCSE}(\boldsymbol{x})$ and $\boldsymbol{h}_{\hat{\boldsymbol{x}}} = \text{SimCSE}(\hat{\boldsymbol{x}})$. Higher coherence means more correlation to the given prompt.

– **Diversity** Diversity measures the occurrence of generation at different n-gram levels. It is defined as:

$$\textbf{Diversity} = \prod_{n=2}^{4}\frac{|\text{unique n-grams}(\hat{\boldsymbol{x}})|}{\text{—total n-grams}(\hat{\boldsymbol{x}})|} \tag{4}$$

A higher diversity score suggests fewer repeated words in the generated text.