AdaInf: Adaptive Inference for Resource-Constrained Foundation Models

Zhuoyan Xu [1] **Khoi Duc Nguyen** [1] **Preeti Mukherjee** [2] **Somali Chaterji** [2] **Yingyu Liang** [1 3] **Yin Li** [1 2]

Abstract

Foundation models have emerged as a powerful tool in AI, yet come with substantial computational cost, limiting their deployment in resource-constraint devices. Several recent research has been dedicated to improving the efficiency of foundation models. These prior solutions often yield models with static accuracy and latency footprint, and thus fall short in responding to potential runtime perturbations, including varying input characteristics (e.g., a static video vs. a dynamic one) or changing resource availability (e.g., contention due to other programs on the device). To bridge this gap, we introduce **AdaInf**—an adaptive inference framework that treats a foundation model as a collection of execution branches, and learns a scheduler to decide on which branch to execute, accounting for the input data and a compute budget. We demonstrate preliminary results on CIFAR and ImageNet with vision and vision-language models and across convolutional networks and Transformers. Our results show that AdaInf can achieve varying accuracy and latency trade-offs. When compared to latest method, AdaInf attains a major improvement in accuracy under a wide range of latency budgets.

1. Introduction

Foundation models have revolutionized AI in recent years, celebrating tremendous success in vision [7, 30, 6], language [5, 27, 1, 29], and multimodal learning [32, 28, 22]. These models, trained on broad data, offer the potential to adapt to a wide range of downstream tasks, leading to some of the most exciting developments and applications in AI to date. However, the high performance of these models comes at a cost of increased model capacity and computation complexity. This poses a significant challenge for deploying these models on edge devices, where efficient inference is essential and the computation budget is limited [40].

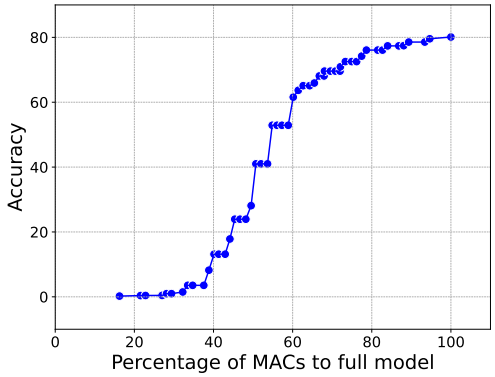


Figure 1. Multiple execution branches of ResNet50 [12] pretrained on ImageNet. Each point refers a branch. x-axis: percentage of MACs a branch uses in comparison to full model. y-axis: The model accuracy on validation set of ImageNet.

Extensive research has been dedicated to improving the efficiency of deep models, broadly applicable to foundation models. This is often achieved by optimizing the model architecture during training [42, 33], or by approximating computational procedures at inference [3, 51]. A significant drawback of these approaches is that they produce models with static accuracy and latency footprint. Such static models fall short in responding to the runtime perturbations when integrated in real-world computing systems, severely restricting their applicability. Example perturbations include varying input characteristics (e.g., a static video vs. a dynamic one) or changing resource availability (e.g., contention due to other programs on the device). Several recent works have explored adaptive inference of deep models [40, 23, 34, 35]. However, they only consider the adaptation to the input data, and can not fulfill varying latency budgets for the same input.

In departure from prior solutions, we investigate *adaptive* inference of foundation models, aiming at an inference procedure that can achieve varying accuracy and latency trade-offs in response to *both input characteristics and resource contention*. Our work builds on two key insights. *First*, our basic premise is that existing large pretrained models has built-in redundancy. This is because modern training techniques for deep models adopt aggressive regularization to

[1]University of Wisconsin-Madison, [2]Purdue University, [3]The University of Hong Kong. Correspondence to: Yin Li <yin.li@wisc.edu>.

Work presented at the ES-FoMo-II Workshop at ICML 2024.

ensure generalization, in which input data is simplified (e.g., data augmentation [41]), model components are dropped (e.g., stochastic depth [16, 49, 43, 44]), and redundancy within components is encouraged (e.g., multiple redundant attention head [24]). This redundancy allows us to treat a model as a collection of execution branches. *Second*, our key intuition is that these execution branches can be tailored for runtime conditions, thereby achieving adaptive inference. For example, each branch can have a different latency budget, and under this budget it can be tasked to specialize in regions of the input space, as shown in Figure 1.

To this end, we present an adaptive inference framework dubbed **AdaInf**. AdaInf treats a foundation model as a collection of execution branches. It further learns a scheduler to decide on the branch to execute, based on a compute budget in terms of Multiply-Accumulate Operations (MACs), as well as the input data. Specifically, we train a lightweight scheduler that accounts for the latency budget and input data, and predicts the best execution branch that meets the budget while likely achieving a high accuracy. Importantly, we adapt the base foundation model using parameter efficient fine-tuning [14], in tandem with the learning of the scheduler. In doing so, we ensure the compatibility between the foundation model and the scheduler, without the need of updating the base model weights. In addition to this content-aware scheduler, we also consider a content-agnostic baseline.

We conduct preliminary experiments on CIFAR and ImageNet using pre-trained ResNet and CLIP models. Our results shows that AdaInf can achieve varying accuracy and latency trade-offs in response to the input data and the latency budget. Further, when compared to latest method, AdaInf attains a major improvement in accuracy under a wide range of latency budgets.

2. Related Work

Several adaptive inference methods have been proposed to dynamically allocate model components during inference, aiming to improve both efficiency and accuracy. For convolutional networks, methods have been developed to skip layers during the inference [9, 19, 45, 2, 50]. For vision transformers, various approaches have been proposed to enhance efficiency, such as selecting different patches of images [46, 33, 31], and using different attention heads and blocks [23]. Additionally, similar ideas have been explored for language models, where models actively select tokens during inference [34]. While these methods provide algorithms for efficient inference by using a subset of model components, they are limited by their design to select a single execution plan per input without considering the compute budget. This design results in fixed accuracy and MACs for each input and cannot adapt to varying compute budget for the same input. In contrast, our work considers the compute budget as MACs limits, and develops a framework that predicts feasible plans under varying budgets for each input.

3. Adaptive Inference for Foundation Models

We formally define our problem of adaptive inference. We assume that a pre-trained foundation model can be decomposed into a collection of execution branches. While in this paper we focus on branches formed by skipping some of the layers within the model, this concept can be easily extended to other operations including resizing an input image or dropping certain attention heads in a Transformer block. Given a sample input and a MACs requirement denoting the compute budget, the goal of adaptive inference is design a scheduler to select a execution branch of the model, such that the inference compute cost falls below the budget, and the expected prediction performance is maximized.

Let f_θ be the foundation model in consideration, and $g_\beta(\cdot,\cdot)$ a light-weighted scheduler. Given a input sample (x, y) and MACs requirement $M \in \mathbb{R}$, $g_\beta(\cdot,\cdot)$ will output the execution plan for the forward pass of foundation model f, as $p = g_\beta(x, M)$. In this case, p can indicate the skipping of intermediate layers within the model. With minor abuse of the symbols, we denote the prediction outputted by executing p on f_θ as $\widehat{f}(x, p)$, and resulting inference MACs as $\widehat{M}(x, p)$.

We propose **AdaInf**, a framework that adaptively determines the execution plan to skip or retain certain layers during inference. **AdaInf** aims at maximizing the prediction performance of $\widehat{f}(x, p)$, while ensuring the computing budget (i.e., $\widehat{M}(x, p) \leq M$). To this end, we consider the following loss function for training.

$$\mathcal{L} = \mathcal{L}_{\text{CE}}(y, \widehat{f}(x, p)) + \lambda \mathcal{L}_{macs}(\widehat{M}, M), \qquad (1)$$

where $\mathcal{L}_{\text{CE}}$ is the standard cross entropy loss to maximize prediction performance, $\mathcal{L}_{macs}$ denotes the hinge loss (i.e., $\mathcal{L}_{macs}(\widehat{M}, M) = \max\{0, \widehat{M}(x, p) - M\}$.) to constraint the computing budget, and λ is a coefficient balancing the two loss terms. We set $\lambda = 1$ in our experiments.

Our key design choice is to employ the loss function for learning the scheduler and adapting the foundation model with parameter-efficient fine-tuning (e.g., LoRA [15]). This design allows us to use any pre-trained foundation model checkpoint,[1] while incorporating a scheduler and an adapter that are trained for adaptive inference, without the need of altering the weights in the original model.

In what follows, we describe our design of execution branches, and present the learning of the scheduler.

[1]The pre-trained model may be trained with or without using the stochastic depth technique.

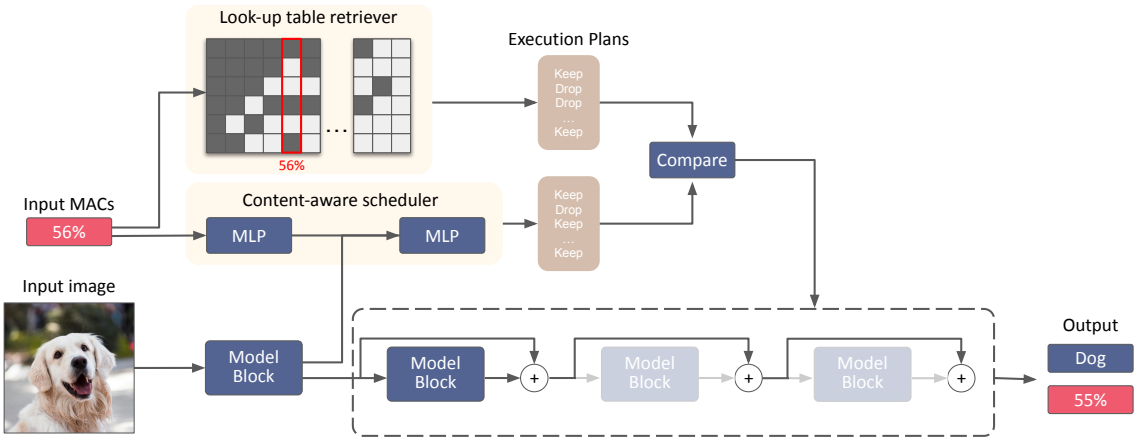


Figure 2. The **AdaInf** framework, which consists of a look-up-table retriever, a scheduler and foundation model. Once receiving MACs requirement l, look-up-table retriever identifies the best model branch under requirement and output the plan. In the meantime, the sample input will feed into first block of main recognition model to get embedding, the embedding will be part of the input to the scheduler, the scheduler takes embedding of both image input and MACs input, combine them together and predict the execution plan. The framework will compare the plan by retriver and scheduler, choose the best one and guide the forward pass of rest of foundation model.

Design of Execution Branches. Most current foundation models feature backbones with residual connections. For simplicity, we consider a foundation model f_θ consisting of N blocks, each linked by a residual connection. For a model block denoted as $\mathcal{F}$ with input x and output y, the output is $y = x + \mathcal{F}(x)$. We consider skipping some of these blocks. To skip a block, we set the block forward as $y = x$. As shown in Figure 1, skipping certain blocks during inference does not lead to significant accuracy drops. To make the input compatible, the first block is always retained, while the remaining blocks can be dropped by the scheduler.

Scheduler Learning We consider two types of schedulers: (1) a content-agostic one implemented using a lookup table; and (2) a content-aware one realized using a deep network.

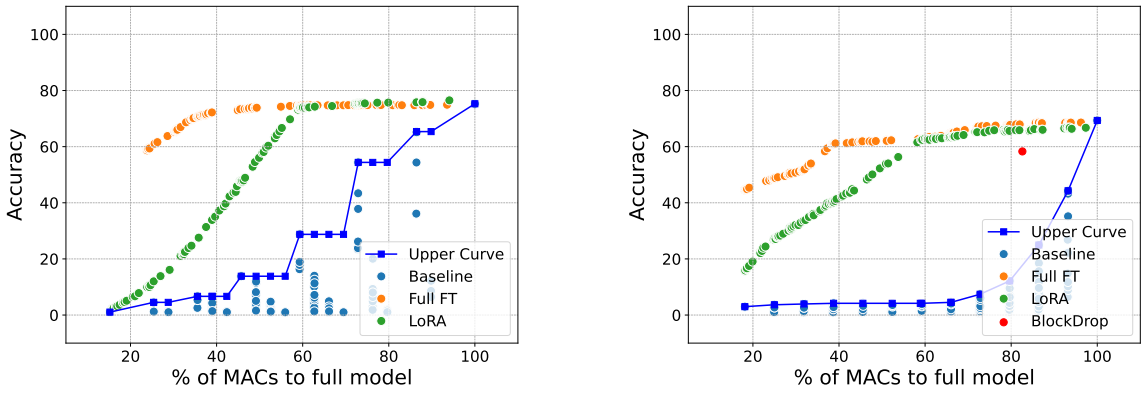
• *Content-agnostic Scheduler*. This approach selects execution branches solely based on compute budget without considering the input. In this case, we construct a lookup table for the scheduler. Specifically, we enumerate all branches (2^{N-1} in total). We evaluate all branches' accuracy on the validation set, recording their accuracy and associated MACs. We select the best branch in each MACs stage and saved them in a lookup table (as number of branches N goes larger, it's hard to enumerate all 2^{N-1} branches, we would random select a subset of $M = 128$ branches). At inference time, given a MACs requirement input, this content-agnostic scheduler selects a branch in the lookup table that has (1) a compute budget satisfying the MACs requirement, and (2) the highest recorded accuracy on the validation set. This process is illustrated as a *look-up-table retriever* in Figure 2. While this approach is simple and efficient, it fails to consider the input characteristics.

• *Content-aware Scheduler* To allow adaptive selection of different execution branches tailored to individual inputs, we further consider learning a content-aware scheduler. This scheduler takes both the input sample and the compute budget, and outputs an execution plan (i.e., a selected execution branch). Specifically, the input sample x passes through the first block of the model to obtain embeddings, while the MACs input M is processed by a simple MLP. The scheduler then receives the two embeddings, concatenates them, and outputs the probability of retaining each block $p = g(x, M)$, where the output $p \in \mathbb{R}^{N-1}$ with N being the total number of blocks in the foundational model f, each entry is a binary value deciding whether to keep or drop certain block. This pipeline is illustrated in Figure 2. For training the scheduler, we use Gumbel-Softmax trick to allow the back-propagation of the gradients. We refer reader to Appendix B for full experimental details.

4. Experiments

Experimental Setup. We experiment with ResNet18 and ResNet32 to demonstrate the idea and then expand our setting into CLIP models [32] as an exemplary foundation model. We pretrained ResNet on CIFAR100. We use a pretrained CLIP model checkpoint from OpenCLIP [17], which was trained on LAION-400m [36]. Within our AdaInf framework, we treat the vision encoder of the CLIP models as the recognition model. We finetune the vision transformer in CLIP while keeping the other components unchanged.

We test our pipeline on CIFAR and ImageNet, finetuning the model and training the scheduler on the training set, and evaluating them on the validation set, respectively. Initially,



(a) Results of ResNet18 on CIFAR100.

(b) Results of ResNet32 on CIFAR100.

Figure 3. Results on ResNet pretrained on CIFAR100. Baseline: Look-up-table baseline. Upper Curve: The upper curve of the baseline. Full FT: Results on fully finetune the ResNet. LoRA: LoRA finetune on ResNet. BlockDrop: results in [50].

we construct the Look-up Table as described in Section 3. We train our content-aware scheduler and foundation model using the LoRA technique [14]. MACs are normalized to percentages relative to the full model's capacity. During the training phase, for each sample in the batch, we randomly sample MACs uniformly from 0 to 1 as the MACs input. We update the foundation model and content-aware scheduler based on the loss described in (1), using the Adam optimizer with a learning rate of 1e-3 and CosineAnnealing decay.

In the evaluation phase, we uniformly sample $S = 128$ MACs percentages from 0 to 1 as settings of our computing budgets. These 128 settings are separately evaluated. For each setting, we input the MACs percentage and perform inference on all images from the validation set. We then compute the average accuracy and corresponding MACs for each setting, and plot their trade-offs.

Results. Figure 3 displays the performance of ResNet on CIFAR100. We compare the outcomes using our trained content-aware scheduler against those obtained from a look-up-table baseline. Our framework consistently achieves performance improvements over the baseline across various MACs requirements. For example, with ResNet18, LoRA finetuning in conjunction with our trained scheduler attains an accuracy comparable to that of the full model while utilizing only 60% of the MACs. Full finetuning reaches the same accuracy level using just 40% of the full MACs. To ensure a fair comparison with related work, we include the results of the BlockDrop study [50] in Figure 3. Since they trained a single model without considering MACs input, their results are represented as a single point in our findings. Our results show a significant improvement over theirs. Figure 4 illustrates the results from the LoRA-finetuned OpenCLIP model on ImageNet, where we observe similar performance gains. On average, our finetuned models show a 20% improvement in performance compared to the baseline.

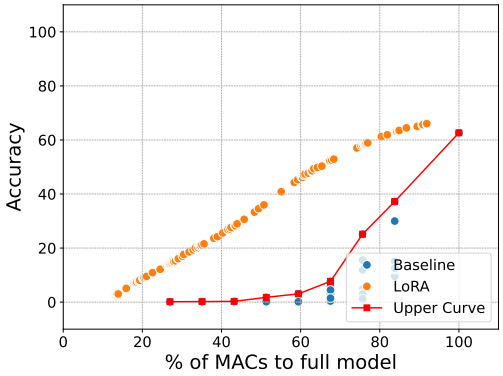


Figure 4. Results on ViT encoder of CLIP pretrained on LAION-400m. Baseline: Look-up-table baseline constructed in Section 3. Upper Curve: The upper curve of the baseline.

5. Conclusion

In this study, we present AdaInf, an adaptive inference framework that considers execution branches within a pre-trained model, and dynamically selects different branch based on the input sample and a latency budget during inference. The heart of AdaInf lies in the learning of a scheduler to decide on which branch to execute, aiming at maximizing the accuracy while enforcing the latency budget. Through experiments on CIFAR and ImageNet with vision and vision-language models, we demonstrate that our method attains an average accuracy improvement of 20% over the best average model baseline under the specified latency budget (in terms of MACs). Our method also outperform latest approaches under the same MACs budget, offering a more flexible framework. Admittedly, our work is at an early stage and we will further investigate. We consider that our preliminary results is worth reporting and will provide useful insight to the adaptive inference of foundation models.

Impact Statement

Our work aims to improve the efficiency of foundation models in handling budget limited tasks. We foresee no immediate negative ethical impact. We illustrate the empirical results of adapting foundation models under different computing budgets. We hope our work will facilitate practical deployment of foundation models, while offering better understanding of these models.

References

[1] Anthropic. The claude 3 model family: Opus, sonnet, haiku. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf, 2024.

[2] Bengio, E., Bacon, P.-L., Pineau, J., and Precup, D. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

[3] Bolya, D., Fu, C.-Y., Dai, X., Zhang, P., Feichtenhofer, C., and Hoffman, J. Token merging: Your vit but faster. In *The Eleventh International Conference on Learning Representations*, 2022.

[4] Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

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

[6] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 2020.

[7] Chen, X., Xie, S., and He, K. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[8] Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[9] Figurnov, M., Collins, M. D., Zhu, Y., Zhang, L., Huang, J., Vetrov, D., and Salakhutdinov, R. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1039–1048, 2017.

[10] Gao, T., Fisch, A., and Chen, D. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

[11] Gu, S., Levine, S., Sutskever, I., and Mnih, A. Muprop: Unbiased backpropagation for stochastic neural networks. *arXiv preprint arXiv:1511.05176*, 2015.

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

[13] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[14] Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[15] Hu, E. J., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2023.

[16] Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 646–661. Springer, 2016.

[17] Ilharco, G., Wortsman, M., Wightman, R., Gordon, C., Carlini, N., Taori, R., Dave, A., Shankar, V., Namkoong, H., Miller, J., Hajishirzi, H., Farhadi, A., and Schmidt, L. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

[18] Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

[19] Li, H., Wu, Z., Shrivastava, A., and Davis, L. S. 2d or not 2d? adaptive 3d convolution selection for efficient video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6155–6164, 2021.

[20] Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 2021.

[21] Lin, B. Y., Tan, K., Miller, C. S., Tian, B., and Ren, X. Unsupervised cross-task generalization via retrieval augmentation. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=kB9jrZDenff.

[22] Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning, 2023.

[23] Meng, L., Li, H., Chen, B.-C., Lan, S., Wu, Z., Jiang, Y.-G., and Lim, S.-N. Adavit: Adaptive vision transformers for efficient image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12309–12318, 2022.

[24] Michel, P., Levy, O., and Neubig, G. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

[25] Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022.

[26] Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[27] OpenAI. Introducing ChatGPT. https://openai.com/blog/chatgpt, 2023. Accessed: 2023-09-10.

[28] OpenAI. Gpt-4v(ision) system card. https://cdn.openai.com/papers/GPTV_System_Card.pdf, 2023.

[29] OpenAI. GPT-4 technical report. *arXiv preprint arxiv:2303.08774*, 2023.

[30] Oquab, M., Darcet, T., Moutakanni, T., Vo, H. V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Howes, R., Huang, P.-Y., Xu, H., Sharma, V., Li, S.-W., Galuba, W., Rabbat, M., Assran, M., Ballas, N., Synnaeve, G., Misra, I., Jegou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P. Dinov2: Learning robust visual features without supervision. *arXiv:2304.07193*, 2023.

[31] Pan, B., Panda, R., Jiang, Y., Wang, Z., Feris, R., and Oliva, A. Ia-red2: Interpretability-aware redundancy reduction for vision transformers. *Advances in Neural Information Processing Systems*, 34:24898–24911, 2021.

[32] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 2021.

[33] Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., and Hsieh, C.-J. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=jB0Nlbwlybm.

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

[35] Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Susano Pinto, A., Keysers, D., and Houlsby, N. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

[36] Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., and Komatsuzaki, A. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

[37] Shi, Z., Ming, Y., Fan, Y., Sala, F., and Liang, Y. Domain generalization via nuclear norm regularization. In *Conference on Parsimony and Learning (Proceedings Track)*, 2023.

[38] Shi, Z., Wei, J., Xu, Z., and Liang, Y. Why larger language models do in-context learning differently? In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023. URL https://openreview.net/forum?id=2J8xnFLMgF.

[39] Shi, Z., Wei, J., Xu, Z., and Liang, Y. Why larger language models do in-context learning differently? In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=WOa96EG26M.

[40] Song, Y., Mi, Z., Xie, H., and Chen, H. Powerinfer: Fast large language model serving with a consumer-grade gpu. *arXiv preprint arXiv:2312.12456*, 2023.

[41] Steiner, A. P., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J., and Beyer, L. How to train your vit? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=4nPswr1KcP.

[42] Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

[43] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

[44] Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., and Jégou, H. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 32–42, 2021.

[45] Wang, X., Yu, F., Dou, Z.-Y., Darrell, T., and Gonzalez, J. E. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 409–424, 2018.

[46] Wang, Y., Huang, R., Song, S., Huang, Z., and Huang, G. Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=M0J1c3PqwKZ.

[47] Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

[48] Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[49] Wightman, R., Touvron, H., and Jégou, H. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021.

[50] Wu, Z., Nagarajan, T., Kumar, A., Rennie, S., Davis, L. S., Grauman, K., and Feris, R. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8817–8826, 2018.

[51] Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., and Singh, V. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 14138–14148, 2021.

[52] Xu, Z., Shi, Z., Wei, J., Li, Y., and Liang, Y. Improving foundation models for few-shot learning via multitask finetuning. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023. URL https://openreview.net/forum?id=szNb8Hp3d3.

[53] Xu, Z., Shi, Z., and Liang, Y. Do large language models have compositional ability? an investigation into limitations and scalability. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=4XPeF0SbJs.

[54] Xu, Z., Shi, Z., Wei, J., Mu, F., Li, Y., and Liang, Y. Towards few-shot adaptation of foundation models via multitask finetuning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=1jbh2e0b2K.

A. Full Related Work

Foundation models [4] are generally trained using self-supervised learning [6, 13, 26] on extensive datasets. Adapting foundation models to downstream tasks has recently received significant attention. In the vision domain, the standard practice involves learning a simple function, such as linear probing, on the representation from a foundation model, while keeping the model frozen or performing minimal fine-tuning [6, 13, 52, 37, 54]. In NLP, prompt-based fine-tuning [10, 14] has become prevalent, where a prediction task is transformed into a masked language modeling problem during fine-tuning. Instruction tuning [47, 21, 8] has emerged as a way to enhance language models' ability to follow natural language instructions, including prompts, examples, and constraints. This approach aims to improve multi-task learning on training tasks and generalization to unseen tasks. With the advances in large language models, parameter-efficient tuning has emerged as an attractive solution. Prompt tuning [18, 20] learns an extra prompt token for a new task, while updating minimal or no parameters in the model backbone. Another promising approach is in-context learning [25, 48, 38, 53, 39], where the model is tasked to make predictions based on contexts supplemented with a few examples, with no parameter updates. Our paper focuses on adapting foundation models to new tasks under computational constraints. We propose a framework that generates viable plans for various MAC (Multiply-Accumulate Operations) budgets for each input sample, aiming to minimize performance degradation.

B. Optimization

In this section we provide the details in our training pipeline.

During training phase, the training for scheduler will only occur when content-aware pipeline is executed, since look-up-table involves no parameter update for the scheduler.

One optimization challenge is the output from last MLP layer of content-aware scheduler is probability of keeping each block. In forward pass, the we perform sampling process from probability to binary decision vector p. The sampling process is non-differentiable, preventing the computation of gradients beyond this point. A common workaround involves using a score function estimator [11, 50]; however, this method often suffers from high variance and slow convergence. Instead, we employ the reparameterization method, specifically the Gumbel-Max trick, to draw samples from the probabilities while keeping the process differentiable. Given probability $p \in \mathbb{R}^{N-1}$, we apply Gumbel-Max trick to each element of p independently, treating each entry as probability of Bernoulli random variable. To get p from probability, we have

$$b_i = \frac{\exp\left(\left(\log\left(p_i\right) + g_1\right)/\tau\right)}{\exp\left(\left(\log\left(1 - p_i\right) + g_2\right)/\tau\right) + \exp\left(\left(\log\left(p_i\right) + g_1\right)/\tau\right)}$$

for $i = 1, 2, \ldots, N-1$, where $g_1, g_2 \overset{iid}{\sim} \text{Gumbel}(0, 1)$. The $\text{Gumbel}(0, 1)$ distribution can be sampled using inverse transform sampling by drawing $u \sim \text{Unif}(0, 1)$ and computing $g = -\log(-\log(u))$. τ here is the tunable temperature parameter that affects the smoothness of the sampling process.

This technique allows for the sampling of discrete distributions while maintaining differentiability of the process, facilitating gradient-based optimization even in the presence of discrete decision variables in the model.