"Deep Incubation: Training Large Models by Divide-and-Conquering" Supplementary Material

1. Adopting Advanced Recipes

In this section, we delve into the complementarity of our approach with advanced training recipes. We employ the highly-optimized DeiT-III training recipe [4] as an example and selected two prominent model architectures: ViT-B [1] and Swin-B [2]. The comparative outcomes are presented in Tab. 1.

model	#params	FLOPs	orig.	DeiT III	ours + DeiT III*
ViT-B	87M	17.6G	81.8	83.1	**83.3**
Swin-B	88M	15.4G	83.5	82.2	**83.8**

Table 1: Results on ImageNet-1K with advanced recipe.

Note that the DeiT-III paper [4] introduces numerous refinements to the standard DeiT training protocol. These enhancements encompass the adoption of the LAMB optimizer, the use of binary cross-entropy loss, adjustments to the stochastic depth rate, the introduction of ThreeAugment, extension of training epochs (400 epochs for example), and several other modifications. We find that not all techniques from the original DeiT-III framework proved to be helpful when combined with our method. For instance, the binary cross-entropy loss is not found to benefit our method and is thus removed from our experiments. Hence, our approach in conjunction with the DeiT-III configuration is denoted as Ours + DeiT III*. Our preliminary findings may indicate the complementarity of our method with more advanced training strategies.

2. Decentralized Modular Training

We additionally discuss an intriguing possibility enabled by the independent nature of modular training: decentralized modular training (DMT). Typically, a deep model is trained as a whole with centralized compute and centralized data. While the modular training process can train the model with decentralized compute (*i.e.*, each module can be trained on a different machine with no communication needed), the training data is still centralized.

Thus, we extend the idea of decentralized training to the training data, enabling the modular training process to be executed in a fully decentralized manner, with both decentralized compute resources and decentralized data. Practically, we can divide the training set into several subsets, where each machine needs only to cache one subset and use it to train one module. This may be particularly useful when the machines are of limited storage capacity.

We empirically evaluate the performance of our method in this scenario. We first pre-train the meta model with the full training set. Then, we divide the training set into 4 non-overlapping subsets in a class-balanced manner, and distribute the subsets along with the pre-trained meta model to 4 machines. In this way, each machine can train one module with its cached subset. We keep the total number of training iterations unchanged during modular training. Finally, we collect the trained modules and fine-tune the assembled model on the full training set. The results are presented in Table 2 (denoted as DMT). On both ImageNet and CIFAR-100, Deep Incubation can still achieve decent performance despite that only 25% of the data is available to each machine.

method	IN-1K / ViT-B	CIFAR100 / DeiT-T-128
E2E-DeiT [3]	81.8	69.4
DeiT + Ours	**82.4**(+0.6)	**77.2**(+7.8)
DeiT + Ours (DMT)	**82.2**(+0.4)	**74.8**(+5.4)

Table 2: Decentralized modular training results. IN-1K: ImageNet-1K. Here, the training set is divided into 4 subsets, and each module is trained on one subset.

3. Limitations and Future Work

Currently, our method is primarily tailored to training computer vision models using supervised learning, limiting its applicability to other domains or other training techniques like self-supervised learning. Meanwhile, the Deep Incubation pipeline is structured in stages, with separate phases for pre-training meta models, modular training, and fine-tuning. Additionally, the system's flexibility is constrained by the necessity to adjust the meta model's width for different models. This means that a new meta model may need to be trained when dealing with a model that has a different width, increasing the complexity of the process.

Looking forward, we see several promising avenues for enhancing our method. Extending the approach to multimodal settings and embracing alternative training paradigms, such as self-supervised learning, would make our method more general and flexible. Streamlining the process by combining the different training stages as well as enabling the sharing of meta models across different architectures could make the pipeline easier. Furthermore, the wide availability of pre-trained models online offers a potential opportunity. Exploring the use of these pre-trained models as replacements for the trained-from-scratch meta models could facilitate the Deep Incubation pipeline and pave the way for exciting developments in the field.

References

[1] 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. In *ICLR*, 2020. 1

[2] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1

[3] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herv'e J'egou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 1

[4] Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *ECCV*, 2022. 1