ThinkingViT: Nested Thinking Vision Transformer for Elastic Inference

Ali Hojjat [1] **Janek Haberer** [1] **Sören Pirk** [1] **Olaf Landsiedel** [2 1]

Abstract

Vision Transformers (ViTs) deliver state-of-the-art performance, yet their fixed computational budget prevents scalable deployment across heterogeneous hardware. Recent nested Transformer architectures mitigate this by embedding nested subnetworks within a single model to enable scalable inference. However, these models allocate the same amount of compute to all inputs, regardless of their complexity, which leads to inefficiencies. To address this, we introduce *ThinkingViT*, a nested ViT architecture that employs progressive thinking stages to dynamically adjust inference computation based on input difficulty. *ThinkingViT* initiates inference by activating a small subset of the most important attention heads and terminates early if predictions reach sufficient certainty. Otherwise, it activates additional attention heads and re-evaluates the input. At the core of *ThinkingViT* is our Token Recycling mechanism, which conditions each subsequent inference stage on the embeddings from the previous stage, enabling progressive improvement. Due to its backbone-preserving design, *ThinkingViT* also serves as a plugin upgrade for vanilla ViTs. Experiments show that *ThinkingViT* surpasses nested baselines by up to 2.0 percentage points (p.p.) in accuracy at the same throughput and by up to 2.9 p.p. at equal GMACs on ImageNet-1K. The source code is available at https://github.com/ds-kiel/ThinkingViT.

[1]Department of Computer Science, Kiel University, Kiel, Germany [2]Department of Computer Science, Institute for Networked Cyber Physical Systems, Hamburg University of Technology, Hamburg, Germany. Correspondence to: Ali Hojjat <ali.hojjat@cs.uni-kiel.de>, Janek Haberer <janek.haberer@cs.uni-kiel.de>, Sören Pirk <soeren.pirk@cs.uni-kiel.de>, Olaf Landsiedel <olaf.landsiedel@tuhh.de>.

Efficient Systems for Foundation Models Workshop, International Conference on Machine Learning (ICML), Vancouver, Canada, 2025.

1. Introduction

Motivation: Vision Transformer (ViT) (Dosovitskiy et al., 2021) have achieved state-of-the-art results across numerous image recognition tasks (Liu et al., 2022; Hatamizadeh & Kautz, 2024), yet their non-elastic inference pipelines impose a uniform and often excessive computational cost. This lack of flexibility poses a significant limitation in real-world deployments, where devices vary widely in computational power and latency constraints (Xu et al., 2025). From high-throughput servers to mobile and embedded platforms, modern applications require elastic inference (Cai et al., 2024), where a model can adjust its computational footprint to match the capabilities of the target hardware. Without such adaptability, ViTs struggle to meet the performance and efficiency tradeoffs necessary for scalable and widespread deployment.

Limitations of Prior Work: Recent progress in *nested* Transformer architectures offers a compelling pathway toward elastic inference (Haberer et al., 2024; Devvrit et al., 2024; Zhang et al., 2024). These methods embed multiple nested subnetworks within a single Transformer backbone, enabling dynamic subnetwork selection at inference time. This strategy allows models to operate under varying latency and hardware constraints without any extra tuning or post-training adjustments, all while maintaining a unified parameter set. However, these approaches typically allocate a *fixed* computational budget per input, missing the opportunity to distinguish between simple and complex samples, which leads to inefficiencies, especially under tight resource constraints.

Image-Based Routing: To make nested models adaptable, we must route the input image to the appropriate subnetwork based on its difficulty. While the concept of routing appears in Mixture of Experts (MoE) models (Zhou et al., 2022), we cannot use them out of the box since these routers operate at the token level by directing token embeddings via lightweight Multilayer Perceptrons (MLPs), which struggle to capture the global complexity of the image. In practice, accurately estimating image difficulty demands representational power comparable to that of a full-scale classifier, a requirement that such compact routers cannot meet (Ong et al., 2025; Ding et al., 2025). Moreover, dedicating sub-

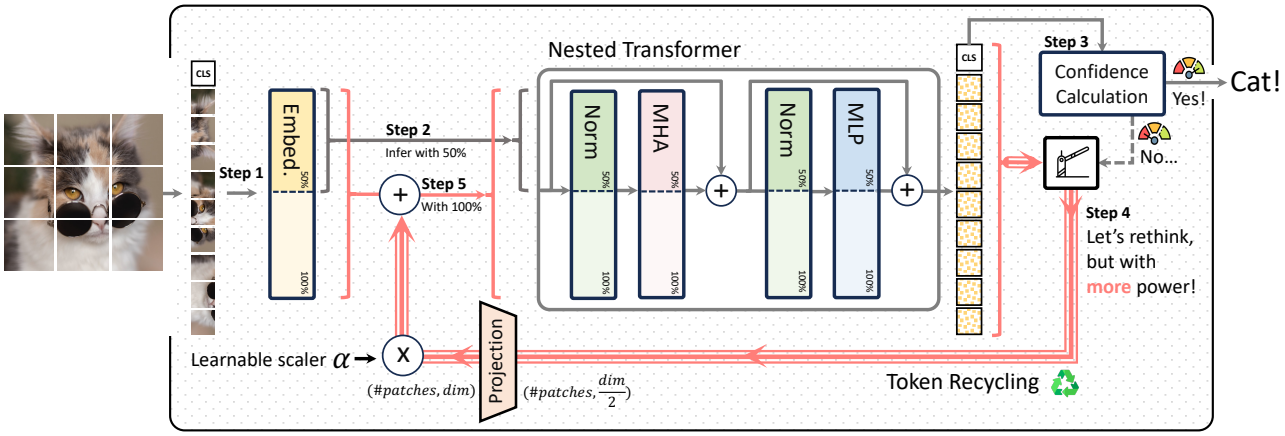


Figure 1. **Nested progressive inference with Token Recycling in *ThinkingViT*.** After embedding the input, *ThinkingViT* first activates a subset of the model, including the first attention heads (e.g., 50%), to produce an initial prediction. Due to the training procedure, these heads capture the most important features. If the certainty exceeds a threshold (easy inputs), inference terminates early to save computation. Otherwise, the resulting tokens are fused back into the input via a learnable scaling factor α, which controls how much prior knowledge is recycled. The model then *thinks more* by reprocessing the fused tokens using a larger subset of the attention heads (e.g., 100%) for a refined prediction. *ThinkingViT* enables elastic inference across different hardware budgets by adjusting the confidence threshold. The number of thinking stages and the proportion of attention heads activated at each stage can be flexibly configured based on efficiency and accuracy trade-offs, see Section 4.2.

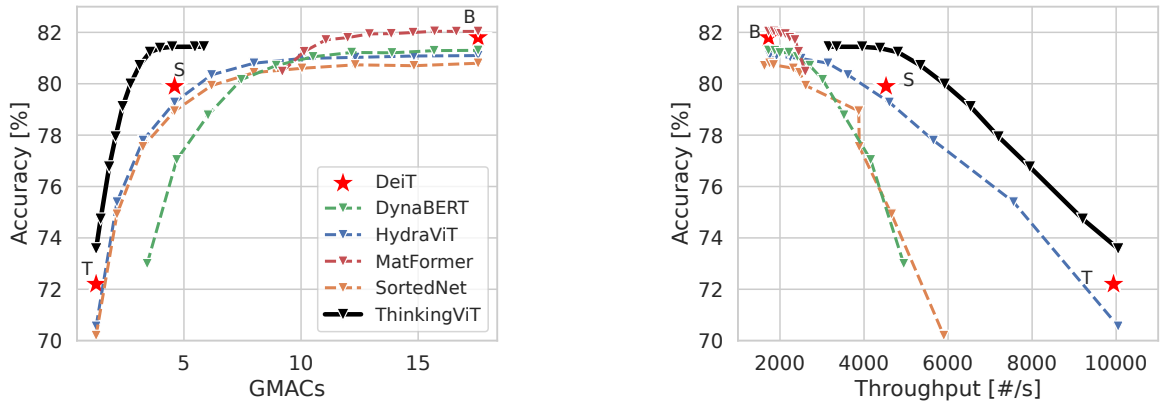
stantial compute to a separate router that merely forwards the input to another model introduces redundancy. Ideally, the routing process itself should contribute useful representations that can be *recycled* in the pipeline to enhance the final prediction, rather than being discarded after a decision is made. This leads to a critical design question: *How can we enable input-aware compute allocation within a **nested** Vision Transformer, without relying on a separate costly routing mechanism?*

***ThinkingViT*:** Inspired by recent advances in thinking-based architectures (Guo et al., 2025; Shao et al., 2024), we introduce *ThinkingViT*, a ViT built upon the vanilla ViT architecture (Dosovitskiy et al., 2021) that dynamically adjusts its inference effort based on input complexity through *nested progressive thinking stages*. *ThinkingViT* begins by activating a subset of the nested network, including the most important attention heads (e.g., 50%) to generate an initial prediction along with a certainty score. If the certainty is sufficiently high (the "aha" moment (Guo et al., 2025)), inference terminates early. Otherwise, the processed embeddings are fused with the original input embeddings and passed through a larger subset of the nested network, using an increased set of attention heads (e.g., 100%) to perform a more thorough re-evaluation. This token fusion mechanism allows the model to avoid thinking from scratch; instead, it **recycles** the knowledge gained in the first pass, refining its prediction with improved accuracy; see Figure 1. Our proposed progressive thinking mechanism enables the model not only to *think more*, but also to *think more powerfully* when processing ambiguous inputs, ensuring that more challenging examples receive greater representational capacity, i.e., more attention heads. In this pipeline, the prediction certainty serves as the central mechanism that governs elastic inference: the model halts early for cases with high certainty and progressively deepens computation for more uncertain inputs. By adjusting the certainty threshold, the model naturally balances performance against efficiency, supporting elastic inference without requiring any separate routing mechanism. Additionally, nesting all stages within a single unified model avoids parameter duplication and enables more efficient training and improved accuracy.

Table 1. Naïve recursion and depth scaling yield limited gains.

Model	Depth	Size [M]	GMACs	Acc.
DeiT-Tiny	12	5.70	1.25	72.20
Naïve Iteration				
+ 1× iteration	24	5.70	3.45	74.00
+ 2× iteration	36	5.70	3.75	74.10
+ 3× iteration	48	5.70	5.00	73.60
Depth-Expanded				
+ 1× extra depth	24	11.55	3.45	77.35
+ 2× extra depth	36	16.39	3.75	77.18
+ 3× extra depth	48	21.73	5.00	75.89
ThinkingViT	**24**	**22.1**	**5.85**	**81.44**

Furthermore, unlike iterative refinement in language models that repeatedly use the same network (Guo et al., 2025), the proposed progressive expansion is necessary in vision models, since naïve iterative chains that simply refeed a ViT's



(a) GMACs vs. Accuracy on ImageNet-1K

(b) Throughput vs. Accuracy on ImageNet-1K

Figure 2. Comparison of *ThinkingViT* with MatFormer (Devvrit et al., 2024), HydraViT (Haberer et al., 2024), SortedNet (Valipour et al., 2023), and DynaBERT (Hou et al., 2020) on ImageNet-1K, evaluated in terms of GMACs and throughput on an A100. All models are built upon the vanilla ViT architecture and the training recipes of (Touvron et al., 2021) with pretrained DeiT-Tiny as the initial checkpoint. *ThinkingViT* is trained with two progressive thinking stages using 3 and 6 attention heads, and consistently surpasses baseline by up to 2.0 p.p. at the same throughput and by up to 2.9 p.p. at the same GMACs. See Appendix G for a comparison with smaller baseline models.

own outputs into the same network fail to deliver consistent improvements and quickly saturate. In some cases, they even degrade performance, as shown in Table 1. Depth-expanded variants exhibit similar limitations. Despite having substantially more parameters, their performance also plateaus and can degrade when too many layers are added (Shen et al., 2022). These findings underscore the need for the progressive expansion strategy adopted in *ThinkingViT*, which delivers substantially better accuracy.

Contributions:

1. We introduce *ThinkingViT*, a thinking-based Vision Transformer (ViT) that brings input adaptivity to nested Transformers by dynamically adapting computation based on image difficulty.

2. *ThinkingViT* executes multiple rounds of inference by progressively activating larger subsets of attention heads, allowing the model to halt early for easy inputs based on the certainty of the predictions, while allocating greater capacity to harder examples that require richer representations.

3. *ThinkingViT* introduces Token Recycling to condition each subsequent round of inference on the features produced in the previous round, improving overall accuracy.

4. *ThinkingViT* achieves up to 2.0 percentage points higher accuracy at equal throughput, and up to 2.9 points higher at the same GMACs compared to baseline models, see Figure 2.

2. Related Work

Nested Models: Beyond approaches that utilize neural architecture search to identify optimal subnetworks within a pretrained model (Cai et al., 2019), there has been a line of research aiming to develop *nested architectures*. Slimmable networks first demonstrated that a single model can operate at multiple widths using shared weights and width-specific normalization (Yu et al., 2018; Yu & Huang, 2019). Many subsequent works build on this idea. For instance, MatFormer (Devvrit et al., 2024), based on Matryoshka Representation Learning (Kusupati et al., 2022), introduces multiple nested subsets within the hidden layer of MLP. DynaBERT (Hou et al., 2020) slices the Multi-head Attention (MHA) and MLP layers, but does not slice embeddings across layers since it relies on knowledge distillation. SortedNet (Valipour et al., 2023) generalizes nesting across MLPs, Normalization Layer (NORM), MHA, and embedding dimensions, although it retains a fixed number of attention heads. HydraViT (Haberer et al., 2024) and Slicing ViT (Zhang et al., 2024) enable slicing across embeddings, NORM, MLP, and MHA, and support a dynamic number of heads. However, like the other mentioned methods, it applies a fixed compute budget per input, limiting its ability to adapt based on input complexity and leading to inefficiencies under constrained resources.

Routing: Routing mechanisms got popular in the MoE framework (Zhou et al., 2022) and have since been extended to nested designs such as MoNE (Jain et al., 2024), MoD (Raposo et al., 2024), and AMoD (Gadhikar et al., 2025). Flextron introduces a surrogate loss predictor to guide token routing (Cai et al., 2024). These methods typ-

ically rely on lightweight MLPs, which lack the representational capacity to route at the image level, where decisions often demand the reasoning power of a full classifier. Ensemble-based strategies like Selective Query (Kag et al., 2023), OCCAM (Ding et al., 2025), and RouteLLM (Ong et al., 2025) perform input-aware routing using full models as gates. However, these approaches operate over a fixed set of pretrained models (e.g., CNNs or Transformers) and do not enable any knowledge transfer between the router and the routed models. In contrast, *ThinkingViT* recycles tokens across stages, allowing each round to build on previous inferences for improved prediction.

Thinking: Recent advancements in Large Language Models (LLM) have introduced mechanisms for adaptive reasoning depth (El-Kishky et al., 2025; Openai: Jaech et al., 2024). These methods often use reinforcement learning to dynamically adjust the number of reasoning steps based on input complexity (Guo et al., 2025; Shao et al., 2024). However, excessive reasoning length can inflate inference cost without proportional gains in accuracy (Kumar et al., 2025). Motivated by this, recent efforts seek to preserve the benefits of deep reasoning while avoiding redundant computation by introducing test-time adaptability or confidence-based early stopping (Muennighoff et al., 2025; Zeng et al., 2025). These models, however, reuse the same network in each round, while *ThinkingViT* scales up capacity during rethinking, which is essential for vision tasks, see Table 1.

3. *ThinkingViT*

In this section, we introduce the design components of *ThinkingViT*. We first describe how we construct a hierarchy of ordered subnetworks inside the vanilla ViT backbone. Next, we explain the recursive inference loop and how *ThinkingViT* fuses knowledge across sequential rounds. Finally, we show how *ThinkingViT* enables elastic inference through an entropy-based "aha!" criterion that halts thinking once it achieves sufficient certainty.

3.1. Nested Vision Transformer

ThinkingViT is built on top of the standard ViT architecture. Let $V_{D,H}$ be a ViT model with H attention heads and the embedding dimension of D. ViT begins by tokenizing the input image x into P non-overlapping patches, each of which is projected into a D-dimensional embedding vector $\mathcal{E}^D$ using a CNN-based patch embedding function. After adding positional encodings, the resulting sequence is processed by L standard Transformer blocks:

$$z^l = \text{Block}^l(z^{l-1}) \quad \text{for } l = 1, \dots, L \tag{1}$$

Inducing nested subnetworks: Inspired by (Haberer et al., 2024), *ThinkingViT* induces n nested subnetworks within this architecture. Each subnetwork is denoted as V_{d_i,h_i} and is constructed using the first d_i embedding values of each token and the first h_i attention heads from the full model. To build such a nested structure, *ThinkingViT* slices all components of the ViT, including the embedding layer, attention modules, MLP blocks, and normalization layers, according to these indices. This yields contained subnetworks with progressively increasing embedding dimensionality and attention capacity:

$$V_{d_1,h_1}(x) \subset V_{d_2,h_2}(x) \subset \cdots \subset V_{d_n,h_n}(x), \tag{2}$$

$$d_1 < d_2 < \cdots < d_n, \quad h_1 < h_2 < \cdots < h_n \tag{3}$$

3.2. Looping nested Vision Transformers through Token Recycling

Inspired by recent advances in reasoning models (Guo et al., 2025; Shao et al., 2024), *ThinkingViT* operates through multiple rounds of progressive refinement using a "Token Recycling" mechanism. Although, in practice, we find that two subnetworks are sufficient to achieve strong performance (as demonstrated in Figure 3), we present the general multi-round formulation here for clarity and broader applicability.

After constructing the nested architecture, *ThinkingViT* begins by using the smallest subnetwork V_{d_i,h_i}, with the embedding dimension of d_i and h_i attention heads. This subnetwork processes the input image to predict the class and generate token embeddings z^L. Subsequently, to refine its prediction, the model expands computational capacity by activating a larger subnetwork $V_{d_j,h_j}(x)$, featuring h_j attention heads, where $h_i < h_j$ and $d_i < d_j$. Afterwards, it fuses the produced token embeddings z^L with the new input embeddings $\mathcal{E}^{d_j}(x) \in \mathbb{R}^{P \times d_j}$, through a projection layer and scaled by a learnable parameter α, determining the weight of embeddings "recalled" from the previous step:

$$\mathcal{E}^{d_j}_{\text{fused}} = \alpha \cdot \text{Proj}_{d_i \to d_j}(z^L) + \mathcal{E}^{d_j}(x), \tag{4}$$

This *Token Recycling* mechanism allows the model to reuse prior representations instead of reprocessing from scratch. The process continues iteratively and unlike language models that reuse the same network for reasoning (Guo et al., 2025), our approach increases model capacity by activating progressively larger subsets of attention heads, enhancing representational capacity at each step. Progressive expansion is essential in vision tasks, where naive recursion quickly plateaus, as shown in Table 1. Appendix D and Appendix C compare recycling and fusion strategies in detail.

3.3. Training all subnetworks jointly

During training, *ThinkingViT* executes all n thinking rounds and minimizes a weighted classification loss $\mathcal{L}_{\text{cls}}$

across all stages:

$$\mathcal{L} = \sum_{i=1}^{n} \lambda_i \cdot \mathcal{L}_{\text{cls}}(V_{d_i,h_i}(x), y) \tag{5}$$

where y is the ground-truth label and λ_i controls the contribution of each subnetwork to the global objective. For a small number of subnetworks, it is computationally feasible to optimize all subnetworks simultaneously, as the gradient computation graph remains reasonably compact. However, as n increases, training all subnetworks jointly becomes computationally intensive. In such cases, we adopt strategies such as the sandwich rule (Yu & Huang, 2019) and stochastic subnetwork sampling (Haberer et al., 2024) to reduce overhead.

3.4. The "Aha!" moment

By inducing nested iterative subnetworks inside the model, *ThinkingViT* builds a hierarchy of n nested thinking steps, where each step operates on top of the previous one to refine the prediction. However, different inputs have different complexities, and for some inputs, the model becomes confident after only k iterations, where $k < n$ (e.g., after a single round of thinking). The "aha!" moment occurs when the model recognizes that it has reached sufficient certainty, allowing it to allocate an appropriate number of thinking rounds based on the input complexity (Guo et al., 2025). Let f_k denote the softmax output produced at the k-th iteration. After the k^{th} round, the model measures its certainty using Shannon entropy:

$$\mathcal{H}(f_k) = -\sum_{c=1}^{C} f_k^{(c)} \log f_k^{(c)}, \tag{6}$$

where C is the number of classes. If the entropy $\mathcal{H}(f_k)$ is lower than a predefined threshold τ, inference halts early and the model accepts the current prediction. Otherwise, *ThinkingViT* activates a larger subnetwork to further refine the result. This simple metric performs on par with more complex criteria on ImageNet-1K (Jitkrittum et al., 2023). Our design also allows alternative routing modules (Kag et al., 2023; Ding et al., 2025) to be integrated as drop-in replacements without changing the architecture or training.

3.5. Elastic inference

ThinkingViT supports elastic inference by using its own certainty to determine when to stop. The entropy threshold τ governs the trade-off between efficiency and accuracy. A low threshold enforces stricter confidence requirements, causing most inputs to undergo more thinking steps, which increases computational cost but improves accuracy. In contrast, a high threshold allows earlier exits for more inputs, reducing latency and computation cost at the expense of potential accuracy loss. This flexibility enables users to tune the model according to specific resource and performance needs.

Table 2. Impact of attention expansion and loss weighting on *ThinkingViT* with two rounds of progressive thinking on ImageNet-1K.

Model Variant	Loss Weight	Acc. [%] 1st Round	Acc. [%] 2nd Round
ThinkingViT 2H → 3H	[0.5, 0.5]	65.35	74.13
ThinkingViT 3H → 6H	[0.5, 0.5]	73.58	81.44
ThinkingViT 3H → 6H	[0.4, 0.6]	73.22	81.43
ThinkingViT 3H → 6H	[0.6, 0.4]	73.93	81.28
ThinkingViT 3H → 9H	[0.5, 0.5]	72.51	82.02
ThinkingViT 3H → 9H	[0.4, 0.6]	71.71	82.15
ThinkingViT 3H → 9H	[0.6, 0.4]	73.02	81.92
ThinkingViT 3H → 12H	[0.5, 0.5]	72.03	81.70
ThinkingViT 3H → 12H	[0.4, 0.6]	70.78	81.80
ThinkingViT 3H → 12H	[0.6, 0.4]	72.23	81.51

3.6. Advantages of the isomorphic architecture of vanilla ViT

As shown in prior works (Devvrit et al., 2024; Shukla et al., 2024; Valipour et al., 2023), the uniform layer structure of vanilla ViT, with consistent embedding sizes, token counts, and attention heads, makes it well suited for slicing and allows straightforward selection across layers. In contrast, hierarchical models like Swin Transformer (Liu et al., 2022) vary these dimensions due to window-based partitioning, which complicates slicing. A similar preference for isomorphic designs has been observed in LLMs (Cai et al., 2024; 2025). Moreover, vanilla ViT enables *ThinkingViT*'s Token Recycling through a simple projection layer, while hierarchical models require more complex mechanisms to align the mismatched dimensions, i.e., number of tokens and embedding sizes.

4. Evaluation

In this section, we first evaluate *ThinkingViT* on ImageNet-1K (Russakovsky et al., 2015) and its variants, comparing it to the SOTA nested baselines. Then, we analyze how *ThinkingViT* leverages entropy as a signal to allocate computation based on input difficulty. Lastly, we highlight a key limitation of our method: the reduced effectiveness of *ThinkingViT*'s binary routing when faced with uniformly challenging inputs.

4.1. Setup

Implementation details: We run all experiments on ImageNet-1K (Russakovsky et al., 2015) at a resolution

Table 3. ThinkingViT performance with three rounds of thinking on ImageNet-1K, demonstrating that *ThinkingViT* naturally scales to deeper thinking hierarchies and yields higher final accuracy.

Model	Acc. [%] 1st Round	2nd Round	3rd Round
ThinkingViT 2H (33%) → 3H (50%) → 6H (100%)	64.62	73.56	81.43
ThinkingViT 3H (25%) → 6H (50%) → 12H (100%)	70.77	80.00	82.35

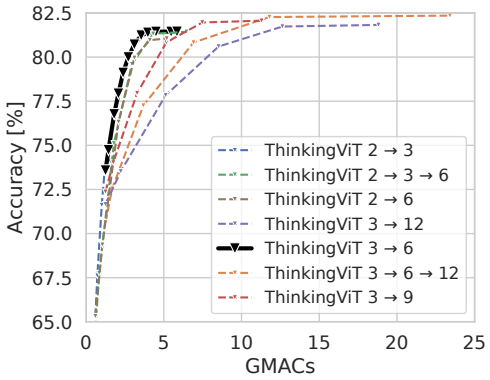


Figure 3. Accuracy vs. GMACs for *ThinkingViT* variants on ImageNet-1K. 3H → 6H achieves the best trade-off, while 2H → 3H → 6H covers the widest range with only a slight accuracy drop. See Appendix I for details.

of 224 × 224, using models implemented in `timm` (Wightman, 2019) and trained following the setup of Touvron et al. (2021), with all models initialized with a pretrained DeiT-Tiny checkpoint. We train on a cluster with NVIDIA A100 GPUs, taking about 10 minutes per epoch on 2 GPUs. During prototyping, we conducted roughly 100 training runs, each lasting 300 epochs.

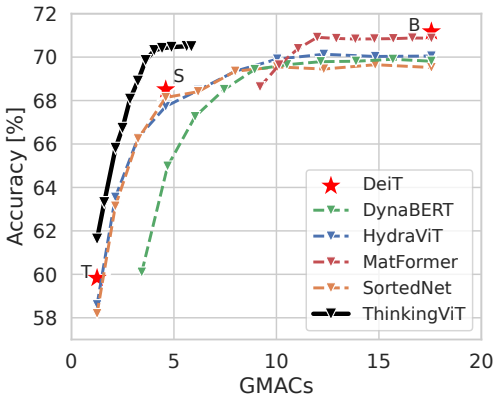
Baselines: We compare *ThinkingViT* to several baselines: MatFormer (Devvrit et al., 2024), which slices only the hidden layer of MLP while keeping MHA and embedding dimensions fixed; DynaBERT (Hou et al., 2020), which slices both MHA and the hidden layer of MLP while keeping embeddings fixed; SortedNet (Valipour et al., 2023), which slices all embeddings, NORM, MHA, and MLP, while keeping the number of heads fixed; and HydraViT (Haberer et al., 2024), which slices MHA, embeddings, NORM, and MLP, while also changes the number of heads. Similar to HydraViT, *ThinkingViT* adopts a slicing strategy across the number of heads, MHA, embeddings, NORM, and MLP layers. However, unlike the above methods that follow static inference paths regardless of input difficulty, *ThinkingViT* introduces input-adaptive computation.

4.2. Trade-offs in attention expansion

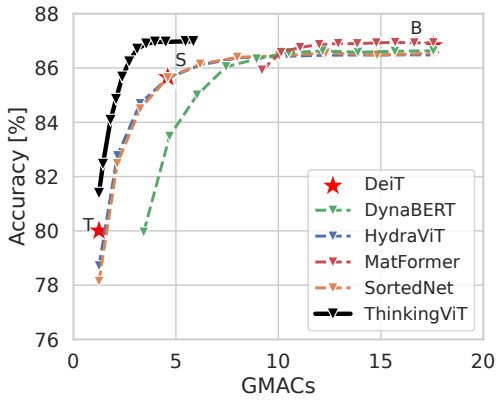
ThinkingViT expands computational capacity through progressive activation of attention heads. As shown in Table 2, starting with 3 heads and expanding to 6 (3H → 6H) yields the best first-round accuracy, while 3H → 9H achieves the highest second-round accuracy on ImageNet-1K. Notably, by using loss weighting we can tune the model to put more focus on the first round or second round based on the desired trade-off between computation and accuracy. We also explore 3-stage variants in Table 3, demonstrating that *ThinkingViT* naturally scales to deeper thinking hierarchies and yields higher final accuracy. Generally, the number of thinking stages and the attention head expansion step size are hyperparameters, and their optimal configuration depends on the dataset and target efficiency objectives. In Figure 3, we evaluate this trade-off and find that the 3H → 6H configuration offers the most favorable balance between accuracy and compute on ImageNet-1K. The 2H → 3H → 6H variant spans the widest GMAC range among the high-performing models, while incurring only a small drop in accuracy compared to 3H → 6H. At the upper end, 3H → 6H → 12H achieves the highest final accuracy, surpassing 3H → 6H by 0.91 percentage points (p.p.), but raises the compute cost from 5.85 GMACs to 23.41 GMACs for only a marginal gain. This substantial increase in cost leads to lower overall efficiency compared to the 3H → 6H setup, underscoring the importance of aligning *ThinkingViT*'s thinking strategy with specific deployment goals. Since we focus on configurations that offer the best trade-off between accuracy and GMACs, we adopt 3H → 6H for subsequent experiments. For a more detailed view, see Appendix I

4.3. Comparing *ThinkingViT* and baselines

Figure 2 compares *ThinkingViT* with SOTA baselines in terms of GMACs and inference throughput, using the 3H → 6H configuration. *ThinkingViT* achieves up to 2.0 p.p. higher accuracy at equal throughput, and up to 2.9 p.p. higher accuracy at the same GMACs compared with its baselines. This improvement stems from three key factors. *First*, *ThinkingViT* makes early predictions using only 3 heads for easy inputs and expands to 6 heads only when needed. *Second*, *ThinkingViT* fuses embeddings from the first stage into the second, leading to better performance than flat 6-head models like HydraViT or DeiT-S. Moreover, *ThinkingViT* achieves 81.44% with only 22.01 M params, which is just 0.36 p.p. lower than DeiT-Base with 86.6 M params. This result further validates the effectiveness of the Token Recycling strategy. Importantly, it also demonstrates



(a) ImageNet-V2



(b) ImageNet-Real

Figure 4. GMACs vs. Accuracy on ImageNet-V2 and ImageNet-Real, similar to Figure 2, *ThinkingViT* has superior performance compared to the baselines.

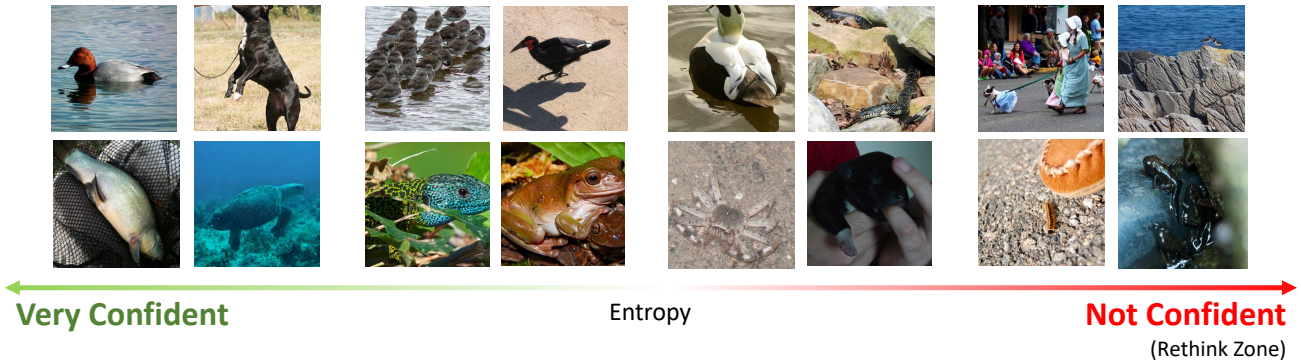


Figure 5. Visualization of images by first-round entropy. *ThinkingViT* confidently classifies simple, clear images in one round, while complex cases with occlusion or clutter show higher entropy and trigger a second round.

that achieving higher accuracy does not necessarily require a larger model; rather, it depends on applying sufficient computation. *Third*, baselines typically employ multiple slicing points throughout the architecture to meet diverse GMACs. However, optimizing for all such configurations can lead to accuracy reduction (Haberer et al., 2024). *ThinkingViT* sidesteps this issue by supporting elastic inference using just a few nested subnetworks, with compute budget adaptability controlled via the entropy threshold. We report detailed performance results in the Appendix A.

To assess robustness, we evaluate *ThinkingViT* on ImageNet-V2 (Recht et al., 2019) and ImageNet-ReaL (Beyer et al., 2020; Russakovsky et al., 2015) in Figure 4, and on ImageNet-R (Hendrycks et al., 2020) in Figure 7. Additional results on ImageNet-A (Hendrycks et al., 2019) and ImageNet-Sketch (Wang et al., 2019) are provided in Appendix H. While the performance gap between *ThinkingViT* and the baselines narrows on ImageNet-R, -A, and -Sketch (see Section 4.5), *ThinkingViT* outperforms the baselines across all these benchmarks. Furthermore, on ImageNet-ReaL, -Sketch, and -R, *ThinkingViT* exceeds the accuracy of DeiT-Base, despite using significantly fewer parameters (22.1M vs. 86.6M) and lower GMACs (5.85 vs. 17.56), which highlights the effectiveness of the *Token Recycling*. We also compare *ThinkingViT* to Early Exit in Appendix B and to several token-based pruning approaches in Appendix F.

4.4. Analyzing entropy as a signal to think deeper

After each stage, *ThinkingViT* uses output entropy to assess certainty and decide if further computation is needed. Figure 6 shows the entropy after the first round of inference on ImageNet-A (Hendrycks et al., 2019), ImageNet-V2 (Recht et al., 2019), and ImageNet-R (Hendrycks et al., 2020), providing insight into how *ThinkingViT* 3H $\rightarrow$ 6H estimates uncertainty across different distribution shifts. On easier datasets like ImageNet-V2, entropy is left-skewed, reflecting confident early predictions. In contrast, harder datasets like ImageNet-A and -R produce right-skewed distributions, triggering deeper inference in *ThinkingViT*. In Figure 8, we

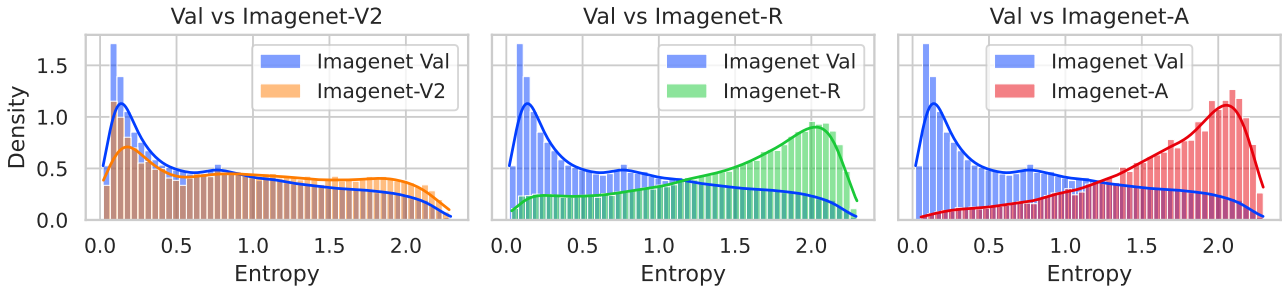


Figure 6. Entropy distribution after the first inference round across ImageNet validation sets. Simpler datasets like ImageNet-V2 show confident early predictions (left-skewed), while harder ones like ImageNet-A and -R show greater uncertainty (right-skewed), which triggers *ThinkingViT* to go for the next round of thinking.

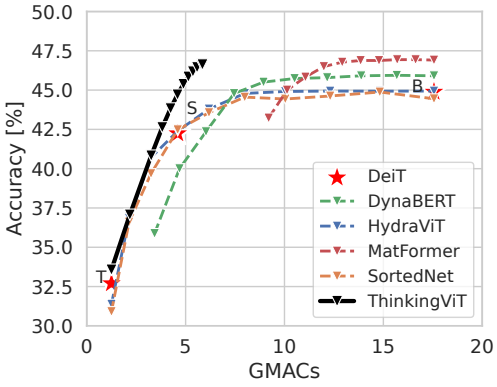


Figure 7. GMACs vs. Accuracy on ImageNet-R. *ThinkingViT*'s superior performance slightly declines when most inputs require a second round of inference.

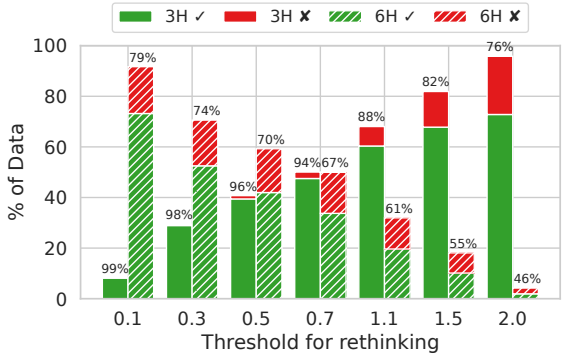


Figure 8. Load distribution of *ThinkingViT* 3H → 6H across entropy thresholds. Bars indicate usage share; numbers show accuracy (green: correct, red: incorrect) on ImageNet-1K. High 3H accuracy at low entropy confirms the effectiveness of entropy-based routing.

show the load distribution on the 3H and 6H submodels under varying entropy thresholds. As entropy decreases, fewer samples proceed to the second stage, and those that exit early are rarely misclassified. This confirms that entropy effectively halts computation on easy inputs without sacrificing accuracy. Furthermore, Figure 5 shows images with different entropy levels to illustrate how *ThinkingViT* adjusts its computation. Easy images such as clear pictures of a single object have low entropy and stop after one round of thinking. More difficult images, including those with several objects, blocked views, or poor lighting, show higher entropy and trigger a second round of thinking to improve accuracy.

4.5. Performance limitation on uniformly hard inputs

Despite its advantages, *ThinkingViT* has a key limitation: the effectiveness of early predictions for simpler inputs relies heavily on the quality of predictions at each stage. At each decision point, the model evaluates its certainty to determine whether to proceed to a deeper stage. This approach works well when input difficulty varies, allowing early predictions on easier examples (see Figure 2). However, when most inputs are difficult, the model frequently advances to deeper stages, which slightly reduces the efficiency gains of early prediction. In Figure 7, we demonstrate this on ImageNet-A, a dataset composed of examples misclassified by ResNet-50 (He et al., 2016), which scores 76% on ImageNet-1K. *ThinkingViT* with 3H achieves 73.6%, indicating that nearly all ImageNet-A samples are too challenging for early classification and thus trigger the second stage. This explains why *ThinkingViT*'s performance slightly degrades on heavily biased datasets. See Appendix J for further discussion of limitations.

5. Conclusion

We introduce *ThinkingViT*, a nested ViT architecture that dynamically adjusts computation during inference based on input complexity. By progressively activating attention heads and employing our novel Token Recycling mechanism, *ThinkingViT* achieves improved computational efficiency and accuracy. Experiments demonstrate significant gains over nested baselines by up to 2.0 p.p. higher

accuracy at equal throughput and up to 2.9 p.p. higher accuracy at equal GMACs on ImageNet-1K. Due to its backbone-preserving design, *ThinkingViT* serves as a plug-in upgrade for vanilla ViTs while remaining compatible with token pruning and patch merging methods. Moreover, the same nested progressive thinking principle is architecture-agnostic, opening a path toward elastic inference in LLMs. We would like to explore these ideas as future work.

Acknowledgments and Disclosure of Funding This research received funding from the Federal Ministry for Digital and Transport under the CAPTN-Förde 5G project (grant no. 45FGU139H), the German Ministry of Transport and Digital Infrastructure through the CAPTN Förde Areal II project (grant no. 45DTWV08D), and the Federal Ministry for Economic Affairs and Climate Action under the Marispace-X project (grant no. 68GX21002E). It was supported in part by high-performance computing resources provided by the Kiel University Computing Centre and the Hydra computing cluster, funded by the German Research Foundation (grant no. 442268015) and the Petersen Foundation (grant no. 602157).

References

Beyer, L., Henaff, O. J., Kolesnikov, A., Zhai, X., and van den Oord, A. Are we done with imagenet? *arXiv preprint arXiv:2002.05709*, 2020.

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*, 2023. URL https://openreview.net/forum?id=JroZRaRw7Eu.

Cai, H., Gan, C., Wang, T., Zhang, Z., and Han, S. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019.

Cai, R., Muralidharan, S., Heinrich, G., Yin, H., Wang, Z., Kautz, J., and Molchanov, P. Flextron: many-in-one flexible large language model. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 5298–5311, 2024.

Cai, R., Muralidharan, S., Yin, H., Wang, Z., Kautz, J., and Molchanov, P. LLamaflex: Many-in-one LLMs via generalized pruning and weight sharing. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AyC4uxx2HW.

Darcet, T., Oquab, M., Mairal, J., and Bojanowski, P. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=2dnO3LLiJ1.

Devvrit, F., Kudugunta, S., Kusupati, A., Dettmers, T., Chen, K., Dhillon, I., Tsvetkov, Y., Hajishirzi, H., Kakade, S., Farhadi, A., et al. Matformer: Nested transformer for elastic inference. *Advances in Neural Information Processing Systems*, 37:140535–140564, 2024.

Ding, D., Xu, B., and Lakshmanan, L. V. S. OCCAM: Towards cost-efficient and accuracy-aware classification inference. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=CUABD2qIB4.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

El-Kishky, A., Wei, A., Saraiva, A., Minaiev, B., Selsam, D., Dohan, D., Song, F., Lightman, H., Clavera, I., Pachocki, J., et al. Competitive programming with large reasoning models. *arXiv preprint arXiv:2502.06807*, 2025.

Gadhikar, A., Majumdar, S. K., Popp, N., Saranrittichai, P., Rapp, M., and Schott, L. Attention is all you need for mixture-of-depths routing. In *First Workshop on Scalable Optimization for Efficient and Adaptive Foundation Models*, 2025. URL https://openreview.net/forum?id=1uDP4ld3eZ.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Haberer, J., Hojjat, A., and Landsiedel, O. Hydravit: Stacking heads for a scalable vit. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=kk0Eaunc58.

Hatamizadeh, A. and Kautz, J. Mambavision: A hybrid mamba-transformer vision backbone. *arXiv preprint arXiv:2407.08083*, 2024.

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.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

Hou, L., Huang, Z., Shang, L., Jiang, X., Chen, X., and Liu, Q. Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems*, 33:9782–9793, 2020.

Jain, G., Hegde, N., Kusupati, A., Nagrani, A., Buch, S., Jain, P., Arnab, A., and Paul, S. Mixture of nested experts: Adaptive processing of visual tokens. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=HbV5vRJMOY.

Jitkrittum, W., Gupta, N., Menon, A. K., Narasimhan, H., Rawat, A. S., and Kumar, S. When does confidence-based cascade deferral suffice? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=4KZhZJSPYU.

Kag, A., Fedorov, I., Gangrade, A., Whatmough, P., and Saligrama, V. Efficient edge inference by selective query. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=jpR98ZdIm2q.

Kumar, A., Roh, J., Naseh, A., Karpinska, M., Iyyer, M., Houmansadr, A., and Bagdasarian, E. Overthink: Slowdown attacks on reasoning llms. *arXiv e-prints*, pp. arXiv–2502, 2025.

Kusupati, A., Bhatt, G., Rege, A., Wallingford, M., Sinha, A., Ramanujan, V., Howard-Snyder, W., Chen, K., Kakade, S., Jain, P., et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022.

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12009–12019, 2022.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J. E., Kadous, M. W., and Stoica, I. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

Openai: Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., and Hsieh, C.-J. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

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.

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pp. 5389–5400, 2019.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Shen, Z., Liu, Z., and Xing, E. Sliced recursive transformer. In *European Conference on Computer Vision*, pp. 727–744. Springer, 2022.

Shukla, A., Vemprala, S., Kusupati, A., and Kapoor, A. Matmamba: A matryoshka state space model. *arXiv preprint arXiv:2410.06718*, 2024.

Teerapittayanon, S., McDanel, B., and Kung, H.-T. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

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.

Touvron, H., Cord, M., and Jégou, H. Deit iii: Revenge of the vit. In *European conference on computer vision*, pp. 516–533. Springer, 2022.

Valipour, M., Rezagholizadeh, M., Rajabzadeh, H., Tahaei, M., Chen, B., and Ghodsi, A. Sortednet, a place for every network and every network in its place: Towards a generalized solution for training many-in-one neural networks. *arXiv preprint arXiv:2309.00255*, 2023.

Wang, H., Ge, S., Lipton, Z., and Xing, E. P. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pp. 10506–10518, 2019.

Wightman, R. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

Xu, G., Hao, Z., Luo, Y., Hu, H., An, J., and Mao, S. Vision transformers on the edge: A comprehensive survey of model compression and acceleration strategies. *arXiv preprint arXiv:2503.02891*, 2025.

Yu, J. and Huang, T. S. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1803–1811, 2019.

Yu, J., Yang, L., Xu, N., Yang, J., and Huang, T. Slimmable neural networks. *arXiv preprint arXiv:1812.08928*, 2018.

Zeng, Z., Cheng, Q., Yin, Z., Zhou, Y., and Qiu, X. Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities? *arXiv preprint arXiv:2502.12215*, 2025.

Zhang, Y., Coskun, H., Ma, X., Wang, H., Ma, K., Chen, S. X., Hu, D. H., and Fu, Y. Slicing vision transformer for flexibile inference. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=zJNSbgl4UA.

Zhou, Y., Lei, T., Liu, H., Du, N., Huang, Y., Zhao, V., Dai, A. M., Le, Q. V., Laudon, J., et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

A. Adaptive Inference Performance of *ThinkingViT* under Varying Entropy Thresholds

Table 4 presents detailed results of *ThinkingViT*'s adaptive inference behavior under different entropy thresholds. We report accuracy, throughput, total compute (in GMACs), and the ratio of inputs that proceed to the second stage of inference of *ThinkingViT* 3H $\rightarrow$ 6H.

Table 4. Performance metrics of *ThinkingViT* 3H $\rightarrow$ 6H across different entropy thresholds

Entropy Threshold	Accuracy	Throughput [#/s]	Params [M]	GMACs	Second Round Call Ratio [%]
0	81.444	3157.09	22.01	5.85	100.0
0.1	81.440	3347.69	22.01	5.47	71.7
0.3	81.438	3955.05	22.01	4.50	70.58
0.5	81.386	4380.71	22.01	3.98	59.29
0.7	81.230	4807.04	22.01	3.55	49.95
0.9	80.714	5342.47	22.01	3.11	40.36
1.1	79.990	5918.90	22.01	2.72	31.97
1.3	79.114	6535.13	22.01	2.38	24.63
1.5	77.936	7201.46	22.01	2.08	18.11
1.7	76.766	7944.38	22.01	1.81	12.13
2	74.736	9203.90	22.01	1.44	4.20
2.5	73.580	10047.60	22.01	1.25	0.0

B. Comparison with Early Exit

Table 5 compares *ThinkingViT* with a standard early exit (Teerapittayanon et al., 2016), applied to DeiT (Touvron et al., 2022) that matches its GMACs and throughput. The baseline consists of 24 layers: the first 12 use 3 attention heads (3H) and the remaining 12 use 6 heads (6H). Similar to *ThinkingViT*, the exits are jointly trained together. At the 3H point, *ThinkingViT* records slightly lower accuracy because the first three heads must generate representations that remain compatible with the later expansion to six heads. Additionally, their weights must be trained in a way that allows the weights of the subsequent three heads to build upon them.

After expanding to 6 attention heads, *ThinkingViT* reaches 81.44% top-1 accuracy, improving by 3.37 p.p. upon the early exit baseline, which attains 78.08%. This improvement likely stems from *ThinkingViT* increasing the number of active attention heads within a fixed 12-layer backbone, allowing gradients to pass through fewer layers (12 layers compared to 24 layers) and reducing some of the optimization challenges associated with deeper networks. These findings indicate that allocating compute along the width dimension by gradually increasing attention heads can be more effective than depth-based early exit strategies used in prior work.

Table 5. Comparison of *ThinkingViT* and Early Exit.

Model	GMACs	Throughput [#]	Params [M]	Accuracy [%]
DeiT-Tiny	1.25	10047.6	5.7	72.2
DeiT-Small	4.6	4603.6	22.1	79.9
Early Exit 3H	1.25	10047.6	27.8	74.51
Early Exit 3H $\rightarrow$ 6H	5.85	3157.1	27.8	78.08
***ThinkingViT* 3H**	1.25	10047.6	22.1	73.58
***ThinkingViT* 3H $\rightarrow$ 6H**	5.85	3157.1	22.1	**81.44 (+3.36)**

C. Ablation of Different Token Recycling Strategies

In Table 6, we compare several design variants for recycling information from the first round of inference to get better performance in the second round. We conduct these experiments on *ThinkingViT* 3H $\rightarrow$ 6H. In the *Layerwise Activation Snapshots* variant, we cache the hidden states at each of the 12 layers from the first round and feed them into the

corresponding layers in the second round. However, this approach performs suboptimally, likely because it forces all intermediate representations from the first round to be reused in the second round, which reduces the performance. In the *Memory Tokens* design, we introduce a set of learnable memory tokens (Darcet et al., 2024) that, in the first round, store helpful information for the second round. We also experiment with hybrid strategies that combine activation snapshots with Memory Tokens or introduce a fresh `[CLS]` token in the second round. Ultimately, we find that the simplest strategy, *Final-Layer Token Recycling*, which reuses the features from the last layer of the first round, achieves the best performance. This indicates that the output of the first round already captures sufficient high-level information to guide the second round effectively, while being comparatively simple to implement.

It is important to note that these experiments were conducted during the early experimental phase of *ThinkingViT*, and due to training resource constraints, they were not trained with the full joint training strategy described in Section 3.3, but rather with the stochastic training method introduced in (Haberer et al., 2024). As a result, their accuracies are slightly lower than those of the final model reported in Section 4.

Table 6. Comparison of design variants for conditioning the second round of inference on the first.

Design Variant	**First Thought Acc. [%]**	**Second Thought Acc. [%]**
DeiT Baseline (Tiny → Small)	72.2	79.9
Layerwise Activation Snapshots	73.794	78.566
Memory Tokens	73.96	78.9
Layerwise Activation Snapshots + new `[CLS]` in second round	73.58	77.94
Layerwise Activation Snapshots + Memory Tokens	73.71	79.04
Final-Layer Token Recycling (*ThinkingViT*)	73.13	**80.05**

D. Ablation of Different Fusing Strategies

Table 7 compares six fusion strategies that recycle the tokens produced in the first round, denoted as z_1, by incorporating them into the initial patch embeddings of the second round, $\mathcal{E}_2$, on *ThinkingViT* with a configuration of $3H \rightarrow 6H$. Each variant forms the second-round input as a linear blend, where α is a learnable scalar. The strategies differ in how the two embeddings are projected to the same dimensional space. We consider two projection strategies: (I) a parameter-free approach that either repeats the embedding or pads the lower-dimensional embeddings with zeros, and (II) a learnable linear projection. Empirically, the learnable projection achieves the best overall accuracy by having the highest accuracy in the second round while maintaining performance comparable to the strongest model configuration in the first round. In addition, we evaluate different initializations of α and observe that initializing with $\alpha = 0$ consistently leads to the best performance. This initialization allows the model to begin training without relying on information from the first round, thereby enabling it to learn the optimal degree of reuse. For example, in *ThinkingViT* configured with 3H → 6H heads, the learned value of α converges to -0.19.

Table 7. Impact of different fusion strategies for integrating first-round tokens (z_1) into second-round embeddings ($\mathcal{E}_2$) during progressive inference on *ThinkingViT* with 3H → 6H.

Fusion Method	**Dim Alignment**	**Note**	**Acc. [%]**	
			1st Round	**2nd Round**
$\alpha \cdot z_1 + \mathcal{E}_2$	Pad z_1 with zeros	Pad the first half, $init(\alpha) = 0$	73.32%	81.37%
$\alpha \cdot z_1 + \mathcal{E}_2$	Pad z_1 with zeros	Pad the second half, $init(\alpha) = 0$	**74.12%**	80.32%
$\alpha \cdot z_1 + \mathcal{E}_2$	Repeat z_1	$init(\alpha) = 1$	72.99%	80.87%
$\alpha \cdot z_1 + \mathcal{E}_2$	Repeat z_1	$init(\alpha) = 0$	73.14%	81.41%
$z_1 + \alpha \cdot \mathcal{E}_2$	Repeat z_1	$init(\alpha) = 0$	72.89%	80.51%
$\alpha \cdot z_1 + \mathcal{E}_2$	Linear (*ThinkingViT*)	$init(\alpha) = 0$	73.58%	**81.44%**

E. Prediction Dynamics Across Two Inference Rounds on ImageNet-1K

Figure 9 presents prediction dynamics across two inference rounds of *ThinkingViT* 3H → 6H on the ImageNet-1K validation set. A small portion of samples, approximately 2%, were correctly classified in the first round but misclassified in the second.

This phenomenon, often attributed to *overthinking* (Kumar et al., 2025), is also observed in other architectures such as ResNet and Swin Transformers (Ding et al., 2025), where smaller models sometimes outperform larger ones on few samples by avoiding unnecessary complexity. The majority of samples, around 70%, were correctly classified in both rounds. These generally correspond to visually simple or unambiguous cases where one round of inference is sufficient. Roughly 10% of samples were initially misclassified but corrected in the second round, demonstrating the benefit of additional reasoning for harder examples. Finally, around 16% remained misclassified across both rounds, indicating that these inputs are intrinsically ambiguous or fall outside the model's capacity, even with increased computation.

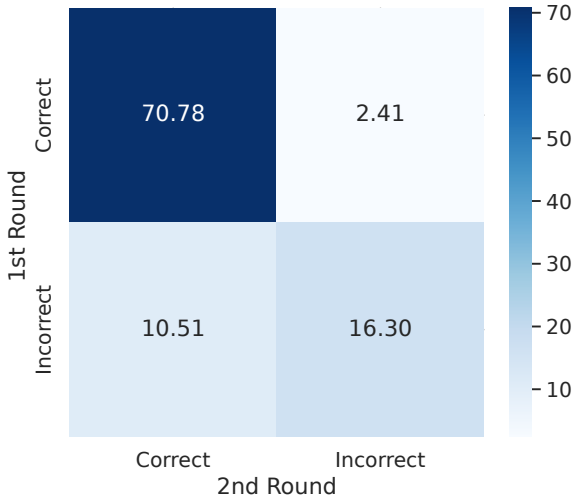


Figure 9. Prediction dynamics of *ThinkingViT* 3H → 6H across two inference rounds on ImageNet-1K.

F. Comparing *ThinkingViT* with Other Adaptive Baselines

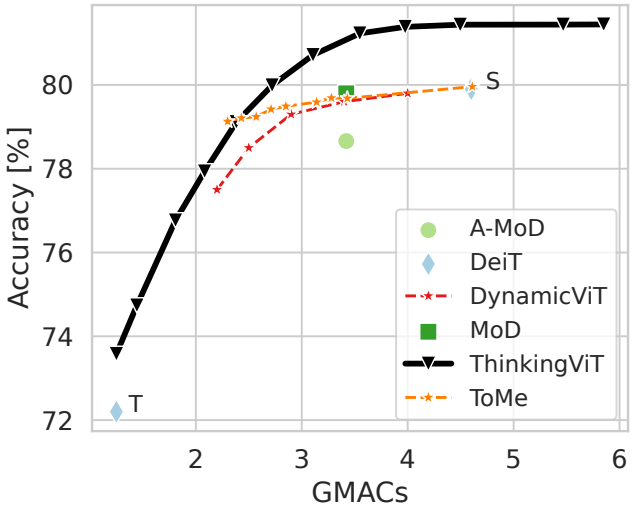


Figure 10. GMACs vs. Accuracy comparison with token-based dynamic models. *ThinkingViT* achieves higher accuracy at the same compute budget.

ThinkingViT builds upon a nested backbone to enable input adaptivity, so our evaluation in Section 4 focuses on nested Transformer baselines. For completeness, we also evaluate against several representative token-level dynamic models, including MoD (Raposo et al., 2024), AMoD (Gadhikar et al., 2025), ToMe (Bolya et al., 2023), and DynamicViT (Rao et al., 2021). As shown in Figure 10, *ThinkingViT* achieves greater scalability and consistently better GMACs-Accuracy

tradeoffs. We note that *ThinkingViT* performs routing at the image level, which is orthogonal to token-level approaches and can be combined with them for added flexibility, suggesting a promising direction for future work.

G. Comparison of Accuracy vs. GMACs for Baselines based on DeiT-Small

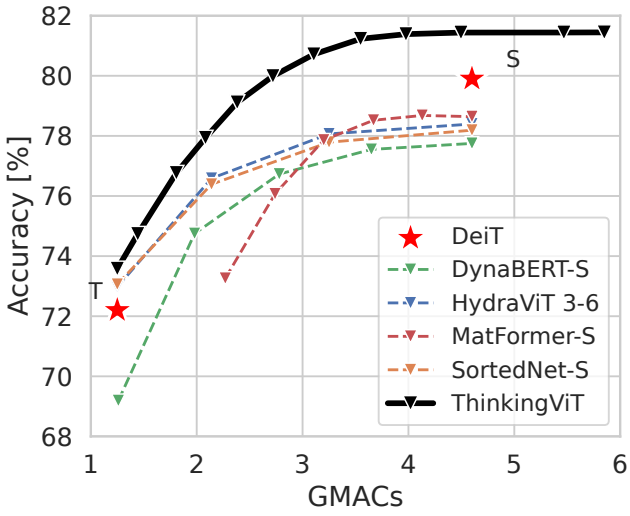
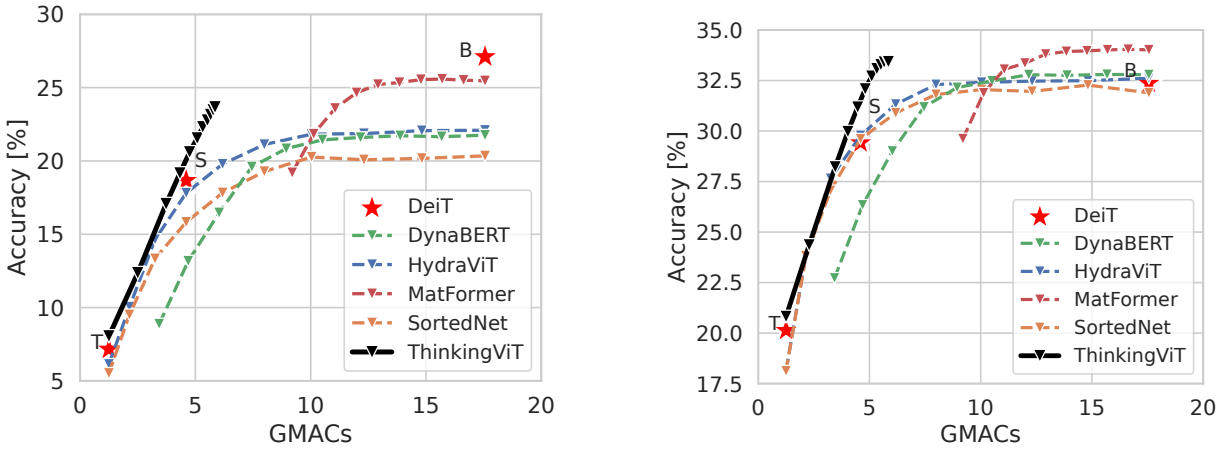


Figure 11. Accuracy versus GMACs on the ImageNet-1K validation set. All baseline models are based on DeiT-Small. *ThinkingViT* consistently outperforms these baselines by achieving higher accuracy at comparable or lower computational cost.

H. Results on ImageNet Variants

Figure 12 shows the full results for GMACs on ImageNet variants. Note that the GMACs for ImageNet-A and ImageNet-V2 are reported in Figure 7 and Figure 4a, respectively.



(a) ImageNet-A

(b) ImageNet-Sketch

Figure 12. Full results of *ThinkingViT* and baselines in terms of GMACs on ImageNet variants.

I. High-Resolution View of *ThinkingViT* Variants

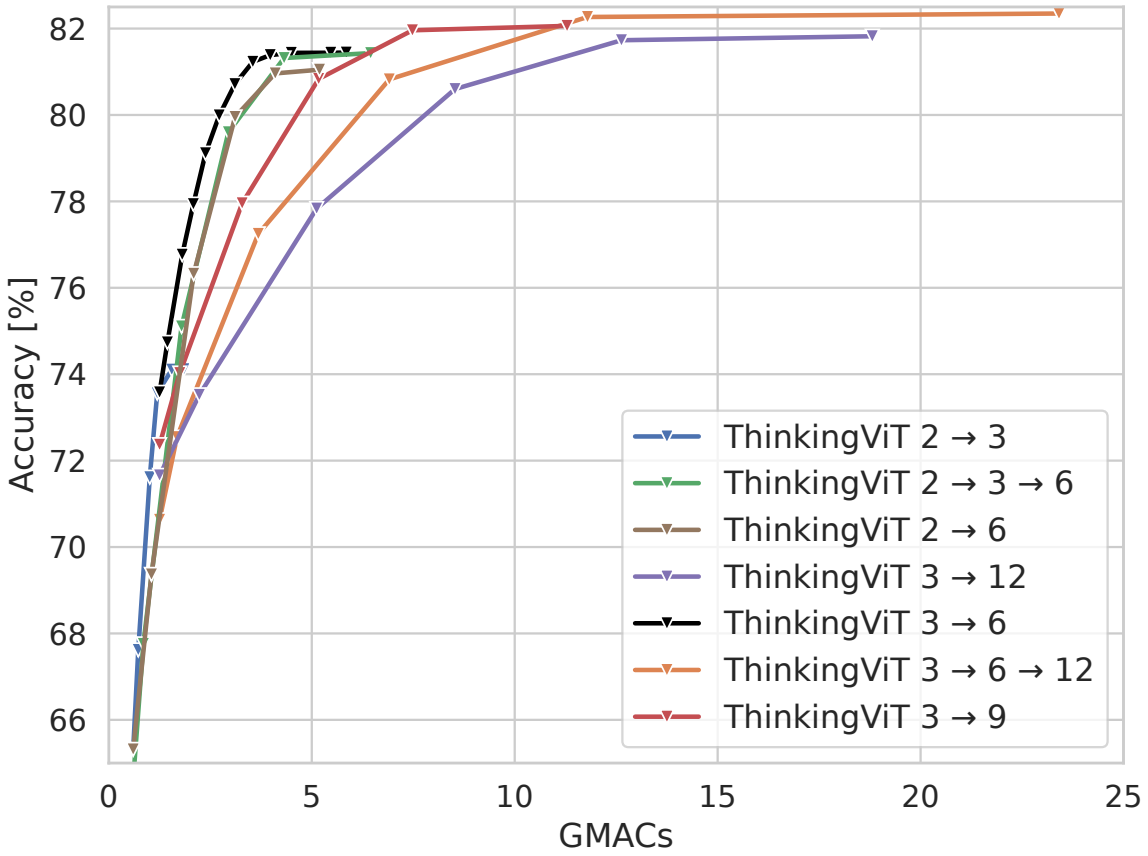


Figure 13. A higher-resolution version of Figure 3.

J. Further Limitations

In addition to the limitation discussed in Section 4.5, we highlight some additional limitations of our approach:

Training Overhead: Compared to training multiple standalone models, nested models like *ThinkingViT* incur higher training costs to reach similar performance levels across all stages. This is a known limitation of nested architectures, which share parameters and require joint optimization to maintain accuracy at varying compute levels.

Jumping Too Far Reduces Effectiveness: When the model transitions from a very small subset of attention heads (e.g., 3H) directly to a full configuration (e.g., 12H) in the second stage, performance suffers compared to using an intermediate stage (e.g., 9H) instead. This suggests that overly aggressive compute expansion can disrupt representational continuity, emphasizing the importance of smooth progression in staged inference.