A-ViT: Adaptive Tokens for Efficient Vision Transformer

Hongxu Yin Arash Vahdat Jose M. Alvarez Arun Mallya Jan Kautz Pavlo Molchanov
NVIDIA
{dannyy, avahdat, josea, amallya, jkautz, pmolchanov}@nvidia.com

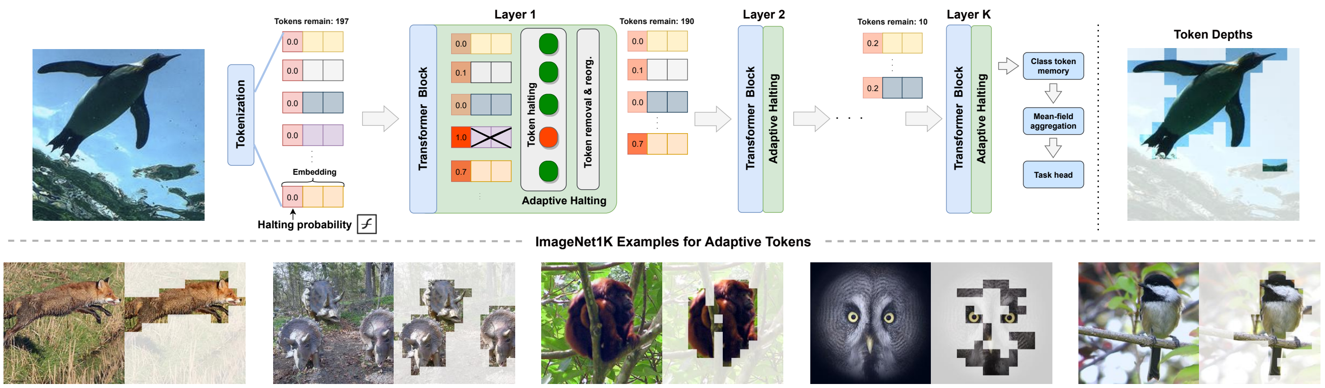


Figure 1. We introduce A-ViT, a method to enable ***adaptive token*** computation for vision transformers. We augment the vision transformer block with adaptive halting module that computes a halting probability per token. The module reuses the parameters of existing blocks and it borrows a single neuron from the last dense layer in each block to compute the halting probability, imposing no extra parameters or computations. A token is discarded once reaching the halting condition. Via adaptively halting tokens, we perform dense compute only on the active tokens deemed informative for the task. As a result, successive blocks in vision transformers gradually receive less tokens, leading to faster inference. Learnt token halting vary across images, yet align ***surprisingly well*** with image semantics (see examples above and more in Fig. 3). This results in immediate, out-of-the-box inference speedup on off-the-shelf computational platform.

Abstract

We introduce A-ViT, a method that adaptively adjusts the inference cost of vision transformer (ViT) for images of different complexity. A-ViT achieves this by automatically reducing the number of tokens in vision transformers that are processed in the network as inference proceeds. We reformulate Adaptive Computation Time (ACT [17]) for this task, extending halting to discard redundant spatial tokens. The appealing architectural properties of vision transformers enables our adaptive token reduction mechanism to speed up inference without modifying the network architecture or inference hardware. We demonstrate that A-ViT requires no extra parameters or sub-network for halting, as we base the learning of adaptive halting on the original network parameters. We further introduce distributional prior regularization that stabilizes training compared to prior ACT approaches. On the image classification task (ImageNet1K), we show that our proposed A-ViT yields high efficacy in filtering informative spatial features and cutting down on the overall compute. The proposed method improves the throughput of DeiT-Tiny by 62% *and DeiT-Small by* 38% *with only* 0.3% *accuracy drop, outperforming prior art by a large margin.*

1. Introduction

Transformers have emerged as a popular class of neural network architecture that computes network outputs using highly expressive attention mechanisms. Originated from the natural language processing (NLP) community, they have been shown effective in solving a wide range of problems in NLP, such as machine translation, representation learning, and question answering [2, 9, 22, 35, 44]. Recently, vision transformers have gained an increasing popularity in the vision community and they have been successfully applied to a broad range of vision applications, such as image classification [11, 16, 32, 43, 48, 55], object detection [3, 7, 39], image generation [20, 21], and semantic segmentation [28, 52]. The most popular paradigm remains when vision transformers form tokens via splitting an image into a series of ordered patches and perform inter-/intra-calculations between tokens to solve the underlying task. Processing an image with vision transformers remains computationally expensive, primarily due to the quadratic number of interactions between tokens [36, 40, 53]. Therefore, deploying vision transformers on data processing clusters or edge devices is challenging amid significant computational and memory resources.

The main focus of this paper is to study how to automati-

Project page at https://a-vit.github.io/.

cally adjust the compute in visions transformers as a function of the complexity of the input image. Almost all mainstream vision transformers have a fixed cost during inference that is independent from the input. However, the difficulty of a prediction task varies with the complexity of the input image. For example, classifying a car versus a human from a single image with a homogeneous background is relatively simple; while differentiating between different breeds of dogs on a complex background is more challenging. Even within a single image, the patches that contain detailed object features are far more informative compared to those from the background. Inspired by this, we develop a framework that adaptively adjusts the compute used in vision transformers based on the input.

The problem of input-dependent inference for neural networks has been studied in prior work. Graves [17] proposed adaptive computation time (ACT) to represent the output of the neural module as a mean-field model defined by a halting distribution. Such formulation relaxes the discrete halting problem to a continuous optimization problem that minimizes an upper bound on the total compute. Recently, stochastic methods were also applied to solve this problem, leveraging geometric-modelling of exit distribution to enable early halting of network layers [1]. Figurnov *et al.* [13] proposed a spatial extension of ACT that halts convolutional operations along the spatial cells rather than the residual layers. This approach does not lead to faster inference as high-performance hardware still relies on dense computations. However, we show that the vision transformer's uniform shape and tokenization enable an adaptive computation method to yield a direct speedup on off-the-shelf hardware, surpassing prior work in efficiency-accuracy tradeoff.

In this paper, we propose an input-dependent adaptive inference mechanism for vision transformers. A naive approach is to follow ACT, where the computation is halted for all tokens in a residual layer simultaneously. We observe that this approach reduces the compute by a small margin with an undesirable accuracy loss. To resolve this, we propose A-ViT, a spatially adaptive inference mechanism that halts the compute of different tokens at different depths, reserving compute for only discriminative tokens in a dynamic manner. Unlike point-wise ACT within convolutional feature maps [13], our spatial halting is directly supported by high-performance hardware since the halted tokens can be efficiently removed from the underlying computation. Moreover, entire halting mechanism can be learnt using existing parameters within the model, without introducing any extra parameters. We also propose a novel approach to target different computational budgets by enforcing a distributional prior on the halting probability. We empirically observe that the depth of the compute is highly correlated with the object semantics, indicating that our model can ignore less relevant background information (see quick examples in Fig. 1 and more examples in Fig. 3). Our proposed approach significantly cuts down the inference cost – A-ViT improves the throughput of DEIT-Tiny by 62% and DEIT-Small by 38% with only 0.3% accuracy drop on ImageNet1K.

Our main contributions are as follows:

- We introduce a method for input-dependent inference in vision transformers that allows us to halt the computation for different tokens at different depth.
- We base learning of adaptive token halting on the existent embedding dimensions in the original architecture and do not require extra parameters or compute for halting.
- We introduce distributional prior regularization to guide halting towards a specific distribution and average token depth that stabelizes ACT training.
- We analyze the depth of varying tokens across different images and provide insights into the attention mechanism of vision transformer.
- We empirically show that the proposed method improves throughput by up to 62% on hardware with minor drop in accuracy.

2. Related Work

There are a number of ways to improve the efficiency of transformers including weight sharing across transformer blocks [26], dynamically controlling the attention span of each token [5, 40], allowing the model to output the result in an earlier transformer block [38, 56], and applying pruning [53]. A number of methods have aimed at reducing the computationally complexity of transformers by reducing the quadratic interactions between tokens [6, 23, 24, 41, 47]. We focus on approaches related to adaptive inference that depends on the input image complexity. A more detailed analysis of the literature is present in [19].

Special architectures. One way is to change the architecture of the model to support adaptive computations [4, 14, 15, 18, 25, 27, 30, 37, 42, 51, 54]. For example, models that represent a neural network as a fixed-point function can have the property of adaptive computation by default. Such models compute the difference to the internal state and, when applied over multiple iterations, converge towards the solution (desired output). For example, neural ordinary differential equations (ODEs) use a new architecture with repetitive computation to learn the dynamics of the process [10]. Using ODEs requires a specific solver, is often slower than fix depth models and requires adding extra constraints on the model design. [54] learns a set of classifiers with different resolutions executed in order; computation stops when confidence of the model is above the threshold. [27] proposed a residual variant with shared weights and a halting mechanism.

Stochastic and reinforcement learning (RL) methods. The depth of a residual neural network can be changed during inference by skipping a subset of residual layers. This is possible since residual networks have the same input and output feature dimensions and they are known to perform feature refinements iteratively. Individual extra models can be learned on the top of a backbone to change the computational graph. A number of approaches [29, 34, 49, 50] proposed to train a separate network via RL to decide when to halt. These approaches require training of a dedicated halting model and their training is challenging due to the high-variance training signal in RL. Conv-AIG [45] learns conditional gating of residual blocks via Gumbel-softmax trick. [46] extends the idea to spatial dimension (pixel level).

Adaptive inference in vision transformers. With the increased popularity, researchers have very recently explored adaptive inference for vision transformers. DynamicViT [36] uses extra control gates that are trained with the Gumbel-softmax trick to halt tokens and it resembles some similarities to Conv-AIG [45] and [46]. Gumbel-softmax-based relaxation solutions might be sub-optimal due to the difficulty of regularization, stochasticity of training, and early convergence of the stochastic loss, requiring multi-stage token sparsification as a heuristic guidance. In this work, we approach the problem from a rather different perspective, and we study how an ACT [17]-like approach can be defined for spatially adaptive computation in vision transformers. We show complete viability to remove the need for the extra halting sub-networks, and we show that our models bring simultaneous efficiency, accuracy, and token-importance allocation improvements, as shown later.

3. A-ViT

Consider a vision transformer network that takes an image $x \in \mathcal{R}^{C\times H\times W}$ (C, H, and W represent channel, height, and width respectively) as input to make a prediction through:

$$y = \mathcal{C} \circ \mathcal{F}^L \circ \mathcal{F}^{L-1} \circ ... \circ \mathcal{F}^1 \circ \mathcal{E}(x), \tag{1}$$

where the encoding network $\mathcal{E}(\cdot)$ tokenizes the image patches from x into the positioned tokens $t \in \mathcal{R}^{K\times E}$, K being the total number of tokens and E the embedding dimension of each token. $\mathcal{C}(\cdot)$ post-processes the transformed class token after the entire stack, while the L intermediate transformer blocks $\mathcal{F}(\cdot)$ transform the input via self-attention. Consider the transformer block at layer l that transforms all tokens from layer $l-1$ via:

$$t^l_{1:K} = \mathcal{F}^l(t^{l-1}_{1:K}), \tag{2}$$

where $t^l_{1:K}$ denotes all the K updated token, with $t^0_{1:K} = \mathcal{E}(x)$. Note that the internal computation flow of transformer blocks $\mathcal{F}(\cdot)$ is such that the number of tokens K can be changed from a layer to another. This offers out-of-the-box computational gains when tokens are dropped due to the halting mechanism. Vision transformer [11, 43] utilizes a consistent feature dimension E for all tokens throughout layers. This makes it easy to learn and capture a *global* halting mechanism that monitors all layers in a joint manner. This also makes halting design easier for transformers compared to CNNs that require explicit handling of varying architectural dimensions, *e.g.*, number of channel, at different depths.

To halt tokens adaptively, we introduce an input-dependent halting score for each token as a halting probability h^l_k for a token k at layer l:

$$h^l_k = H(t^l_k), \tag{3}$$

where $H(\cdot)$ is a halting module. Akin to ACT [17], we enforce the halting score of each token h^l_k to be in the range $0 \leqslant h^l_k \leqslant 1$, and use accumulative importance to halt tokens as inference progresses into deeper layers. To this end, we conduct the token stopping when the cumulative halting score exceeds $1-\epsilon$:

$$N_k = \underset{n\leqslant L}{\text{argmin}} \sum_{l=1}^{n} h^l_k \geqslant 1-\epsilon, \tag{4}$$

where ϵ is a small positive constant that allows halting after one layer. To further alleviate any dependency on dynamically halted tokens between adjacent layers, we mask out a token t_k for all remaining depth $l > N_k$ once it is halted by (i) zeroing out the token value, and (ii) blocking its attention to other tokens, shielding its impact to $t^{l>N_k}$ in Eqn. 2. We define $h^L_{1:K} = \mathbf{1}$ to enforce stopping at the final layer for all tokens. Our token masking keeps the computational cost of our training iterations similar to the original vision transformer's training cost. However, at the inference time, we simply remove the halted tokens from computation to measure the actual speedup gained by our halting mechanism.

We incorporate $H(\cdot)$ into the existing vision transformer block by allocating a single neuron in the MLP layer to do the task. Therefore, we do not introduce any additional learnable parameters or compute for halting mechanism. More specifically, we observe that the embedding dimension E of each token spares sufficient capacity to accommodate learning of adaptive halting, enabling halting score calculation as:

$$H(t^l_k) = \sigma(\gamma \cdot t^l_{k,e} + \beta), \tag{5}$$

where $t^l_{k,e}$ indicates the e^{th} dimension of token t^l_k and $\sigma(u) = \frac{1}{1+\exp^{-u}}$ is the logistic sigmoid function. Above, β and γ are shifting and scaling parameters that adjust the embedding before applying the non-linearity. Note that these two scalar parameters are shared across all layers for all tokens. Only one entry of the embedding dimension E is used for halting score calculation. Empirically, we observe that the simple choice of $e = 0$ (the first dimension) performs well, while varying indices does not change the original performance, as we show later. As a result our halting mechanism does not introduce additional parameters or sub-network beyond the two scalar parameters β and γ.

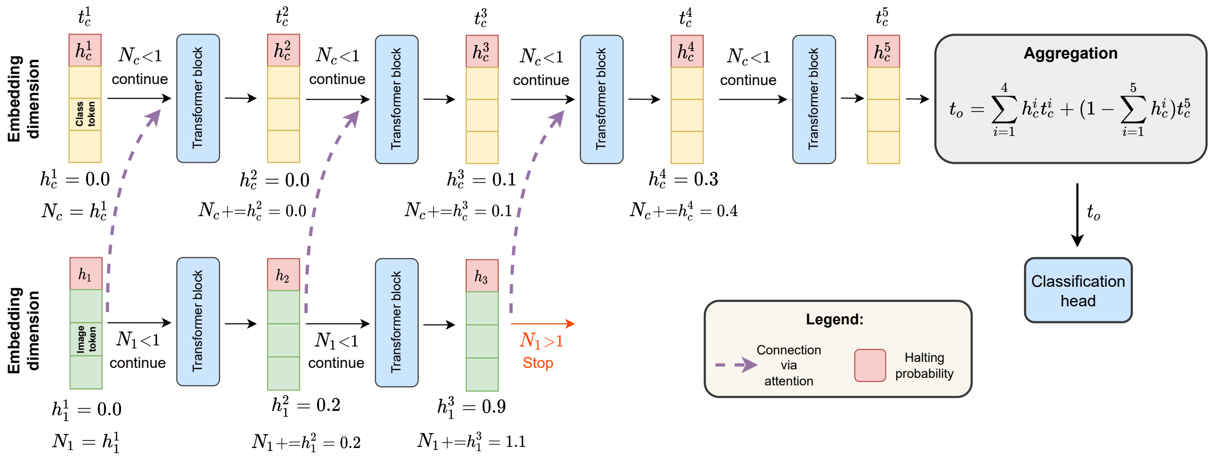


Figure 2. An example of A-ViT: In the visualization, we omit (i) other patch tokens, (ii) the attention between the class and patch token and (iii) residual connections for simplicity. The first element of every token is reserved for halting score calculation, adding no computation overhead. We denote the class token with a subscript c as it has a special treatment. Each token indexed by k has a separate N_k accumulator and stop at different depths. Unlike standard ACT, the mean-field formulation is applied only to the classification token, while other tokens contribute to the class token via attention. This allows adaptive token calculation without the aggregation of image/patch tokens.

To track progress of halting probabilities across layers, we calculate a remainder for each token as:

$$r_k = 1 - \sum_{l=1}^{N_k - 1} h_k^l, \tag{6}$$

that subsequent forms a halting probability as:

$$p_k^l = \begin{cases} 0 & \text{if } l > N_k, \\ r_k & \text{if } l = N_k, \\ h_k^l & \text{if } l < N_k. \end{cases} \tag{7}$$

Given the range of h and r, halting probability per token at each layer is always bounded $0 \leqslant p_k^l \leqslant 1$. The overall ponder loss to encourage early stopping is formulated via auxiliary variable r (reminder):

$$\mathcal{L}_{\text{ponder}} := \frac{1}{K} \sum_{k=1}^{K} \rho_k = \frac{1}{K} \sum_{k=1}^{K} (N_k + r_k), \tag{8}$$

where ponder loss ρ_k of each token is averaged. Vision transformers use a special class token t_k to produce the classification prediction, we denote it as t_c for future notations. This token similar to other input tokens is updated in all layers. We apply a mean-field formulation (halting-probability weighted average of previous states) to form the output token t_o and the associated task loss as:

$$\mathcal{L}_{\text{task}} = \mathcal{C}(t_o), \text{ where } t_o = \sum_{l=1}^{L} p_c^l t_c^l. \tag{9}$$

Our vision transformer can then be trained by minimizing:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{task}} + \alpha_{\text{p}} \mathcal{L}_{\text{ponder}}, \tag{10}$$

where α_{p} scales the pondering loss relative to the the main task loss. Algorithm 1 describes the overall computation flow, and Fig. 2 depicts the associated halting mechanism for visual explanation. At this stage, the objective function encourages an accuracy-efficiency trade-off when pondering different tokens at varying depths, enabling adaptive control.

One critical factor in Eqn. 10 is α_{p} that balances halting strength and network performance for the target application. A larger α_{p} value imposes a stronger penalty, and hence learns to halt tokens earlier. Despite efficacy towards computation reduction, prior work on adaptive computation [13, 17] have found that training can be sensitive to the choice of α_{p} and its value may not provide a fine-grain control over accuracy-efficiency trade-off. We empirically observe a similar behavior in vision transformers.

As a remedy, we introduce a distributional prior to regularize h^l such that tokens are expected to exit at a target depth on average, however, we still allow per-image variations. In this case for infinite number of input images we expect the the depth of token to vary within the distributional prior. Similar prior distribution has been recently shown effective to stablize convergence during stochastic pondering [1]. To this end, we define a halting score distribution:

$$\mathcal{H} := [\frac{\sum_{k=1}^{K} h_k^1}{K}, \frac{\sum_{k=1}^{K} h_k^2}{K}, ..., \frac{\sum_{k=1}^{K} h_k^L}{K}], \tag{11}$$

that averages expected halting score for all tokens across at each layer of network (*i.e.*, $\mathcal{H} \in \mathcal{R}^L$). Using this as an estimate of how halting likelihoods distribute across layers, we regularize this distribution towards a pre-defined prior using KL divergence. We form the new distributional prior regularization term as:

$$\mathcal{L}_{\text{distr.}} = \text{KL}(\mathcal{H} \,||\, \mathcal{H}^{\text{target}}), \tag{12}$$

Algorithm 1 Adaptive tokens in vision transformer without imposing extra parameters.

Input: tokenized input tensor **input** $\in \mathcal{R}^{K\times E}$, K, E being token number and embedding dimension; c is class-token index in K; $0 < \epsilon < 1$
Output: aggregated output tensor **out**, ponder loss ρ

```
t = input
cumul $= \mathbf{0}$                          ▷ Cumulative halting score
$\mathbf{R} = \mathbf{1}$                                ▷ Remainder value
out = 0                                  ▷ Output of the network
$\boldsymbol{\rho} = \mathbf{0}$                          ▷ Token ponder loss vector
$\mathbf{m} = \mathbf{1}$                     ▷ Token mask $\mathbf{m} \in \mathcal{R}^K$
for $l = 1 \ldots L$ do
    t = $\mathcal{F}^l(\mathbf{t} \odot \mathbf{m})$
    if $l < L$ then
        $\mathbf{h} = \boldsymbol{\sigma}(\gamma \cdot \mathbf{t}_{:,0} + \beta)$            ▷ $\mathbf{h} \in \mathcal{R}^K$
    else
        $\mathbf{h} = \mathbf{1}$
    end if
    cumul $+= \mathbf{h}$
    $\boldsymbol{\rho} += \mathbf{m}$                  ▷ Add one per remaining token
    for $k = 1, ..., K$ do
        if cumul$_k < 1 - \epsilon$ then
            $\mathbf{R}_k -= \mathbf{h}_k$
        else
            $\boldsymbol{\rho}_k += \mathbf{R}_k$
        end if
    end for
    if cumul$_c < 1 - \epsilon$ then
        out $+= \mathbf{t}_{c,:} \times \mathbf{h}_c$
    else
        out $+= \mathbf{t}_{c,:} \times \mathbf{R}_c$
    end if
    $\mathbf{m} \leftarrow$ cumul $< 1 - \epsilon$                ▷ Update mask
end for
return out, $\rho = \frac{\text{sum}(\boldsymbol{\rho})}{K}$
```

where KL refers to the Kullback-Leibler divergence, and $\mathcal{H}^{\text{target}}$ denotes a target halting score distribution with a guiding stopping layer. We use the probability density function of Gaussian distribution to define a bell-shaped distribution $\mathcal{H}^{\text{target}}$ in this paper, centered at the expected stopping depth N^{target}. Intuitively, this weakly encourages the expected sum of halting score for each token to trigger exit condition at N^{target}. This offers enhanced control of expected remaining compute, as we show later in experiments.

Our final loss function that trains the network parameters for adaptive token computation is formulated as:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{task}} + \alpha_{\text{d}}\mathcal{L}_{\text{distr.}} + \alpha_{\text{p}}\mathcal{L}_{\text{ponder}}, \qquad (13)$$

where α_{d} is a scalar coefficient that balances the distribution regularization against other loss terms.

4. Experiments

We evaluate our method for the classification task on the large-scale 1000-class ImageNet ILSVRC 2012 dataset [8] at the 224×224 pixel resolution. We first analyze the performance of adaptive tokens, both qualitatively and quantitatively. Then, we show the benefits of the proposed method over prior art, followed by a demonstration of direct throughput improvements of vision transformers on legacy hardware. Finally, we evaluate the different components of our proposed approach to validate our design choices.

Implementation details. We base A-ViT on the data-efficient vision transformer architecture (DeiT) [43] that includes 12 layers in total. Based on original training recipe[1], we train all models on only ImageNet1K dataset without auxiliary images. We use the default 16×16 patch resolution. For all experiments in this section, we use Adam for optimization (learning rate $1.5 \cdot 10^{-3}$) with cosine learning rate decay. For regularization constants we utilize $\alpha_{\text{d}} = 0.1, \alpha_{\text{p}} = 5 \cdot 10^{-4}$ to scale loss terms. We use $\gamma = 5, \beta = -10$ for sigmoid control gates $H(\cdot)$, shared across all layers. We use the embedding value at index $e = 0$ to represent the halting probability ($H(\cdot)$) for tokens. Starting from publicly available pretrained checkpoints, we finetune DeiT-T/S variant models for 100 epochs, respectively, to learn adaptive tokens without distillation. We denote the associated adaptive versions as A-ViT-T/S respectively. In what follows, we mainly use the A-ViT-T for ablations and analysis before showing efficiency improvements for both variants afterwards. We find that mixup is not compatible with adaptive inference, and we focus on classification without auxiliary distillation token – we remove both from finetuning. Applying our finetuning on the full DeiT-S and DeiT-T results in a top-1 accuracy of 78.9% and 71.3%, respectively. For training runs we use 8 NVIDIA V100 GPUs and automatic-mixed precision (AMP) [33] acceleration.

4.1. Analysis

Qualitative results. Fig. 3 visualizes the tokens' depth that is adaptively controlled during inference with our A-ViT-T over the ImageNet1K validation set. Remarkably, we observe that our adaptive token halting enables longer processing for highly discriminative and salient regions, often associated with the target class. Also, we observe a highly effective halting of relatively irrelevant tokens and their associated computations. For example, our approach on animal classes retains the eyes, textures, and colors from the target object and analyze them in full depth, while using fewer layers to process the background (*e.g.*, the sky around the bird, and ocean around sea animals). Note that even background tokens marked as not important still actively participate in

[1] Based on official repository at https://github.com/facebookresearch/DeiT.

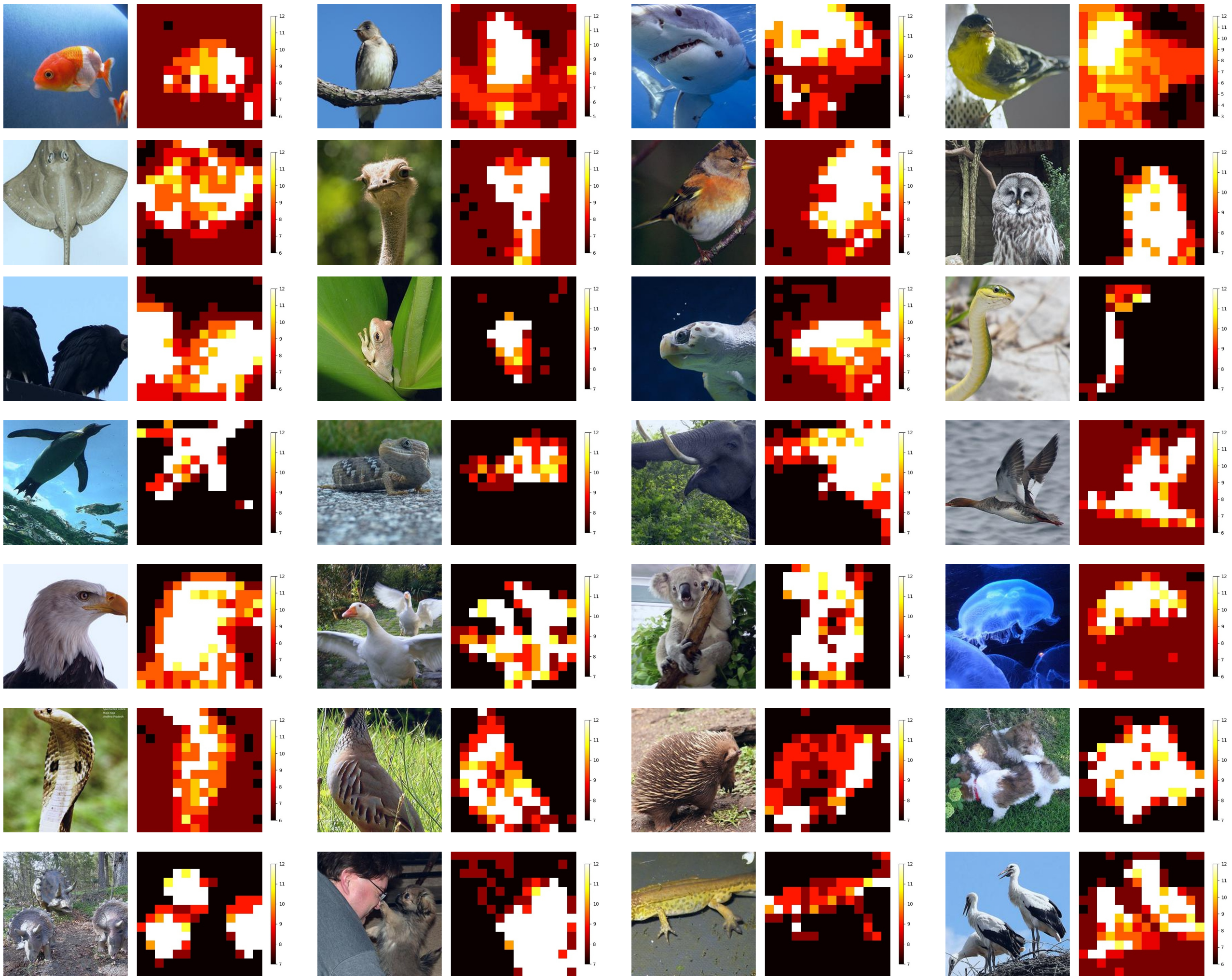
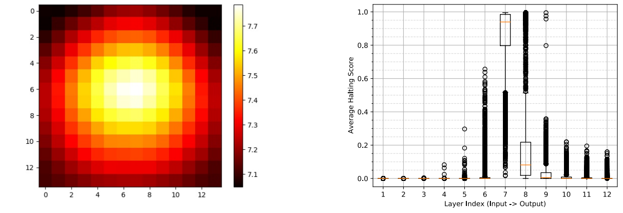


Figure 3. Original image (left) and the dynamic token depth (right) of A-ViT-T on the ImageNet-1K validation set. **Distribution of token computation highly aligns with visual features.** Tokens associated with informative regions are adaptively processed deeper, robust to repeating objects with complex backgrounds. Best viewed in color.

classification during initial layers. In addition, we also observe the inspiring fact that adaptive tokens can easily (i) keep track of repeating target objects, as shown in the first image of the last row in Fig. 3, and (ii) even shield irrelevant objects completely (see second image of last row).

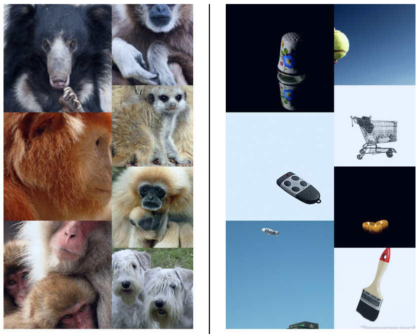
Token depth distribution. Given a complete distinct token distribution per image, we next analyze the dataset-level token importance distributions for additional insights. Fig. 4 (a) depicts the average depth of the learnt tokens over the validation set. It demonstrates a 2D Gaussian-like distribution that is centered at the image center. This is consistent with the fact that most ImageNet samples are centered, intuitively aligning with the image distribution. As a result, more compute is allocated on-the-fly to center areas, and computational cost on the sides is reduced.

Halting score distribution. To further evaluate the halting behavior across transformer layers, we plot the average layer-wise halting score distribution over 12 layers. Fig. 4 (b) shows box plots of halting scores averaged over all tokens per layer per image. The analysis is performed on 5K randomly sampled validation images. As expected, the halting score gradually increases at initial stages, peaks and then decreases



(a) (b)

Figure 4. (a) Average depth of tokens per image patch position for A-ViT-T on ImageNet-1K validation set. (b) Halting score distribution across the transformer blocks. Each point associated with one randomly sampled image, denoting average token score at that layer.



Hard samples. Easy samples.

Figure 5. Visual comparison of hard and easy samples from the ImageNet-1K validation set determined by average token depth. Note that all images above be *correctly classified* – only difference is that hard samples require more depths for tokens to process their semantic information. Tokens in the left images exit approximately 5 layers later compared to the right images.

for deeper layers.

Sharp-halting baseline. To further compare with static models of the same depth for performance gauging, we also train a DeiT-T with 8 layers as a sharp-halting baseline. We observe that our A-ViT-T outperforms this new baseline by $+1.4\%$ top-1 accuracy at a similar throughput. Although our adaptive regime is on average similarly shallow, it still inherits the expressivity of the original deeper network, as we observe that informative tokens are processed by deeper layers (*e.g.*, until 12$^{\text{th}}$ layer as in Fig. 3).

Easy and hard samples. We can analyse the difficulty of an image for the network by looking at the averaged depth of the adaptive tokens per image. Therefore, in Fig. 5, we depict hard and easy samples in terms of the required computation. Note, all samples in the figure are correctly classified, and only differ by the averaged token depth. We can observe that images with homogeneous background are relatively easy for classification, and A-ViT processes them much faster than hard samples. Hard samples represent images with informative visual features distributed over the entire image, and hence incur more computation.

Class-wise sensitivity. Given an adaptive inference paradigm, we analyze the change in classification accuracy for various classes with respect to the full model. In particular, we compute class-wise validation accuracy changes before and after applying adaptive inference. We summarize both qualitative and quantitative results in Table 1. We observe that originally very confident or uncertain samples are not affected by adaptive inference. Adaptive inference improves accuracy of the visually dominant classes such as individual furniture and animals.

Rank	**Class-wise Sensitivity to Adaptive Inference** static acc.→adaptive acc.		
	Favoring (acc. incr.)	Sensitive (acc. drop)	Stable
1	throne $_{56\rightarrow74\%}$	muzzle $_{58\rightarrow38\%}$	yellow lady-slipper $_{100\rightarrow100\%}$
2	lakeland terrier $_{64\rightarrow78\%}$	sewing machine $_{80\rightarrow62\%}$	leonberg $_{100\rightarrow100\%}$
3	cogi $_{60\rightarrow74\%}$	vaccume $_{37\rightarrow28\%}$	proboscis monkey $_{100\rightarrow100\%}$
4	african elephant $_{54\rightarrow68\%}$	flute $_{38\rightarrow20\%}$	velvet $_{10\rightarrow10\%}$
5	soft-coated wheaten terrier $_{68\rightarrow82\%}$	shovel $_{64\rightarrow46\%}$	laptop $_{14\rightarrow14\%}$

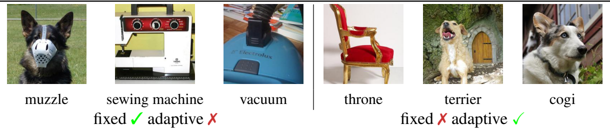


Table 1. Ranking of stable and sensitive classes to adaptive computation in A-ViT compared to fixed computation graph that executes the full model for inference. Sample images included for top three classes that favor or remain sensitive to adaptive computation.

4.2. Comparison to Prior Art

Next, we compare our method with previous work that study adaptive computation. For comprehensiveness, we systematically compare with five state-of-the-art halting mechanisms, covering both vision and NLP methods that tackle the dynamic inference problem from different perspectives: (i) adaptive computation time [17] as ACT reference applied on halting entire layers, (ii) confidence-based halting [31] that gauges on logits, (iii) similarity-based halting [12] that oversees layer-wise similarity, (iv) pondering-based halting [1] that exits based on stochastic halting-probabilities, and (v) the very recent DynamicViT [36] that learns halting decisions via Gumble-softmax relaxation. Details in appendix.

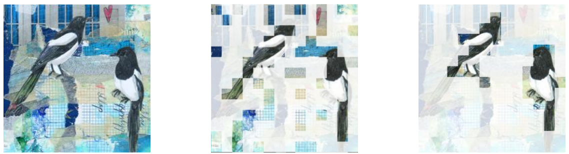
Performance comparison. We compare our results in Table 2 and demonstrate simultaneous performance improvements over prior art in having smaller averaged depth, smaller number of FLOPs and better classification accuracy. Notably our method involves no extra parameters, while cutting down FLOPs by 39% with only a minor loss of accuracy. To further visualize improvements over the state-of-the-art DynamicViT [36], we include Fig. 6 as a qualitative comparison of token depth for an official sample presented in the work. As noticed, A-ViT more effectively captures the important regions associated with the target objects, ignores the background tokens, and improves efficiency.

Note that both DynamicViT and A-ViT investigate adaptive tokens but from two different angles. DynamicViT utilizes Gumbel-Softmax to learn halting and incorporates a control for computation via a multi-stage token keeping ratio; it provides stronger guarantees on the latency by simply setting the ratio. A-ViT on the other hand takes a complete probabilistic approach to learn halting via ACT. This enables it to freely adjust computation, and hence capture enhanced semantic and improve accuracy, however requires a distributional prior and has a less intuitive hyper-parameter.

Hardware speedup. In Table 3, we compare speedup on off-the-shelf GPUs. See appendix for measurement details. In contrast to spatial ACT in CNNs that require extra computation flow and kernel re-writing [13], A-ViT enables speedups out of the box in vision transformers. With only 0.3% in accuracy drop, our method directly improves the throughputs of DeiT small and tiny variants by 38% and 62% without requiring hardware/library modification.

Method	Efficiency			Top-1 Acc. ↑
	Params. free	Avg. depth ↓	FLOPs ↓	
Baseline [43]	-	12.00	1.3G	71.3
ACT [17]	✗	10.01	1.0G	71.0
Confidence threshold [31]	✓	10.63	1.1G	65.8
Similarity gauging [12]	✓	10.68	1.1G	69.4
PonderNet [1]	✓	9.74	1.0G	66.2
DynamicViT [36]	✗	7.62	0.9G	70.9
Ours	✓	**7.23**	**0.8G**	**71.0**

Table 2. Comparison with prior art that studies dynamic inference halting mechanisms for transformers. Avg. depth specifies the mean depths of the tokens over the entire validation set.



Original DynamicViT (Rao *et al.* [36]) **Ours**

Figure 6. Visual comparison compared to prior art on token distribution for a sample taken from the public repository of DynamicViT by Rao *et al.* [36]. Only shaded (non-white) tokens are processed by all 12 layers. Our method better captures the semantics of the target class, drops more tokens, and saves more computation.

4.3. Ablations

Here, we perform ablations studies to evaluate each component in our method and validate their contributions.

Token-level ACT via $\mathcal{L}_{\text{ponder}}$. One noticeable distinction of this work from conventional ACT [17] is a full exploration of spatial redundancy in image patches, and hence their tokens. Comparing the first and last row in Table 2, we observe that our fine-grained pondering reduces token depths by roughly 3 layers, and results in 25% more FLOP reductions compared to the conventional ACT.

Distributional prior via $\mathcal{L}_{\text{distr.}}$. Incorporating the distributional prior allows us to better guide the expected token depth towards a target average depth, as seen in Fig. 7. As opposed to α_{p} that indirectly gauges on the remaining efficiency and usually suffers from over-/under-penalization, our distributional prior guides a quick convergence to a target depth level, and hence improves final accuracy. Note that a distributional prior complements the ponder loss in guiding overall halting towards a target depth, but it cannot capture remainder information – using ACT-agnostic distributional prior alone results in an accuracy drop of more than 2%.

"Free" embedding to learn halting. Next we justify the usage of a single value in the embedding vector for halting score computation and representation. In the embedding vectors, we set one entry at a random index to zero and analyze the associated accuracy drop without any finetuning of the model. Repeating 10 times for DeiT-T/S variants, the ImageNet1K top-1 accuracy only drops by $0.08\% \pm 0.04\%/0.04\% \pm 0.03\%$, respectively. This experiment demonstrates that one element in the vector can be used for another task with minimal impact on the original performance. In our experiments, we pick the first element in the vector and use it for the halting score computation.

Method	Efficiency		Top-1 Acc.↑	Throughput
	Params. ↓	FLOPs ↓		
ViT-B [11]	86M	17.6G	77.9	0.3K imgs/s
DeiT-S [43]	22M	4.6G	78.9	0.8K imgs/s
DynamicViT [36]	23M	3.4G	78.3	1.0K imgs/s
A-ViT-S	22M	3.6G	78.6	1.1K imgs/s
A-ViT-S + distl.	22M	3.6G	80.7	1.1K imgs/s
DeiT-T [43]	5M	1.2G	71.3	2.1K imgs/s
DynamicViT [36]	5.9M	0.9G	70.9	2.9K imgs/s
A-ViT-T	5M	0.8G	71.0	3.4K imgs/s
A-ViT-S + distl.	5M	0.8G	72.4	1.1K imgs/s

Table 3. Throughput improvement enabled via adaptive tokens. Models with **+ distil.** is augmented with distillation token.

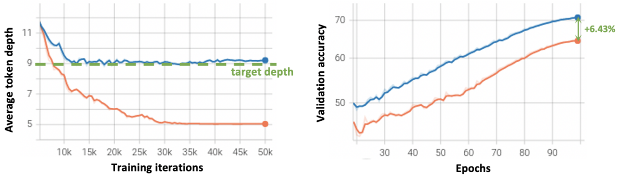


Figure 7. Training curves with (blue) and without (yellow) distributional priors towards a target depth of 9 layers. Both lines share the exact same training hyper-parameter set with the only difference in including the distributional prior guidance. As opposed to α_{p} that over-penalizes the networks, $\mathcal{L}_{\text{distr.}}$ guides a very fast convergence towards the target depth and yields a 6.4% accuracy gain.

Layer-wise networks to learn halting. We continue to examine viability to leverage extra networks for halting learning. To this end we add an extra two-layer learnable network (with input/hidden dimensions of 192/96, internal/output gates as `GeLU`/`Sigmoid`) on top of embeddings of each layer in A-ViT-T. We observed a very slight increase in accuracy of +0.06% with +0.2M parameter and −12.6% inference throughput overhead, as auxiliary nets have to be executed sequentially with ViT layers. Given this tradeoff, we base learning of halting on existing ViT parameters.

5. Limitations & Future Directions

In this work we primarily focused on the classification task. However, extension to other tasks such as video processing can be of great interest, given not only spatial but also temporal redundancy within input tokens.

6. Conclusions

We have introduced A-ViT to adaptively adjust the amount of token computation based on input complexity. We demonstrated that the method improves vision transformer throughput on hardware without imposing extra parameters or modifications of transformer blocks, outperforming prior dynamic approaches. Captured token importance distribution adaptively varies by input images, yet coincides surprisingly well with human perception, offering insights for future work to improve vision transformer efficiency.

References

[1] Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder. In *ICML Workshop*, 2021.

[2] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[4] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *NeurIPS*, 2018.

[5] Ting Chen, Ji Lin, Tian Lin, Song Han, Chong Wang, and Denny Zhou. Adaptive mixture of low-rank factorizations for compact neural modeling, 2019.

[6] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers. In *ICLR*, 2021.

[7] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. UpDETR: Unsupervised pre-training for object detection with transformers. In *CVPR*, 2021.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[10] Chengyu Dong, Liyuan Liu, Zichao Li, and Jingbo Shang. Towards adaptive residual network training: A neural-ode perspective. In *ICML*, 2020.

[11] 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*, 2021.

[12] Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. In *ICLR*, 2020.

[13] Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *CVPR*, 2017.

[14] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

[15] Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley Osher, and Wotao Yin. JFB: Jacobian-free backpropagation for implicit networks. *https://arxiv.org/abs/2103.12803*, 2021.

[16] Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. LeVit: A vision transformer in convnet's clothing for faster inference. In *ICCV*, 2021.

[17] Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

[18] Qiushan Guo, Zhipeng Yu, Yichao Wu, Ding Liang, Haoyu Qin, and Junjie Yan. Dynamic recursive neural network. In *CVPR*, 2019.

[19] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *TPAMI*, 2021.

[20] Drew A Hudson and C Lawrence Zitnick. Generative adversarial transformers. *arXiv preprint arXiv:2103.01209*, 2021.

[21] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. TransGAN: Two transformers can make one strong GAN. In *NeurIPS*, 2021.

[22] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling bert for natural language understanding. In *EMNLP*, 2020.

[23] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *ICML*, 2020.

[24] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020.

[25] Peter Kontschieder, Madalina Fiterau, Antonio Criminisi, and Samuel Rota Bulo. Deep neural decision forests. In *CVPR*, 2015.

[26] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*, 2020.

[27] Sam Leroux, Pieter Simoens, Bart Dhoedt, P Molchanov, T Breuel, and J Kautz. IamNN: Iterative and adaptive mobile neural network for efficient image classification. In *ICLR Workshop*, 2018.

[28] Zhiqi Li, Wenhai Wang, Enze Xie, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, Tong Lu, and Ping Luo. Panoptic segformer. *arXiv preprint arXiv:2109.03814*, 2021.

[29] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *NeurIPS*, 2017.

[30] Lanlan Liu and Jia Deng. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. In *AAAI*, 2018.

[31] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. FastBERT: A self-distilling bert with adaptive inference time. In *ACL*, 2020.

[32] 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.

[33] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *ICLR*, 2018.

[34] Augustus Odena, Dieterich Lawson, and Christopher Olah. Changing model behavior at test-time using reinforcement learning. *arXiv preprint arXiv:1702.07780*, 2017.

[35] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018.

[36] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021.

[37] Samuel Rota Bulo and Peter Kontschieder. Neural decision forests for semantic image labelling. In *CVPR*, 2014.

[38] Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A Smith. The right tool for the job: Matching model and instance complexities. In *ACL*, 2020.

[39] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris M Kitani. Rethinking transformer-based set prediction for object detection. In *ICCV*, 2021.

[40] Thierry Tambe, Coleman Hooper, Lillian Pentecost, Tianyu Jia, En-Yu Yang, Marco Donato, Victor Sanh, Paul Whatmough, Alexander M Rush, David Brooks, et al. EdgeBERT: Sentence-level energy optimizations for latency-aware multi-task nlp inference. In *MICRO*, 2020.

[41] Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *ICML*, 2020.

[42] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In *ICPR*, 2016.

[43] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[45] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *ECCV*, 2018.

[46] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *CVPR*, 2020.

[47] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[48] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[49] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. SkipNet: Learning dynamic routing in convolutional networks. In *ECCV*, 2018.

[50] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *CVPR*, 2018.

[51] Wenhan Xia, Hongxu Yin, Xiaoliang Dai, and Niraj K Jha. Fully dynamic inference with deep neural networks. *TETC*, 2021.

[52] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, 2021.

[53] Huanrui Yang, Hongxu Yin, Pavlo Molchanov, Hai Li, and Jan Kautz. NViT: Vision transformer compression and parameter redistribution. *arXiv preprint arXiv:2110.04869*, 2021.

[54] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *CVPR*, 2020.

[55] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token ViT: Training vision transformers from scratch on imagenet. In *ICCV*, 2021.

[56] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. BERT loses patience: Fast and robust inference with early exit. In *NeurIPS*, 2020.