# EXITS NEED ONLY CLASSIFY: EFFICIENT EARLY EXITING FOR VISION TRANSFORMERS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Anytime Early Exit (EE) Vision Transformers (ViT) adapt inference to image difficulty along an accuracy-speedup Pareto frontier. Transformer-based exits are the most accurate option but incur excessive per-exit computation and parameter overheads, so the field has turned to cheaper, weaker exits. We show this overhead is avoidable, based on one key insight: unlike an encoder layer that refines tokens for later layers, an EE classifier has one job—to pool and summarize global class information. Three optimizations follow directly. First, because only the `[CLS]`-token is relevant for image classification, an exit can compute the `[CLS]`-token update *exactly* at a fraction of a full encoder update's cost. Second, image patches that are less relevant to classification can be pruned, trading small accuracy losses for additional speedup. Third, because all exits have this one objective, carefully constructed weight sharing allows for reduced exit parameters with small accuracy loss. We instantiate these in Class-only ANytime image Transformers (CANiT) on a frozen ViT backbone. On ImageNet-1k, CANiT improves accuracy over prior EE work by 0.88–3.35 percentage points at matched speedup, with similar gains across three additional datasets. Ablations isolate the contributions of `[CLS]`-token-only computation, patch pruning, and weight sharing.

## 1 INTRODUCTION

Not all images require equal resources for correct classification (Wang et al., 2021; Yin et al., 2022; Liang et al., 2022). While extensive calculations are needed to correctly classify "hard" images, fewer resources are sufficient for "easy" images[1]. Early Exit (EE) networks (Figure 1a) leverage this insight through numerous intermediate EE classifiers (Teerapittayanon et al., 2017; Hu et al., 2019), terminating early at an EE classifier when an image's prediction confidence exceeds a threshold. EE networks are anytime, i.e., capable of trading inference accuracy for inference latency along a Pareto frontier to meet fluctuating demands without any retraining (Zilberstein, 1996).

MLP-based exits — the most common EE classifier architecture in prior work — attach a lightweight MLP to the `[CLS]` token without pooling information from other tokens. Because they lack their own pooling mechanisms, they perform poorly on a frozen backbone (Table 1, MLP-EE (Bakhtiarnia et al., 2021)). To compensate, existing solutions fine-tune the backbone encoders to conduct the pooling instead (Table 1, EnViT (Dong et al., 2026)). This solution improves shallow exits but degrades later ones because the backbone is modified from its original objective.

Transformer-based exits, on the other hand, pool global class information using Multi-Headed Self Attention (MHSA) and demonstrate superior exit-wise accuracy without fine-tuning the backbone (Table 1, ViT-EE (Bakhtiarnia et al., 2021)). However, Transformer-based exits require substantially more parameters and fail to achieve the speedups of MLP-based approaches (Figure 2). These shortcomings are due to naive application of Transformer encoders as intermediate EE classifiers.

In short, *the task of all exits should be pooling global classification information without performing unnecessary computation*. We identify three key principles based on this observation. First,

[1]Following prior work (Huang et al., 2018; Yang et al., 2020; Wang et al., 2021), we treat image difficulty as a property of the (image, anytime model) pair rather than the image alone: for correct, high-confidence classification, an image is "easy" if an earlier exit is sufficient and "hard" if deeper layers are required.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

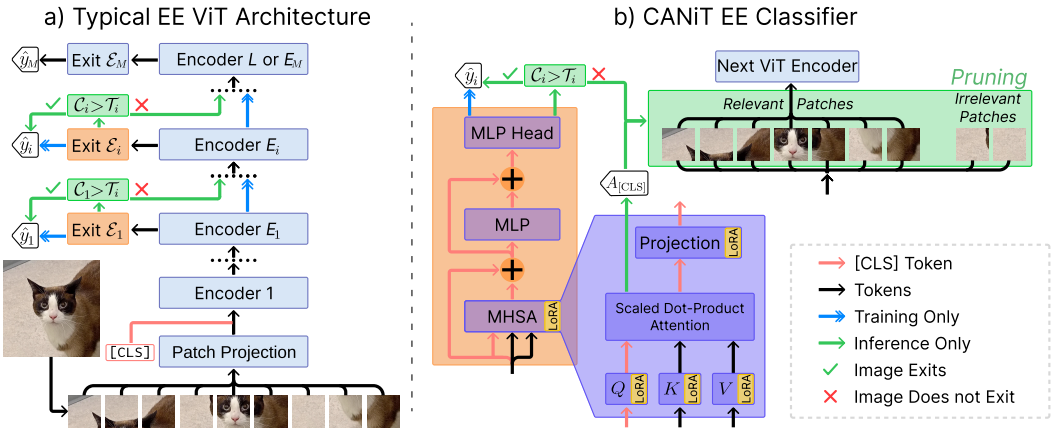



Figure 1: Color denotes layer attributes: ViT Backbone, Early Exit Classifier, CANiT layers. **a)** A typical early exit (EE) ViT with $L$ backbone encoders and $M$ exit classifiers. During training, all $M$ exits are evaluated and optimized simultaneously. During inference, a sample exits early with label $\arg\max \hat{\boldsymbol{y}}_i$ at exit $\mathcal{E}_i$ if its prediction confidence ($\mathcal{C}_i$) exceeds a pre-determined threshold ($\mathcal{T}_i$), i.e., $\mathcal{C}_i > \mathcal{T}_i$. Otherwise, computation proceeds directly to the next backbone encoder layer. **b)** CANiT's attention-based EE classifier. CANiT's encoders compute only `[CLS]`-token updates during training and inference (Section 3.1, Eqs. 6 & 7). During inference, CANiT trades small accuracy losses for additional speedups by pruning tokens with the lowest exit attention scores ($\boldsymbol{A}_{\texttt{[CLS]}}$) before proceeding to the next layer (Section 3.2). Weight sharing and per-exit LoRA reduce EE parameters with small accuracy loss (Section 3.3, Eq. 8).

multi-headed attention is ideal for global pooling, and because the exit only needs to produce a classification output, any computation not serving that output is redundant and can be dropped without changing the result. Second, an exit's `[CLS]`-token attention scores, computed at each exit, also signal each patch's relevance; irrelevant patches can be pruned to trade a small accuracy loss for speedup. Finally, all EE classifiers must transform intermediate feature representations at different depths to a shared output label space. Under this shared regime, classifiers may share exit weights with reasonable impacts on exit accuracy.

Class-only ANytime image Transformers (CANiT) applies these principles to a frozen backbone (Figure 1b). Restricting attention to `[CLS]` token alone reduces the per-exit FLOPs by $\approx 84\%$ with no change in output. Pruning patches by the `[CLS]`-token's exit attention score—already computed at the exit—adds an inference-time knob that further extends CANiT's Pareto frontier. CANiT-P, our parameter efficient variant, reduces parameter overheads by $\approx 75\%$ through low-rank adaptation (LoRA) (Hu et al., 2021) and weight sharing with a small impact on the Pareto frontier.

With these optimizations, CANiT and CANiT-P dominate the Pareto frontier across datasets (Figure 3). Extensive ablations isolate the contributions of `[CLS]`-token-only computation, patch pruning, and weight sharing. In summary, our main contributions are as follows:

1. **Exits need only classify.** We identify that EE classifiers have narrower requirements than the backbone: they only have to produce class labels. From this observation, three principles follow: computation that does not serve classification can be losslessly removed; input patches less relevant to classification can be pruned to trade accuracy for speed; and exits share the same objective so they can also share weights.
2. **CANiT.** We realize these principles on a *frozen* backbone, preserving its peak accuracy. `[CLS]`-token-only computation cuts exit FLOPs by ≈84%, with no change in output. Per-exit LoRA enables weight-sharing across exits. Finally, we vary the pruning rate at inference as an additional accuracy-speedup configuration parameter.
3. **Pareto dominance across datasets.** CANiT dominates the accuracy-speedup frontier, improving accuracy compared to prior EE work by 0.88–3.35 percentage points on ImageNet-1k and up to 3.91 on three additional datasets. We show—through batched-inference analysis rarely reported for EE ViTs—that throughput closely tracks FLOP gains.

| Methods | Backbone Fine-tuned | EE Arch. | Param. Overhead | $i$-th Exit Accuracy (%) 1 | 3 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| MLP-EE | × | MLP | 0.4M$_{(+0.5\%)}$ | 17.03 | 50.30 | 89.94 | **91.60** |
| EnViT$^{\dagger}$ | ✓ | MLP | 0.4M$_{(+0.5\%)}$ | 61.94 | 84.67 | 88.85 | 89.94 |
| ViT-EE | × | Encoder | 35.8M$_{(+41.4\%)}$ | **67.26** | **85.72** | **91.81** | **91.60** |

Table 1: Comparison between EE architectures on CIFAR-100. $^{\dagger}$ denotes author reported accuracies. Param. overhead shows absolute and relative (to the backbone) increase in parameters. Joint training of MLP-based exits and backbone improves accuracy at shallow exits but sacrifices accuracy of later exits. Transformer-based exits achieve superior per-exit accuracy without joint training but require more parameters and are not Pareto optimal.

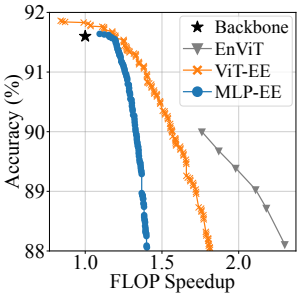



Figure 2: Pareto frontier of EE architectures shown in Table 1 on CIFAR-100.

## 2 Related Work

**Early Exits.** EE networks have been shown to significantly reduce compute and latency requirements of deep neural networks compared to static baselines (Huang et al., 2018).

A body of work concern the exit architecture and training recipe. SDN (Kaya et al., 2018) employs a weighted training strategy to prioritize different exits and BYOT (Zhang et al., 2019) proposes self-distillation to improve exit accuracy. Bakhtiarnia et al. (2021) explore numerous classifier architectures (e.g., MLP-EE and ViT-EE) on frozen and fine-tuned backbones. LGViT (Xu et al., 2023) combines convolutional- and Transformer-based exits for better Pareto performance on a frozen backbone. EnViT (Dong et al., 2026) investigates exit-aware structured dropout on a fine-tuned backbone to improve shallow exit performance without compromising later exits. Across these works, lightweight exits minimize overhead but provide limited global aggregation, while Transformer-based exits improve accuracy at a much greater computational and parameter cost.

Prior work has also proposed alternative exit criteria, including patience-based decisions (Zhou et al., 2020), aggregated confidence in BEEM (Bajpai & Hanawal, 2025), calibrated sample weighting in COSEE (He et al., 2025), and risk control (Jazbec et al., 2024). CANiT uses confidence-based exiting, but the exit decision criterion is orthogonal to the rest of our contributions.

**Adaptive Token Computation.** An alternative class of strategies reduces computational requirements without early exits by decreasing the image patch cardinality prior to inference (Chen et al., 2023; Wang et al., 2021) or online (Liang et al., 2022; Yin et al., 2022; Liu et al., 2024; Rao et al., 2021). He et al. (2021) compose EE and patch pruning in the NLP domain by comparing pairwise importance across all tokens and learning the optimal exiting threshold and pruning rate online. Liu et al. (2024) employ early exits during training to improve token scoring and overall performance but do not exit early, disabling early exits during inference to prevent runtime or parameter overheads.

More recent token-reduction methods have reduced or eliminated the need for retraining. Adaptive Token Sampling (ATS) (Fayyaz et al., 2022) adaptively varies the number of tokens for each input and can be applied to pretrained ViTs with no additional tuning. Token Merging (ToMe) (Bolya et al., 2023) also speeds up pretrained ViTs without retraining by merging redundant tokens. Zero-TPrune (Wang et al., 2024) performs zero-shot token pruning by ranking tokens from the pretrained attention graph, and can switch configurations at the start of inference without retraining. These methods demonstrate that token computation reduction can be applied while preserving the backbone. However, none of these techniques provide early-exit classification. In all cases, inputs continue through the full backbone depth using the particular token-reduction mechanism.

**Class-focused Computation.** CaiT (Touvron et al., 2021b) introduces class-attention layers where all tokens are attended but only the `[CLS]`-token is updated. We make a related, but separate observation for early-exit branches: because only the `[CLS]`-token is used by the classifier, the patch-token outputs of a conventional Transformer encoder are not needed. CANiT therefore omits those patch-token updates while exactly preserving the classifier output.

Because all exits perform the same classification task, there is an additional opportunity to reduce parameter overhead through parameter sharing and low-rank adaptation (Hu et al., 2021). CANiT uses shared exit weights with lightweight per-exit adaptation to reduce the parameter duplication otherwise incurred by multiple Transformer exits.

**Summary.** Prior work has proposed depth adaptation through early exits, width adaptation through token reduction, and class-focused computation for efficient aggregation. CANiT instead takes the functional requirements of the early-exit classifier as its starting point. This perspective exposes three sources of Transformer-exit overhead: computation within the exit that cannot affect its prediction, token computation for samples that continue through the backbone, and over-parameterized exits. CANiT addresses these together to retain the accuracy advantages of Transformer exits while improving the accuracy–speedup Pareto frontier.

## 3 Class-only ANytime image Transformers

**Preliminaries.** We consider the prevalent ViT architecture of Dosovitskiy et al. (2021) with $L$ encoders. An image $x$ with label $y$ is projected into the patch embeddings $\boldsymbol{X}_{patch} \in \mathbb{R}^{N \times E}$ ($N$ and $E$ are the patch cardinality and embedding dimension) and a learnable embedding $\boldsymbol{x}_{\texttt{[CLS]}} \in \mathbb{R}^E$ (i.e., `[CLS]` token) is prepended to patches. Let $\boldsymbol{Z}^{(\ell)} \in \mathbb{R}^{(N+1) \times E}$ and $\boldsymbol{Z}_0^{(\ell)} \in \mathbb{R}^E$ denote the feature representation and `[CLS]` token after layer $\ell \in \{1, \ldots, L\}$, with $\boldsymbol{Z}^{(0)} = [\boldsymbol{x}_{\texttt{[CLS]}}; \boldsymbol{X}_{patch}]$. Token features are refined through $L$ consecutive encoders consisting of a $\texttt{MHSA}$ layer and $\texttt{MLP}_{enc}$ layers (Eqs. 1&2) and class predictions $\hat{\boldsymbol{y}}$ are produced by applying an $\texttt{MLP}_{head}$ classification head to $L$-th `[CLS]` token (Eq. 3). $\texttt{MLP}_{enc}$ contains two feedfoward layers with a GELU non-linearity. The classification $\texttt{MLP}_{head}$ is a single layer. Layernorm (LN) is applied to all inputs.

$$\boldsymbol{Z}'^{(\ell)} = \texttt{MHSA}\left(\texttt{LN}\left(\boldsymbol{Z}^{(\ell-1)}\right)\right) + \boldsymbol{Z}^{(\ell-1)} \tag{1}$$

$$\boldsymbol{Z}^{(\ell)} = \texttt{MLP}_{enc}\left(\texttt{LN}\left(\boldsymbol{Z}'^{(\ell)}\right)\right) + \boldsymbol{Z}'^{(\ell)} \tag{2}$$

$$\hat{\boldsymbol{y}} = \texttt{MLP}_{head}\left(\texttt{LN}\left(\boldsymbol{Z}_0^{(L)}\right)\right) \tag{3}$$

We refine this formulation for the EE setting. Let $M$ denote the number of exits, each located after backbone encoders $E_1, \ldots, E_M$ with $E_1 < \cdots < E_M = L$. Then exit prediction $\hat{\boldsymbol{y}}_i$ and its confidence $\mathcal{C}_i$ are defined as follows,

$$\hat{\boldsymbol{y}}_i = \mathcal{E}_i\left(\boldsymbol{Z}^{(E_i)}\right), \; \hat{\boldsymbol{y}}_M = \texttt{MLP}_{head}\left(\texttt{LN}\left(\boldsymbol{Z}_0^{(L)}\right)\right) \tag{4}$$

$$\mathcal{C}_i = \max \mathcal{P}(\hat{\boldsymbol{y}}_i) \tag{5}$$

where $\mathcal{E}_i(\cdot)$ denotes the $i$-he exit head and $\mathcal{P}(\cdot)$ is the SOFTMAX function.

### 3.1 Exit Architecture

As discussed, ViT-EE exhibits high accuracy but is computationally expensive during inference. Critically, ViT-EE's encoders refine both patch and `[CLS]` tokens. While early exit classification benefits from pooling global class information, it needs only refine the `[CLS]`-token.

To this end, CANiT's Transformer-based exit classifiers attend to all input tokens but refine only the `[CLS]`-token. Let $Q_i, K_i, V_i, P_i$ denote the projection functions of the $i$-th exit's MHSA with $h$ attention heads of dimension $d$. CANiT calculates key and value matrices for all tokens but queries are only calculated for the `[CLS]`-token (Eq. 6). The resultant exit attention score $\boldsymbol{A}_{\texttt{[CLS]},i} \in \mathbb{R}^{h \times (N+1)}$ (Eq. 6) and refined exit token $\boldsymbol{Z}^{*(E_i)} \in \mathbb{R}^{1 \times E}$ (Eq. 7) are defined as follows,

$$\boldsymbol{A}_{\texttt{[CLS]},i} = \mathcal{P}\left(\left(Q_i\left(\texttt{LN}\left(\boldsymbol{Z}_0^{(E_i)}\right)\right) K_i\left(\texttt{LN}\left(\boldsymbol{Z}^{(E_i)}\right)\right)^{\mathsf{T}}\right)/\sqrt{d}\right) \tag{6}$$

$$\boldsymbol{Z}^{*(E_i)} = P_i\left(\boldsymbol{A}_{\texttt{[CLS]},i}\, V_i\left(\texttt{LN}\left(\boldsymbol{Z}^{(E_i)}\right)\right)\right) + \boldsymbol{Z}_0^{(E_i)} \tag{7}$$

Exit prediction $\hat{\boldsymbol{y}}_i$ is obtained by applying $\texttt{MLP}_{enc}$ and $\texttt{MLP}_{head}$ to $\boldsymbol{Z}^{*(E_i)}$ (similar to Eqs. 2 & 3). This computation is exact. Only the `[CLS]`-token reaches the $\texttt{MLP}_{head}$ and `LN` and $\texttt{MLP}_{enc}$ act on each token independently, so the patch token updates that a full encoder layer would execute do not influence $\hat{\boldsymbol{y}}_i$. Eqs. 6 & 7 therefore produce the `[CLS]` of a full encoder layer while omitting the $N$ patch-token queries and their `MLP` updates. Section 4.3 demonstrates empirically that this `[CLS]`-token-only exiting reduces FLOP overheads by $\approx 84\%$ and confirms that accuracy is unaffected.

### 3.2 Post-hoc Patch Pruning

CANiT exits early if an image's prediction confidence $\mathcal{C}_i$ at exit $E_i$ exceeds threshold $\mathcal{T}_i$. Otherwise, CANiT prunes image patches before proceeding to the next backbone layer. Pruning reduces computation in all subsequent layers but costs accuracy, so CANiT removes only the patches least relevant to classification. The exit already provides the necessary information for this as its `[CLS]`-token based exit attention score $\boldsymbol{A}_{\texttt{[CLS]},i}$ is trained to pool class-relevant information at that exit. Thus, its scores are a direct signal of each patch's relevance to classification. This follows the observation of EViT (Liang et al., 2022) that the attention scores signal patch importance, but CANiT uses the exit's own attention rather than the backbone's (as done by EViT).

Let $\boldsymbol{Z}^{(E_i)} \in R^{(N+1)\times E}$ and $\kappa$ denote input feature representation for exit $E_i$ and the global patch pruning rate across all exits. To prune, we average the class attention scores across all attention heads and drop the $\lfloor \kappa \cdot N \rfloor$ image patches corresponding to the lowest averaged scores. Patches are dropped for every image at every intermediate exit as long as the image has not exited early. During inference, all exits prune patches at the same pruning rate $\kappa$; however, to trade off accuracy for speedup, we sweep $\kappa$ jointly with $\mathcal{T}_i$ across configurations. Section 4.2 compares CANiT to prior patch pruning methods; Section 4.3 isolates patch pruning's contributions.

### 3.3 Weight Sharing & Per-Exit LoRA

All EE classifiers transform intermediate feature representations from varying depths to a shared output label space. Thus, while exit weights may be shared across exits, sharing could degrade per-exit accuracy. To minimize this degradation, CANiT-P shares all exit weights except two: per-exit LoRA (Hu et al., 2021) applied to `MHSA`'s feedfoward layers and layernorm affine transformation weights. These per-exit weights balance parameter efficiency and accuracy by adapting each exit's transformation to the exit's respective input representation. Our ablations (Section 4.3) indicate that CANiT-P achieves a reasonable trade-off between accuracy and parameter efficiency.

Let $\boldsymbol{W}_s \in \mathbb{R}^{n\times p}$ denote the weight matrix of a shared feedfoward layer. We introduce per-exit weight matrices $B_i \in \mathbb{R}^{n\times r}$ and $C_i \in \mathbb{R}^{r\times p}$, where the rank $r \ll \min(n, p)$. Eq. 8 outlines the modified operation of the shared feedfoward layer. We use a rank $r = 32$ for our experiments.

$$\mathcal{F}_{\text{LoRA,i}}(\boldsymbol{Z}) = \boldsymbol{W}_s\boldsymbol{Z} + \boldsymbol{B}_i\boldsymbol{C}_i\boldsymbol{Z} \tag{8}$$

### 3.4 Training

We attach CANiT's exit encoders to a frozen pretrained backbone and train them using a combination of two loss functions. First, we minimize the cross entropy loss between all exit predictions and the ground truth label (Eq. 9). Second, similar to prior work (Liu et al., 2020; Xu et al., 2023), predictions from the preexisting final exit (i.e., exit $E_M$) guide the appended EE heads through soft distillation via the Kullback-Leibler divergence loss (Eq. 10). The (hyperparameterized) $\lambda$ balances the EE head loss between the ground truth and soft distillation and $\tau$ is the temperature of the distillation. Section 4.3 evaluates the impact of soft distillation.

$$\mathcal{L} = \mathcal{L}_{CE}(\hat{\boldsymbol{y}}_M, y) + \sum_{m=1}^{M-1} \lambda\, \mathcal{L}_{CE}(\hat{\boldsymbol{y}}_m, y) \qquad \text{Ground truth loss} \tag{9}$$

$$+ \sum_{m=1}^{M-1} (1-\lambda)\tau^2\, \text{KL}(\mathcal{P}(\hat{\boldsymbol{y}}_m/\tau), \mathcal{P}(\hat{\boldsymbol{y}}_M/\tau)) \qquad \text{Soft distillation for EE heads} \tag{10}$$

Table 2: EE Comparison. † denotes accuracy and speedup reported by Dong et al. (2026). CANiT and CANiT-P exhibit the **highest** and second highest EE accuracy and FLOP speedup.

| Methods | ImageNet-1k | | CIFAR-100 | | Foods-101 | | Flowers-102 | |
|---|---|---|---|---|---|---|---|---|
| | Acc. | Speedup | Acc. | Speedup | Acc. | Speedup | Acc. | Speedup |
| Prior Backbone† | 81.63% | 1.00x | 91.66% | 1.00x | 91.19% | 1.00x | 97.67% | 1.00x |
| SDN† | 78.56% | 1.68x | 88.12% | 1.86x | 88.45% | 2.12x | 95.07% | 2.15x |
| BYOT† | 77.50% | 1.63x | 87.20% | 1.82x | 87.93% | 2.03x | 94.89% | 2.00x |
| LGViT† | 78.61% | 1.51x | 88.14% | 1.94x | 88.46% | 1.97x | 93.14% | 1.63x |
| BEEM† | 76.54% | 1.69x | 88.87% | 1.66x | 89.15% | 2.06x | 93.04% | 2.14x |
| COSEE† | 76.67% | 1.65x | 87.50% | 1.84x | 87.78% | 1.89x | 91.96% | 1.48x |
| EnViT† | 80.45% | 1.72x | 89.23% | 2.05x | 89.15% | 2.20x | 95.33% | 2.23x |
| Our Backbone | 81.80% | 1.00x | 91.60% | 1.00x | 91.31% | 1.00x | 96.65% | 1.00x |
| ViT-EE | 80.18% | 1.30x | 88.21% | 1.83x | 89.15% | 2.14x | 94.34% | 2.22x |
| CANiT | **81.37%** | **1.75x** | **91.11%** | **2.12x** | **91.06%** | **2.99x** | **96.45%** | **2.81x** |
| CANiT-P | 80.97% | 1.74x | 90.95% | 2.09x | 91.00% | 2.95x | 96.44% | 2.71x |

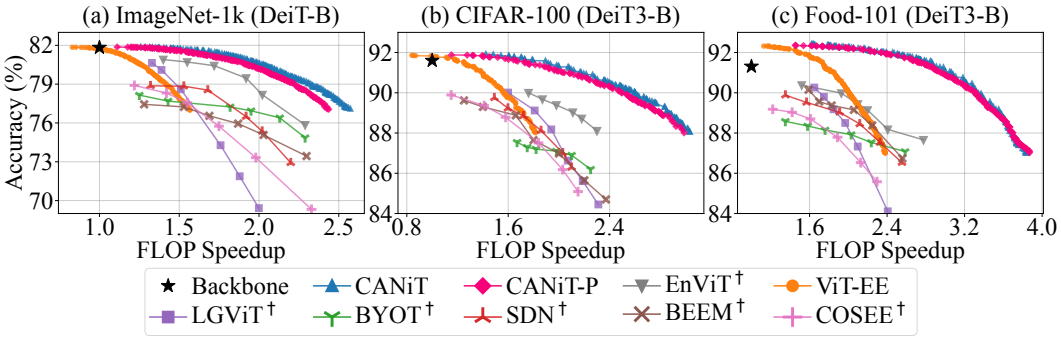



Figure 3: CANiT's accuracy-speedup Pareto frontier dominates prior baselines across datasets. † denotes accuracy and speedup reported by Dong et al. (2026).

## 4 Experiments

We evaluate CANiT on four standard image datasets: ImageNet-1k (Deng et al., 2009), CIFAR-100 (Krizhevsky, 2009), Flowers-102 (Nilsback & Zisserman, 2008), and Food-101 (Bossard et al., 2014). Our ablations isolate the contributions of `[CLS]` only exiting, patch pruning, and weight sharing. Implementation and reproducibility details are provided in Appendices B & C.

**EE Configuration and Baselines.** Following Dong et al. (2026), we place intermediate EE classifiers after every other layer and obtain accuracy-compute Pareto frontiers by adjusting $T_1 \dots T_E$ & $\kappa$. We compare CANiT to EE baselines reported by Dong et al. (2026).[2] Their backbones are not publicly released, so we use the closest match for each dataset. For ImageNet-1k, we use DeiT-B (Touvron et al., 2021a) with accuracy 81.80%, which nearly matches theirs (81.63%). For other datasets, our fine-tuned DeiT3-B (Touvron et al., 2022) models were the closest match to EnViT's backbone.[3] Table 2 reports the backbone accuracies from prior work and our models.

**Pruning Configuration and Baselines.** For fair evaluation, we compare CANiT to baselines not fine-tuned for pruning as reported by Wang et al. (2024) on ImageNet-1k with a DeiT-S backbone.

**Metrics.** We compare CANiT's accuracy to EE baselines along the accuracy-speedup Pareto frontier and to pruning baselines at matched FLOPs. We report Top-1 classification accuracy on the validation set and measure the FLOP speedup ratio similar to prior work (Dong et al., 2026),

[2]Unlike Dong et al. (2026), we report ViT-EE results with a frozen backbone.

[3]DeiT-B and DeiT3-B are architecturally similar but have different training recipes on ImageNet-1k. However, they use the same fine-tuning recipe for other datasets; we adopted this fine-tuning recipe for our work.

Table 3: FLOP speedup vs Throughput speedup on ImageNet-1k with DeiT-B backbone.

| Metric | Backbone | CANiT | | | | |
|---|---|---|---|---|---|---|
| Accuracy | 81.80% | 81.85% | 81.69% | 81.37% | 80.57% | 79.32% |
| FLOP Speedup | 1.00x | 1.24x | 1.50x | 1.75x | 2.00x | 2.26x |
| Throughput Speedup | 1.00x | 1.09x | 1.29x | 1.49x | 1.68x | 1.86x |
| Throughput / FLOP | 1.00 | 0.88 | 0.86 | 0.85 | 0.84 | 0.82 |

Table 4: Patch pruning comparison. * denotes accuracy reported by Zero-TPrune (ZTP, Wang et al. (2024)). Highest accuracy at matched FLOPs and highest speedup at matched accuracy are in bold.

| | FLOPs @ matched Acc | | | | Acc @ matched FLOPs | | | | | Acc @ matched FLOPs | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | DeiT-S | ZTP* | CANiT | CANiT-P | ATS* | ToMe* | ZTP* | CANiT | CANiT-P | ZTP* | CANiT | CANiT-P |
| GFLOPs | 4.60 | 3.97 | **3.39** | 3.78 | 3.00 | 2.95 | 2.97 | 2.95 | 2.98 | 2.50 | 2.51 | 2.50 |
| Acc | 79.8 | 79.8 | 79.8 | 79.8 | 79.2 | 78.9 | 79.4 | **79.7** | 79.6 | 79.1 | **79.2** | 78.8 |

$$\text{FLOP Speedup} = \frac{N \times C_M}{\sum_{i=1}^{M} n_i \times C_i} \tag{11}$$

where $N, n_i, C_i$ are the total image count, the count of images exiting at the $i$-th exit, and the FLOPs up to the $i$-th exit (measured using PyTorch's FlopCounterMode (Paszke et al., 2019)), respectively.

We also evaluate CANiT's batched throughput speedup on a single NVIDIA A40 GPU against its FLOP speedup. Appendix B.3 outlines our batched EE inference setup, which uses PyTorch's native `compile` and `scaled_dot_product_attention` to maximize throughput. Our experimental setup achieves ≈4170 and ≈1700 img/s on DeiT-S and DeiT-B baselines without any speedup. We do not compare to prior work's author-reported throughput as their profiling setups (e.g., batch size, compilation, measurement tools, etc.) are unspecified.

### 4.1 Comparison to EE baselines

Table 2 provides a comprehensive comparison across all datasets. Figure 3 depicts accuracy-speedup Pareto frontiers. CANiT and CANiT-P achieve state-of-the-art accuracy and speedups due to their dominant Pareto frontiers. On ImageNet-1k, CANiT's accuracy outperforms EnViT (prior EE state-of-the-art) by 0.88–3.35 percentage points across a range of matched FLOP speedups. Similar or larger (as high as 3.91% over EnViT) improvements are visible on other datasets. CANiT-P remains close to CANiT (which has no sharing) and outperforms EnViT, demonstrating the frontier is robust to parameter sharing: CANiT-P retains the accuracy advantages of Transformer exits at a quarter of CANiT's parameter overhead. Note that for ImageNet-1k and Food-101, our backbones are slightly more accurate than prior work's. CANiT's accuracies exceed this difference. Finally, on CIFAR-100 and Food-101, CANiT's 5th early exit achieves higher accuracy than the backbone's original accuracy; as such, CANiT *improves the backbone accuracy at higher speed than the backbone*.

However, algorithmic FLOP gains do not always translate directly into equal throughput gains. Table 3 compares CANiT's FLOP speedup to its throughput speedup under batched inference to confirm that FLOP speedups are a meaningful indicator of performance. We find that throughput speedup is 82–88% of FLOP speedup across all exits. The ratio is relatively stable, meaning that our FLOP-based comparisons closely track actual performance gains. FLOP and throughput speedups do not perfectly match due to two sources: 1) CPU-side bookkeeping for pruning and early exiting and 2) reduced GPU utilization for later exits where early exits have shrunk the effective batch. The latter is more evident at higher FLOP speedups where the batch size shrinks more aggressively.

### 4.2 Comparison to Patch Pruning Baselines

Table 4 provides a comparison to patch pruning baselines that are not fine-tuned for pruning. CANiT achieves the highest accuracy at matched FLOP speedup and the highest speedup at matched accu-

Table 5: [CLS]-only exiting ablation on CIFAR-100 with a DeiT3-B backbone. Accuracy and standard deviation are measured over 5 runs. Statistically significant results are in **bold**. GFLOPs overhead shows absolute and relative (to the backbone) increase in model FLOPs. ViT-EE+SD denotes ViT-EE with soft distillation. CANiT is trained similarly to ViT-EE+SD but with [CLS]-token only exiting. Statistically, the difference in exit performance between the two is insignificant and due to different experimental seeds.

| Method | GFLOPs Overhead | Average $i$-th Exit Accuracy (%) ± Standard Deviation of five Runs | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 |
| ViT-EE | $7.3_{(+41.4\%)}$ | 67.26±0.26 | 77.22±0.24 | 85.72±0.31 | **90.07±0.18** | 91.81±0.04 | **91.6±0** |
| ViT-EE+SD | $7.3_{(+41.4\%)}$ | **68.42±0.21** | **78.89±0.19** | **86.41±0.19** | **90.30±0.14** | **91.88±0.04** | **91.6±0** |
| CANiT ($\kappa = 0$) | $1.2_{(+6.4\%)}$ | **68.41±0.22** | **78.79±0.32** | **86.48±0.20** | **90.28±0.09** | **91.92±0.07** | **91.6±0** |

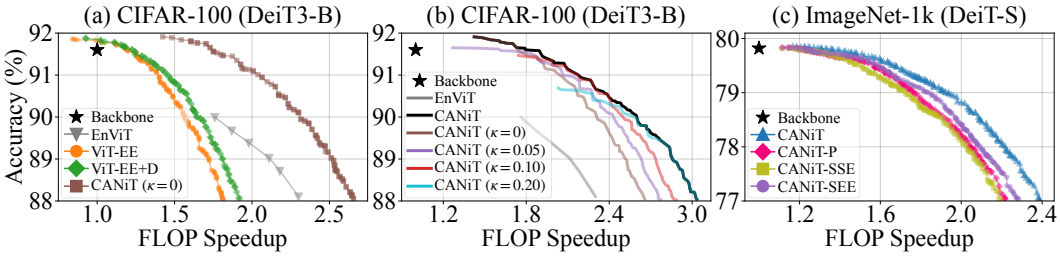



Figure 4: Pareto frontiers for **(a)** exit architecture, **(b)** patch pruning[4], and **(c)** weight sharing.

racy, while CANiT-P is competitive against prior baselines. These results highlight the complementary benefits of early exiting and patch pruning for anytime image classification. Importantly, these benefits are gained without incorporating more advanced pruning techniques (e.g. redundant token reduction (Bolya et al., 2023; Wang et al., 2024)); an approach that combines importance and redundancy criteria could lead to additional improvements.

### 4.3 Ablations

We isolate the contributions of our methods to understand their impact on the anytime Pareto frontier.

**Soft Distillation.** Although soft distillation is not a core contribution of our work, we nevertheless evaluate its impact and find that it improves exit accuracy on all datasets except ImageNet-1k (e.g., ViT-EE vs ViT-EE+SD in Table 5; full results in Appendix A). ImageNet-1k models underperform by up to 2.5% at shallow exits with soft distillation. We therefore train all ImageNet-1k models only on the ground-truth loss (i.e., $\lambda = 1$).

**Exit Architecture.** To evaluate the impact of [CLS]-token only exiting on model accuracy, Table 5 shows the results of training models with the [CLS]-token only architecture (CANiT) and without (ViT-EE+SD) on CIFAR-100 over five runs. [CLS]-token only exiting does not statistically impact exit performance—minor differences are due to random experimental seeds—but significantly reduces per-exit FLOP overheads. As a result, CANiT exhibits superior Pareto frontiers (Figure 4a) even without patch pruning (i.e., ($\kappa = 0$)). The impacts of FLOP reduction are also evident in per-epoch training durations, which were reduced by $\approx 38\%$ on average.

**Pruning Signal.** We compare CANiT's exit-attention-based pruning signal to one based on the backbone's attention score and random pruning across a range of pruning rates in Table 6. No model has undergone any fine-tuning for token pruning and CANiT does not exit early. CANiT achieves the highest accuracies, with the gap widening at higher pruning rates. We attribute this to the exit's attention score being trained for immediate classification, while the backbone's attention must serve downstream layers. The backbone's pruning signal is cheaper in FLOPs, since it requires no exit heads, but that cost is recovered by early exiting in our end-to-end comparison.

[4]Pareto point markers are removed for better visibility.

Table 6: Impact of pruning signal on the backbone's original exit accuracy across a range of pruning rates on ImageNet-1k with DeiT-S. CANiT's pruning signal incurs lower accuracy loss than one based on the backbone's `[CLS]` attention score or random pruning. Highest accuracy is in **bold**.

| Pruning Signal | Top1-Accuracy at Pruning Rate ($\kappa$) | | | | | | |
|---|---|---|---|---|---|---|---|
| | 0.00 | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 |
| Random Token | **79.82** | 79.37 | 78.52 | 77.52 | 76.08 | 74.12 | 71.31 |
| Backbone's `[CLS]` Attention Score | **79.82** | 79.72 | 79.29 | 78.63 | 77.58 | 76.11 | 73.95 |
| CANiT's Exit Attention Score | **79.82** | **79.77** | **79.50** | **79.18** | **78.51** | **77.40** | **75.78** |

Table 7: Parameter ablation on ImageNet-1k on DeiT-S. Param. overhead shows absolute and relative (to the backbone) increase in parameters. E, S, and L respectively denote exclusive (not shared), shared, LoRA (Section 3.3) parameters for each layer type. CANiT-P achieves most of the accuracy of unshared exits at just 25% of the parameter overhead.

| Method | Layer Type | | | Param. | $i$-th Exit Accuracy over five runs | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | `MLPs` | `MHSA` | `LN` | Overhead | 1 | 2 | 3 | 4 | 5 | 6 |
| CANiT | E | E | E | 10.8M$_{(+49.0\%)}$ | 51.85 | 64.42 | 71.70 | 76.88 | 79.44 | 79.82 |
| CANiT-SEE | S | E | E | 4.5M$_{(+20.6\%)}$ | 50.67 | 63.15 | 70.28 | 75.84 | 78.89 | 79.82 |
| CANiT-P | S | L | E | 2.7M$_{(+12.1\%)}$ | 49.00 | 62.18 | 69.43 | 75.39 | 78.54 | 79.82 |
| CANiT-SSE | S | S | E | 2.2M$_{(+9.8\%)}$ | 47.02 | 60.88 | 68.46 | 74.78 | 78.35 | 79.82 |

**Patch Pruning as a Knob.** No single $\kappa$ value is Pareto dominant. Figure 4b depicts the Pareto frontier of CIFAR-100 models on DeiT-B with fixed $\kappa$ values (CANiT, $\kappa$ = X) and CANiT, which uses variable $\kappa$ values for a dominant Pareto frontier. Low $\kappa$ values have a higher peak accuracy but achieve lower FLOP speedups. Higher rates, on the other hand, degrade the maximum possible accuracy but accomplish higher FLOP speedups. CANiT achieves the best of both by treating pruning as a performance knob. When high accuracy is required, CANiT uses a low pruning rate. When high FLOP speedups are demanded, CANiT uses a high pruning rate.

**Weight Sharing & Per-Exit LoRA** We analyze weight sharing by progressively sharing more exit parameters on ImageNet-1k (Table 7; Pareto frontiers in Figure 4c). Sharing the exit `MLPs` (CANiT-SEE) reduces parameter overhead from 49.0% to 20.6% at a cost of at most 1.5 points at any exit and degrades the Pareto frontier. Sharing the `MHSA` is more costly as fully shared attention (CANiT-SSE) loses another 3.7 points at the first exit, while per-exit LoRA (CANiT-P) recovers 2 points of this loss for a 2.3% growth in parameter count. Exclusive LayerNorm parameters account only for 9.2K trainable parameters and do not meaningfully impact performance (Appendix A). These results support the design decisions in Section 3.3; specifically, that attention is more sensitive to weight sharing and requires per-exit adaptation, which LoRA provides at negligible cost.

## 5 Limitations, Future Work, and Conclusions

**Limitations and Future Work.** `[CLS]`-token only exiting and $\boldsymbol{A}_{\texttt{[CLS]}}$-based patch pruning rely on the original ViT architecture as described by Dosovitskiy et al. (2021). Thus, CANiT is not applicable to alternative architectures that do not use the `[CLS]` token. While CANiT-P reduces CANiT's parameters by $\approx 75\%$, it still requires more parameters than either approach. Thus, CANiT and CANiT-P are less ideal on memory constrained systems. CANiT's pruning signal is based on token importance. An alternative method is to prune redundant tokens (e.g., Bolya et al. (2023) and Wang et al. (2024)). Augmenting CANiT with an additional redundant-based token pruning signal could further improve end to end performance. Finally, CANiT's throughput could be improved with more efficient bookkeeping and GPU utilization techniques; multi-batch early-exit inference is an active area of research (Subramaniyan et al., 2025; She et al., 2024).

**Conclusion.** We have presented CANiT, a framework for anytime EE ViTs. Our empirical results show that our principled optimizations are effective for anytime ViTs, yielding stronger early exits and improved accuracy-compute Pareto frontiers.

### AI USE STATEMENT

In this work, we used generative AI tools for manuscript preparation to improve readability, clarity, grammar, and spelling. Additionally, we used generative AI tools for literature retrieval and discovery. Plotting scripts were prototyped by generative AI and further edited by the authors. We have not used generative AI tools for implementations of the framework and experiments. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### REPRODUCIBILITY STATEMENT

Algorithmic details are provided in our Section 3, experimental and implementation details are provided in Appendix B, reproducibility information is provided in Appendix C.

## REFERENCES

Divya Jyoti Bajpai and Manjesh Kumar Hanawal. Beem: Boosting performance of early exit dnns using multi-exit classifiers as experts. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 62520–62535, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/9c84feb75eae1ef6389f31b3ef050b6a-Paper-Conference.pdf.

Arian Bakhtiarnia, Qi Zhang, and Alexandros Iosifidis. Multi-Exit Vision Transformer for Dynamic Inference, October 2021.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JroZRaRw7Eu.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp. 446–461, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10599-4.

Mengzhao Chen, Mingbao Lin, Ke Li, Yunhang Shen, Yongjian Wu, Fei Chao, and Rongrong Ji. Cf-vit: a general coarse-to-fine method for vision transformer. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN 978-1-57735-880-0. doi: 10.1609/aaai.v37i6.25860. URL https://doi.org/10.1609/aaai.v37i6.25860.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, June 2009. doi: 10.1109/CVPR.2009.5206848.

Yonghao Dong, Qiang He, Penghong Rui, Zhenzhe Zheng, Zhao Li, Feifei Chen, Hai Jin, and Yun Yang. Envit: enhancing the performance of early-exit vision transformers via exit-aware structured dropout-enabled self-distillation. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press, 2026. ISBN 978-1-57735-906-7. doi: 10.1609/aaai.v40i25.39225. URL https://doi.org/10.1609/aaai.v40i25.39225.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021.

Mohsen Fayyaz, Soroush Abbasi Koohpayegani, Farnoush Rezaei Jafari, Sunando Sengupta, Hamid Reza Vaezi Joze, Eric Sommerlade, Hamed Pirsiavash, and Jürgen Gall. Adaptive Token Sampling for Efficient Vision Transformers. In Shai Avidan, Gabriel Brostow, Moustapha Cissé,

Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 396–414, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-20083-0. doi: 10.1007/978-3-031-20083-0_24.

Jianing He, Qi Zhang, Hongyun Zhang, Xuanjing Huang, Usman Naseem, and Duoqian Miao. Cosee: consistency-oriented signal-based early exiting via calibrated sample weighting mechanism. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i22.34576. URL https://doi.org/10.1609/aaai.v39i22.34576.

Xuanli He, Iman Keivanloo, Yi Xu, Xiang He, Belinda Zeng, Santosh Rajagopalan, and Trishul Chilimbi. Magic Pyramid: Accelerating Inference with Early Exiting and Token Pruning, October 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, October 2021.

Hanzhang Hu, Debadeepta Dey, Martial Hebert, and J. Andrew Bagnell. Learning anytime predictions in neural networks via adaptive loss balancing. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19, pp. 3812–3821, Honolulu, Hawaii, USA, January 2019. AAAI Press. ISBN 978-1-57735-809-1. doi: 10.1609/aaai.v33i01.33013812.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q. Weinberger. Multi-Scale Dense Networks for Resource Efficient Image Classification, June 2018.

Metod Jazbec, Alexander Timans, Tin Hadži Veljković, Kaspar Sakmann, Dan Zhang, Christian A. Naesseth, and Eric Nalisnick. Fast yet safe: Early-exiting with risk control. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 129825–129854. Curran Associates, Inc., 2024. doi: 10.52202/079017-4124.

Yiğitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *International Conference on Machine Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:147703951.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *2013 IEEE International Conference on Computer Vision Workshops*, pp. 554–561, 2013. doi: 10.1109/ICCVW.2013.77.

A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not All Patches are What You Need: Expediting Vision Transformers via Token Reorganizations, April 2022.

Dongyang Liu, Meina Kan, Shiguang Shan, and Xilin Chen. A simple romance between multi-exit vision transformer and token reduction. In *International Conference on Learning Representations*, 2024.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. FastBERT: A Self-distilling BERT with Adaptive Inference Time, April 2020.

Maria-Elena Nilsback and Andrew Zisserman. Automated Flower Classification over a Large Number of Classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pp. 722–729, December 2008. doi: 10.1109/ICVGIP.2008.47.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, 2019.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient Vision Transformers with Dynamic Token Sparsification, October 2021.

Yechao She, Tuo Shi, Jianping Wang, and Bin Liu. Dynamic Batching and Early-Exiting for Accurate and Timely Edge Inference. In *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*, pp. 1–6, June 2024. doi: 10.1109/VTC2024-Spring62846.2024.10682995.

Srinivasan Subramaniyan, Rudra Joshi, Xiaorui Wang, and Marco Brocanelli. SEEB-GPU: Early-Exit Aware Scheduling and Batching for Edge GPU Inference. In *Proceedings of the Tenth ACM/IEEE Symposium on Edge Computing*, SEC '25, pp. 1–16, New York, NY, USA, December 2025. Association for Computing Machinery. ISBN 979-8-4007-2238-7. doi: 10.1145/3769102.3772715.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. BranchyNet: Fast Inference via Early Exiting from Deep Neural Networks, September 2017.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357. PMLR, 18–24 Jul 2021a. URL https://proceedings.mlr.press/v139/touvron21a.html.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 32–42, October 2021b.

Hugo Touvron, Matthieu Cord, and Hervé Jégou. DeiT III: Revenge of the ViT. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*, pp. 516–533, Berlin, Heidelberg, October 2022. Springer-Verlag. ISBN 978-3-031-20052-6. doi: 10.1007/978-3-031-20053-3_30.

Hongjie Wang, Bhishma Dedhia, and Niraj K. Jha. Zero-tprune: Zero-shot token pruning through leveraging of the attention graph in pre-trained transformers. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16070–16079, 2024. doi: 10.1109/CVPR52733.2024.01521.

Yulin Wang, Rui Huang, Shiji Song, Zeyi Huang, and Gao Huang. Not All Images are Worth 16x16 Words: Dynamic Transformers for Efficient Image Recognition. In *Advances in Neural Information Processing Systems*, volume 34, pp. 11960–11973. Curran Associates, Inc., 2021.

Guanyu Xu, Jiawei Hao, Li Shen, Han Hu, Yong Luo, Hui Lin, and Jialie Shen. LGViT: Dynamic Early Exiting for Accelerating Vision Transformer. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, pp. 9103–9114, New York, NY, USA, October 2023. Association for Computing Machinery. ISBN 979-8-4007-0108-5. doi: 10.1145/3581783.3611762.

Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution Adaptive Networks for Efficient Inference, May 2020.

Hongxu Yin, Arash Vahdat, Jose M. Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-ViT: Adaptive Tokens for Efficient Vision Transformer. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10799–10808, June 2022. doi: 10.1109/CVPR52688.2022.01054.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3712–3721, 2019. doi: 10.1109/ICCV.2019.00381.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. BERT Loses Patience: Fast and Robust Inference with Early Exit, October 2020.

Shlomo Zilberstein. Using Anytime Algorithms in Intelligent Systems. *AI Mag.*, 17(3):73–83, September 1996. ISSN 0738-4602. doi: 10.1609/aimag.v17i3.1232.

Table 8: Impact of soft distillation on exit accuracy. ImageNet-1k results were from 1 run. All other datasets were averaged across 5 runs. Bold values indicated statistically significant improvement.

| Dataset | $\lambda$ | $i$-th Exit Accuracy (%) 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| ImageNet-1k (DeiT-S) | 0.5 | 49.31±0.16 | 62.40±0.07 | 70.25±0.09 | 76.36±0.08 | 79.42±0.03 | **79.82±0** |
| | 1.0 | **51.85±0.25** | **64.42±0.19** | **71.70±0.11** | **76.88±0.07** | **79.44±0.05** | **79.82±0** |
| CIFAR-100 (DeiT3-B) | 0.5 | **68.41±0.22** | **78.79±0.32** | **86.48±0.20** | **90.28±0.09** | **91.92±0.07** | **91.60±0** |
| | 1.0 | 67.26±0.26 | 77.22±0.24 | 85.72±0.31 | **90.07±0.18** | 91.81±0.04 | **91.60±0** |
| Foods-101 (DeiT3-B) | 0.5 | **73.52±0.15** | **84.32±0.10** | **90.14±0.06** | **92.11±0.08** | **92.19±0.04** | **91.31±0** |
| | 1.0 | 71.36±0.21 | 82.46±0.15 | 89.35±0.11 | 91.77±0.05 | 91.94±0.07 | 91.31±0 |
| Flowers-102 (DeiT3-B) | 0.5 | **78.57±0.47** | **88.78±0.37** | **94.96±0.2** | **95.92±0.11** | **96.33±0.08** | **96.65±0** |
| | 1.0 | 76.09±0.49 | 87.41±0.47 | 94.45±0.16 | **95.73±0.16** | 96.05±0.16 | **96.65±0** |

Table 9: Parameter ablation on ImageNet-1k on DeiT-S. E, S, and L respectively denote exclusive (not shared), shared, LoRA (Section 3.3) parameters for each layer type. CANiT-P achieves most of the accuracy of unshared exits at just 25% of the parameter overhead.

| Method | Layer Type MLPs | MHSA | LN | $i$-th Exit Accuracy over five runs 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|---|
| CANiT | E | E | E | 51.85±0.25 | 64.42±0.19 | 71.70±0.11 | 76.88±0.07 | 79.44±0.05 | 79.82±0 |
| CANiT-SEE | S | E | E | 50.67±0.10 | 63.15±0.13 | 70.28±0.08 | 75.84±0.04 | 78.89±0.07 | 79.82±0 |
| CANiT-P | S | L | E | 49.00±0.21 | 62.18±0.09 | 69.43±0.07 | 75.39±0.07 | 78.54±0.09 | 79.82±0 |
| CANiT-SSE | S | S | E | 47.02±0.16 | 60.88±0.04 | 68.46±0.14 | 74.78±0.1 | 78.35±0.03 | 79.82±0 |
| CANiT-SSS | S | S | S | 47.04±0.14 | 60.66±0.07 | 68.24±0.18 | 74.6±0.06 | 78.31±0.09 | 79.82±0 |

# A Additional Results

**Impact of Soft Distillation** In general, we find that soft distillation is beneficial for all datasets except ImageNet-1k. Therefore, we train ImageNet-1k with $\lambda = 1.0$ and other datasets with $\lambda = 0.5$. Table 8 shows the impact of this choice for each dataset. ImageNet-1k is negatively impacted soft distillation; all other models benefit.

**Parameter Efficiency Accuracies with Standard Deviation** We include standard deviation for parameter ablation results in Table 9.

# B Experimental and Implementation Details

## B.1 FLOP Measurements

FLOPs were measured with PyTorch's native FlopCounterMode (Paszke et al., 2019), which counts each multiplication and addition as separate floating-point operations. Prior work (e.g., Dong et al. (2026); Wang et al. (2024); Liu et al. (2024)) reports multiply-accumulate operations (MACs) labeled as FLOPs; hence, we halve values reported by FlopCounterMode to match prior work.

## B.2 Pareto Search

We obtain Pareto frontiers by sweeping over $T_1 \dots T_E \in \{0.0, 0.1, 0.2, \dots, 0.9, 1.0\}$ & $\kappa \in \{0, 0.03, 0.05, 0.075, 0.1, 0.125, 0.15, 0.175, 0.2\}$ values and selecting the most accurate configuration for at matched FLOPs. We optimize this search process by recording all exit predictions for a single $\kappa$ and then measuring the inference accuracy by varying confidence thresholds. In hardware setup, finding the Pareto frontier for an ImageNet-1k model takes approximately 25-30 minutes on a single GPU.

## B.3 Throughput with Batched Inference

**Measurement Setup** We measure CANiT's throughput on a single NVIDIA A40 GPU. We compile the model with PyTorch's native `torch.compile` and toggle dynamic shape tracing to pre-

Table 10: Datasets used in our experiments.

| Dataset | Train Size | Validation Size | #classes |
|---|---|---|---|
| ImageNet-1k Deng et al. (2009) | 1,281,167 | 50,000 | 1,000 |
| CIFAR-100 Krizhevsky (2009) | 50,000 | 10,000 | 100 |
| Food-101 Bossard et al. (2014) | 75,750 | 25,250 | 101 |
| Flowers-102 Nilsback & Zisserman (2008) | 2,040 | 6,149 | 102 |

vent recompilation during inference. In addition, we mark input samples' batch dimension as dynamic as an additional hint to the compiler.

Models were ran with "half precision AMP" managed by Lightning Fabric. After a 5 warm up batches, we measure the run time and CUDA time of 100 batches with batch size 256 using PyTorch's native profiler. We measure throughput by dividing the number of processed samples (25,600) by the run time.

**Early Exiting** Our FLOP speedups are averaged over the entire validation set. To measure average throughput behavior for a single configuration, we ensure that each batch has a similar exiting ratio as the entire dataset.

**Book keeping** After each exit, we only keep samples that did not meet the confidence threshold. Label predictions for samples that did meet the threshold are extracted and kept until batch processing is complete.

We reduce the batch size via simple indexing of PyTorch tensors. While naive data dependent indexing leads to CPU-GPU synchronization and graph breaks (which manifest as run time overheads), PyTorch's modern compiler (version 2.x+) handles this process without causing a graph break. Thus, `torch.compile` is a necessary component of our throughput measurement on batched inference.

## C Reproducibility Details

**Hardware & Software** All experiments were conducted on an AMD EPYC 7313P 16-Core Processor with 4× NVIDIA A40 GPUs (48GB VRAM memory each). Backbones for non-ImageNet-1k datasets were trained in parallel across 3 GPUs. ImageNet-1k EE models were trained in parallel across 4 GPUs. Other EE models were trained on a single GPU each. We used PyTorch version 2.14.0 along with CUDA 13.3 in our experiments. We use Lightning Fabric with version 2.6.5 for multi-GPU training. We build on the original DeiT (Touvron et al., 2021a) repository for our work.

**Datasets** We conduct our experiments on 4 different datasets (Table 10). Models were trained on the training set and evaluated on the validation set. We do not compare our work on the Stanford Cars dataset (Krause et al., 2013) as all of our fine-tuned models exceed the backbone accuracy reported by Dong et al. (2026).

**Fine-tuning for non-ImageNet-1k Datasets** We obtained backbones for CIFAR-100, Food-101, and Flowers-102 by fine-tuning pretrained DeiT3-B models originally trained on ImageNet-1k. We used the fine-tuning recipe reported by Touvron et al. (2021a; 2022); respective hyperparameters are listed in Table 11.

**EE Training** We trained EE models with hyperparameters outlined in Table 12 and evaluated on the checkpoint saved at the end of training. Backbone parameters were excluded from optimization.

**LoRA Rank** For CANiT-P, we use LoRA rank $r = 32$ in our experiments. Alternative values could be used to tradeoff parameter overheads for accuracy.

Table 11: Pretraining hyperparameters.

| Hyperparameter | Value |
|---|---|
| Epochs | 1000 |
| Global Batch size | 768 |
| Optimizer | AdamW |
| LR | $7.5 \times 10^{-5}$ |
| Weight Decay | 0.05 |
| LR scheduler | cosine |
| Warmup Epochs | 5 |
| Warmup LR | $10^{-6}$ |
| Auto Augmentation | rand-m9-mstd0.5-inc1 |
| Color Jitter | 0.0 |
| Repeated Augmentation | True |
| Label Smoothing | 0.1 |
| Random Erase Prob. | 0.0 |
| Mixup alpha | 0.8 |
| CutMix alpha | 1.0 |
| drop-path Rate | 0.00 |
| Crop ratio for Eval. | 1.0 |

Table 12: EE training hyperparameters for ImageNet and other datasets.

| Hyperparameter | ImageNet-1k (DeiT-S/B) | Other Datasets (DeiT3-B) |
|---|---|---|
| Epochs | 100 | 200 |
| Global Batch size | 1024 | 256 |
| Optimizer | AdamW | |
| LR for others | $10^{-3}$ | |
| Weight Decay | 0.05 | |
| LR scheduler | cosine | |
| Warmup Epochs | 5 | |
| Warmup LR | $10^{-6}$ | |
| Auto Augmentation | rand-m9-mstd0.5-inc1 | |
| Color Jitter | 0 | |
| Repeated Augmentation | False | |
| Label Smoothing | 0.1 | |
| Random Erase Prob. | 0.1 | |
| Mixup alpha | 0.8 | |
| CutMix alpha | 1.0 | |
| drop-path Rate | 0 | |
| Crop ratio for Eval. | 0.875 | 1.0 |
| $\lambda$ | 1 | 0.5 |
| $\tau$ | 2.0 | |