LGViT: Dynamic Early Exiting for Accelerating Vision Transformer

Guanyu Xu*
Beijing Institute of Technology
Beijing, China
xuguanyu@bit.edu.cn

Jiawei Hao*
Beijing Institute of Technology
Beijing, China
haojiawei7@bit.edu.cn

Li Shen
JD Explore Academy
Beijing, China
mathshenli@gmail.com

Han Hu†
Beijing Institute of Technology
Beijing, China
hhu@bit.edu.cn

Yong Luo
Wuhan University
Wuhan, China
luoyong@whu.edu.cn

Hui Lin
China Academic of Electronics and Information Technology
Beijing, China
linhui@whu.edu.cn

Jialie Shen
City, University of London
London, U.K.
jialie@gmail.com

ABSTRACT

Recently, the efficient deployment and acceleration of powerful vision transformers (ViTs) on resource-limited edge devices for providing multimedia services have become attractive tasks. Although early exiting is a feasible solution for accelerating inference, most works focus on convolutional neural networks (CNNs) and transformer models in natural language processing (NLP). Moreover, the direct application of early exiting methods to ViTs may result in substantial performance degradation. To tackle this challenge, we systematically investigate the efficacy of early exiting in ViTs and point out that the insufficient feature representations in shallow internal classifiers and the limited ability to capture target semantic information in deep internal classifiers restrict the performance of these methods. We then propose an early exiting framework for general ViTs termed **LGViT**, which incorporates heterogeneous exiting heads, namely, local perception head and global aggregation head, to achieve an efficiency-accuracy trade-off. In particular, we develop a novel two-stage training scheme, including end-to-end training and self-distillation with the backbone frozen to generate early exiting ViTs, which facilitates the fusion of global and local information extracted by the two types of heads. We conduct extensive experiments using three popular ViT backbones on three vision datasets. Results demonstrate that our LGViT can achieve competitive performance with approximately 1.8 × speed-up.

*Both authors contributed equally to this research.
†Han Hu is the corresponding author.

MM '23, October 29–November 3, 2023, Ottawa, ON, Canada.

ACM ISBN 979-8-4007-0108-5/23/10…$15.00
https://doi.org/10.1145/3581783.3611762

CCS CONCEPTS

• **Computing methodologies** → **Computer vision tasks**.

KEYWORDS

Vision transformer, early exit, heterogeneous exiting heads, self-distillation

ACM Reference Format:
Guanyu Xu, Jiawei Hao, Li Shen, Han Hu, Yong Luo, Hui Lin, and Jialie Shen. 2023. LGViT: Dynamic Early Exiting for Accelerating Vision Transformer. In *Proceedings of the 31st ACM International Conference on Multimedia (MM '23), October 29–November 3, 2023, Ottawa, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3581783.3611762

1 INTRODUCTION

During the past few years, vision transformers (ViTs) have become fundamental backbones for various multimedia tasks due to their powerful performance and universal structures [7, 30, 33]. With the development of 5G wireless networks and the artificial intelligence of things (AIoT), deploying ViTs on resource-constrained edge devices to enable real-time multimedia applications has become an appealing prospect. However, the high computational complexity of ViTs poses a significant challenge to deploy them on edge devices. For example, ViT-L/16 [26], a typical ViT architecture for computer vision, requires over 180 giga FLOPs for inference and takes 56.79 milliseconds on an NVIDIA Jetson TX2 device to classify an image with 224 × 224 resolution. Given that performance and quality-of-service (QoS) are critical for real-time multimedia systems, deploying such latency-greedy ViTs on resource-constrained edge devices is a challenging task.

Early exiting provides a feasible solution for accelerating the inference of neural networks by terminating forward propagation once the prediction from internal classifiers satisfies a certain criterion. While early exiting has been extensively studied for CNNs and transformer models in NLP, its application to ViTs remains an open problem. The main challenges in developing an efficient

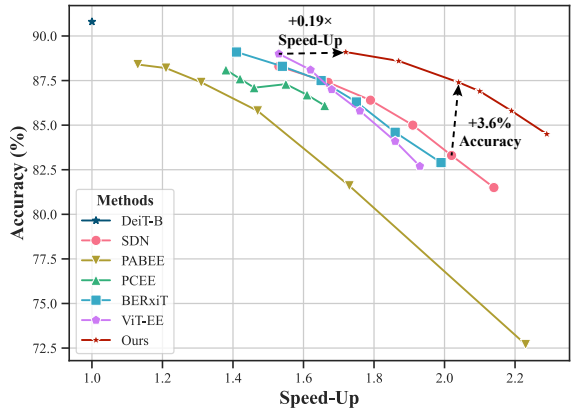

Figure 1: The comparison of performance and efficiency trade-off for the ViT backbone in CIFAR-100. LGViT significantly outperforms other early exiting methods. In particular, LGViT achieves new state-of-the-art 89.1 % accuracy but faster than ViT-EE [1]. Details are in Table 1 and Section 4.2.

early exiting framework for ViTs can be condensed into three key aspects. Firstly, directly applying early exiting strategies on ViTs leads to substantial performance degradation. However, there has been no systematic investigation into what limits the performance. Secondly, minimizing the accuracy drop and further accelerating the inference of early exiting ViTs on edge devices is challenging. Lastly, in the training phase, internal classifiers lose considerable information from the final classifier, resulting in poor performance.

Regarding the first challenge, Kaya *et al.* [9] discovered CNNs can reach correct predictions before their final layer, and they introduced internal classifiers to mitigate the overthinking problem. Sajjad *et al.* [20] examined the impact of dropping layers in transformer models and found that lower layers are more critical for task performance. However, their analyses were limited to CNNs or transformer models and did not consider the constraints of early exiting methods in ViTs. Concerning the second challenge, a series of studies have introduced exiting criteria to determine when to terminate forward propagation [22, 36] or designed advanced backbone networks to balance performance and efficiency [25, 28]. Although Bakhtiarnia *et al.* [1] proposed an early exiting framework for ViT by incorporating additional backbone blocks as exiting heads, a considerable speed-up gap remains between these methods and the constraints imposed by mobile and edge platforms. It is advantageous to design efficient exiting heads for constructing early exiting ViTs with rich feature representations. In relation to the third challenge, distillation-based [14, 35] approaches provide a promising solution to help internal classifiers imitate the final classifiers. However, these methods are only available to the same early exiting head architectures.

To remedy these limitations, we initially conduct probing experiments to examine the direct application of early exiting methods in ViTs. We discover that the performance of early exiting is constrained by: *i)* inadequate feature representations in shallow internal classifiers; *ii)* the weak ability to capture target semantic information in deep internal classifiers. Building on these insights, we then propose an efficient early exiting framework for general ViTs, termed LGViT, which accelerates inference while maintaining almost the same accuracy. In LGViT, we incorporate heterogeneous exiting heads, specifically, the local perception head and global aggregation head, to generate early exiting ViT networks. The local perception head is attached to shallow exiting points to capture local information and learn sufficient feature representations. Conversely, the global aggregation head is connected to deep exiting points to extract global information, thereby enhancing the capture of target semantic feature. To the best of our knowledge, this is the first work to employ heterogeneous exiting heads for early exiting ViTs. Subsequently, we propose a novel two-stage training strategy for early exiting ViTs. During the first stage, we utilize an end-to-end method to help the backbone ViT achieve its full potential. In the second stage, we froze the parameters of backbone and solely update the exiting heads. Self-distillation between exiting heads is employed to minimize information loss. Lastly, we perform extensive experiments to validate the superiority of our proposed framework for accelerating ViT inference, achieving a good efficiency-accuracy trade-off for three ViT backbones on three datasets. For example, as shown in Figure 1, when ViT serves as the backbone, our method can accelerate the inference by 1.72 × with only 1.7 % accuracy drop on the CIFAR-100 dataset.

Our main contributions are summarized as follows:

- We conduct a systematic investigation into the effectiveness of early exiting in ViTs and analyze the issues arising from the vanilla early exiting.
- We propose an efficient early exiting framework termed LGViT for general ViTs, incorporating heterogeneous exiting heads, *i.e.*, local perception head and global aggregation head, to achieve an efficiency-accuracy trade-off.
- We develop a novel two-stage training strategy that facilitates learning among multiple heterogeneous exiting heads and significantly minimizes information loss.
- We perform extensive experiments on three widely-used datasets and representative ViT backbones, demonstrating the superiority of our proposed framework, which achieves an average speed-up of 1.8 × with only 2% accuracy sacrifice.

2 RELATED WORKS

Efficient ViT. Due to their considerable computational cost, ViTs are challenging to deploy on resource-constrained edge devices for real-time inference [21, 24]. Recently several studies have proposed lightweight architectures to enhance performance. For example, Mehta *et al.* [17] incorporate convolution into transformers, combining the strengths of convolution and attention. Maaz *et al.* [16] propose an efficient hybrid architecture and design split depth-wise channel groups encoder to increase the receptive field. Furthermore, a series of methods employ traditional model compression techniques to obtain compact ViTs, such as network pruning [13, 23, 37], knowledge distillation [8, 26] and low-bit quantization [6, 34]. Hao *et al.* [8] utilize patch-level information to help compact student models imitate teacher models. Kwon *et al.* [13] propose a post-training pruning framework with structured sparsity methods.

Early exiting strategy. Early exiting is an effective dynamic inference paradigm that allows confident enough predictions from internal classifiers to exit early. Recent research on early exiting can be broadly categorized into two classes: *1) Architecture design.*

Some studies focus on designing advanced backbone networks to balance performance and efficiency. For example, Teerapittayanon *et al.* [25] first propose to attach internal classifiers at varying depth in DNNs to accelerate inference. Wołczyk *et al.* [28] introduce cascade connections to enhance information flow between internal classifiers and aggregate predictions from multiple internal classifiers to improve performance. These methods scarcely consider the design of the exiting head architecture and nearly all utilize a fully connected layer following a pooler as the exiting head. Bakhtiarnia *et al.* [1] propose to insert additional backbone blocks as early exiting branches into ViT. *2) Training scheme.* Another line of work designs training schemes to enhance the performance of internal classifiers. Liu *et al.* [14] employ self-distillation to help internal classifiers learn the knowledge from the final classifier. Xin *et al.* [32] introduce an alternating training scheme to alternate between two objectives for odd-numbered and even-numbered iterations.

Existing methods for efficient ViT primarily focus on elaborately designing compact ViT structures or applying model compression techniques to compress ViT. Our approach adopts sample-level acceleration for inference by dynamically adapting outputs at different paths based on the confidence of each exit's prediction. Regarding early exiting strategies, the work most related to this paper is CNN-Add-EE and ViT-EE in [1], which use a convolution layer and a transformer encoder as exiting heads, respectively. However, their performance is unsatisfactory and cannot achieve efficient inference. To the best of our knowledge, we first introduce heterogeneous exiting heads to construct early exiting ViTs and achieve an efficiency-accuracy trade-off. On top of the aforementioned studies, We also propose a novel two-stage training scheme to bride the gap between heterogeneous architectures.

3 METHOD

In this section, we first provide the motivation and an overview of the proposed LGViT framework. Then we illustrate the heterogeneous exiting heads and two-stage training strategy. Lastly, we depict the exit policy employed during the inference process.

3.1 Motivation

The early exiting method can halt the forward propagation of neural networks prematurely to provide an efficiency-accuracy trade-off, which has achieved significant performance improvements for CNNs and transformers in NLP. However, naively implementing early exiting on ViT may not yield performance gains for internal classifiers. For instance, the performance of the internal classifier on the fifth and tenth layers decreases by 21.8% and 4.0%, respectively, compared to the original classifier for ViT-B/16 [7] on CIFAR-100 [12]. As illustrated in Figure 2, we compare the attention maps at different exiting points for DeiT-B (the detailed description of probing experiments is presented in Appendix A.2). The deep classifiers can extract target semantic features to identify objects. Therefore, we obtain the following observations:

- ***Observation 1*:** Shallow internal classifiers cannot learn sufficient feature representation.
- ***Observation 2*:** Deep internal classifiers cannot capture target semantic information.

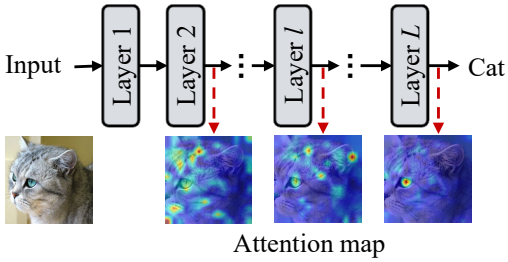

Figure 2: Comparison of attention maps at different exiting positions. The classifiers are omitted. The shallow internal classifiers are difficult to identify object due to inadequate feature capture. The deep internal classifiers do not capture target semantic information compared to the last classifiers.

We also discover that if both convolution and self-attention are employed as exiting architectures, positioned on shallow and deep layers respectively, the model would gain access to a more comprehensive combination of local and global information compared to the vanilla head architecture.

3.2 Overview

Motivated by the aforementioned observations, we propose an early exiting framework for ViTs that incorporates heterogeneous early exiting architectures. An overview of the proposed framework is depicted in Figure 3. It comprises a ViT backbone, multiple local perception heads and multiple global aggregation heads. Initially, a ViT backbone consisting of L encoder blocks is provided. We add M internal classifiers to intermediate blocks of ViT. Generally, M is smaller than the total number of backbone layers, as adding internal classifiers after every layer would result in substantial computation costs. The position of internal classifiers is independent of the backbone layer numbering.

We follow a three-step procedure to construct the early exiting ViT framework:

- **Attaching heterogeneous exiting heads:** Starting from a backbone of ViT, we first select several exiting points along its depth. Then we place local perception heads and global aggregation heads at corresponding exiting points according to their positions.
- **Two-Stage training:** We train the whole early exiting ViT using a novel two-stage training strategy including end-to-end training and self-distillation with the backbone frozen. This can facilitate the integration of global and local information for performance improvement.
- **Dynamic inference:** When the trained model is deployed for inference, each input sample can dynamically determine its output at varying depths based on the confidence of each exiting head prediction.

3.3 Attaching Heterogeneous Exiting Heads

We first introduce the placement of exiting heads, followed by a detailed description of the heterogeneous exiting heads. In this work, exiting points, where exiting heads are positioned, are determined according to an approximately equidistant computational distribution, *i.e.*, the multiply-accumulate operations (MACs) of

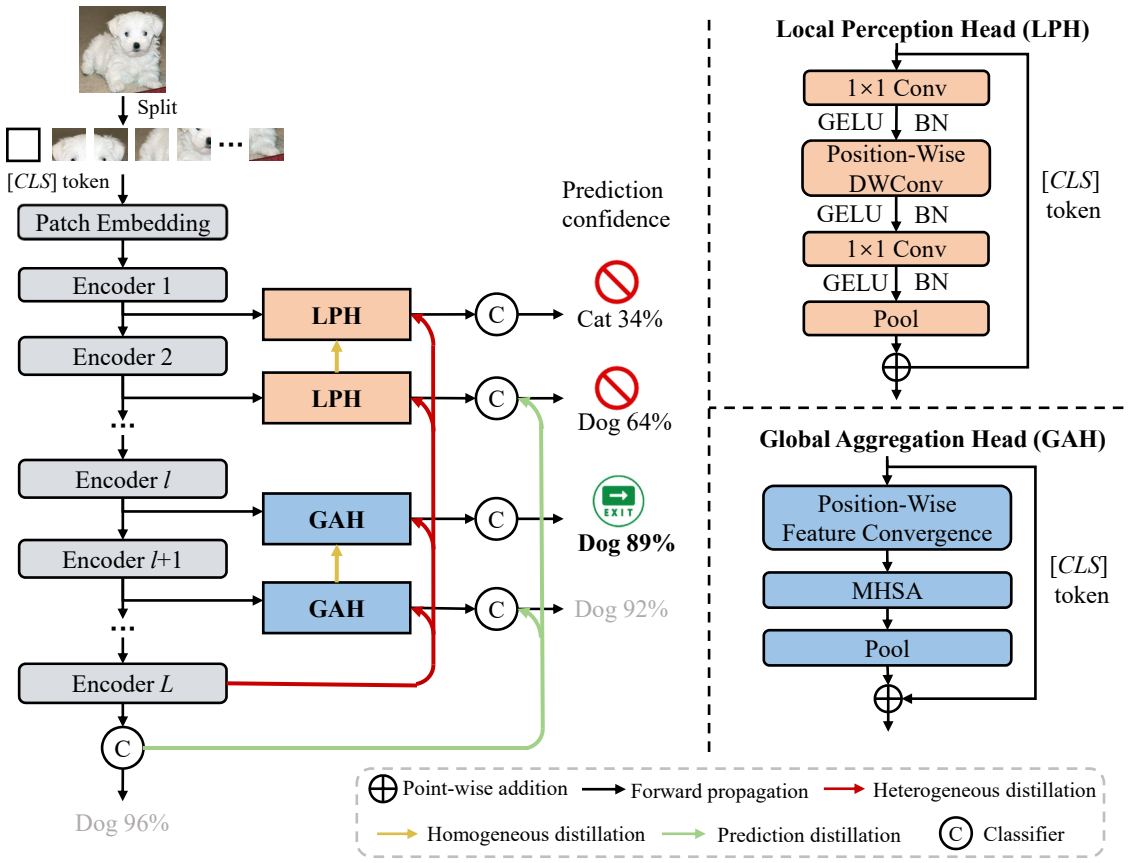

Figure 3: Overview of the proposed early-exiting ViT framework. 1) Given a backbone of ViT, we first attach local perception head (LPH) at lower half exiting points and global aggregation head (GAH) at top half of exiting points. 2) During the training phase, after an end-to-end training of the backbone, all exiting heads are jointly trained through a novel self-distillation utilizing heterogeneous features, homogeneous features and prediction logits as supervision with the backbone frozen. 3) In the inference stage, each input sample dynamically adjusts its exiting path according to the prediction confidence.

intermediate blocks between two adjacent points remain consistent. For the sake of simplicity, exiting points are constrained to be placed at the output of individual encoder blocks. We attach local perception heads, based on convolution, to the lower half of exiting points to enhance local information exploration. Global aggregation heads, based on self-attention, are integrated into the upper half of points to augment global information acquisition.

Local perception head. As analyzed in Section A.2, directly applying early exiting in ViT leads to severe performance degradation for shallow internal classifiers. To mitigate the issue, we introduce a local perception head (LPH) for early exiting framework, which elegantly incorporates convolution into ViT to enhance feature representation learning. It can achieve efficient local information exploration and effective feature integration extracted from the original backbone. As illustrated in the upper right of Figure 3, the proposed LPH first employs a 1×1 convolution layer to expand dimensions. Subsequently, the expanded features are passed to a position-wise depth-wise convolution (PDConv) with $k \times k$ kernel size that depends on the exiting positions m. In order to reduce computation overhead, we employ smaller kernel size convolutions for the deeper exiting points. We employ a decreasing linear mapping function $f(\cdot)$ to determine the kernel size of PDConv, *i.e.*, $k = f(m), m \leq L/2$. For instance, the expanded features at the m-th exiting position are passed to $k \times k$ depth-wise convolution. Note that $k = 0$ means that the expanded features will bypass PDConv and proceed directly to the subsequent part. Thus, the PDConv can be formulated as:

$$\text{PDConv}(\mathbf{X}, m) = \begin{cases} \text{DWConv}_{k\times k}(\mathbf{X}), & f(m) > 0 \\ \mathbf{X}, & f(m) = 0 \end{cases}, \tag{1}$$

where $\text{DWConv}_{k\times k}$ denotes the depth-wise convolution with kernel of size $k \times k$. Then the features are projected back into the original patch space using a 1×1 convolution and then passed to an average pooling layer. Considering that the $[CLS]$ token contains djominant feature representations, it is added to the pooled output to facilitate the fusion of global information from the original backbone and local information from the convolution. Concretely, given a ViT encoder output $\mathbf{X}_{en} \in \mathbb{R}^{N\times D}$, where N represents the number of patches and D denotes the hidden dimension, when the position of exiting points is lower than $L/2$, the output of the

proposed exiting head is given by:

$$\begin{aligned} \mathrm{LPH}(\mathbf{X}_{en}, m) &= \mathrm{Pool}(\mathrm{Conv}_{1\times1}(\mathcal{G}(\mathbf{X}_{en}, m))) + \mathbf{X}_{CLS}, \\ \mathcal{G}(\mathbf{X}_{en}, m) &= \mathrm{PDConv}(\mathrm{Conv}_{1\times1}(\mathbf{X}_{en}), m), \end{aligned} \tag{2}$$

where $\mathbf{X}_{CLS}$ represents $[CLS]$ token and the activation layer is omitted. Gaussian error linear unit (GELU) and batch normalization (BN) are employed after each convolution. The output of LPH is finally passed to the internal classifier. By introducing LPH as the exiting head, shallow internal classifiers can learn adequate feature representation and capture local information, thereby enhancing performance in vision tasks.

Global aggregation head. Based on the discussion in Section A.2, the direct application of early-exit methods to ViT hinders the semantic information capture in deep internal classifiers. We propose a global aggregation head (GAH) and incorporate it at deep exiting points, as illustrated in the lower right of Figure 3. The proposed GAH integrates features from locally adjacent tokens and then compute self-attention for each subset to facilitate target semantic information exploitation. In GAH, we first employ a position-wise feature convergence (PFC) block to aggregate features from the exiting point. In the PFC block, input features $\mathbf{X} \in \mathbb{R}^{N\times D}$ are reshaped to $D \times H \times W$ dimensions and down-sampled with a size of $s \times s$ window. The sampled features $\mathbf{X}_{sample} \in \mathbb{R}^{D\times \frac{H}{s}\times \frac{W}{s}}$ are restored to original dimension format $\frac{N}{s^2} \times D$. The proposed PFC block reshapes the input features to patch format and down-samples them with an $s \times s$ window. The sampled features are then restored to original format. To avoid introducing additional learnable parameters, we employ an average pool with s stride as the implementation of PFC. Analogous to PDConv, the swindow size s of PFC also depends on the exiting position m. Deeper exiting points utilize larger window sizes significantly reducing the computational cost. We employ an increasing linear mapping function $g(\cdot)$ to determine the window size of PFC, *i.e.*, $s = g(m), L/2 < m \leq L$. For example, the input features are passed to sub-sample with a size of $g(m) \times g(m)$ window at the m-th exiting point. Note that the minimum window size is generally set to 2. Consequently, the PFC can be expressed as:

$$\mathrm{PFC}(\mathbf{X}_{en}, m) = \mathrm{Pool}_{g(m)}(\mathbf{X}_{en}), \tag{3}$$

where $\mathrm{Pool}_{g(m)}$ represents an average pool with $g(m)$ stride. The reshaping and recover operations of features are omitted in the equation. PFC not only reduces the computational redundancy but also helps focus on target patches compared to the original MHSA. Then the integrated features are passed through multi-head self-attention (MHSA) and a pool layer. The $[CLS]$ token is also added to the pooled features. Thus, when the position of exiting points is deeper than $L/2$, the proposed GAH can be formulated as:

$$\begin{aligned} \mathrm{GAH}(\mathbf{X}_{en}, m) &= \mathrm{Pool}(\mathrm{MHSA}(\mathrm{PFC}(\mathbf{X}_{en}, m))) + \mathbf{X}_{CLS}, \\ \mathrm{MHSA}(\mathbf{X}) &= \mathrm{softmax}\left(\frac{XW_Q(XW_K)^T}{\sqrt{d}}\right)XW_V, \end{aligned} \tag{4}$$

where the input $\mathbf{X}$ is linearly transformed into query, key and value vectors using transformation matrices W_Q, W_K and W_V; d is the vector dimension. By employing GAH as the exiting head, deep internal classifiers can reduce spatial redundancy of self-attention and capture more target semantic information.

Complexity analysis. To thoroughly understand the computational bottleneck of heterogeneous exiting heads, we compare our proposed LPH + GAH with standard convolution + MHSA by analyzing their floating-point operations (MACs). Given an input feature of size $N \times D$, the FLOPs of standard $k \times k$ convolution are:

$$\mathcal{O}(\mathrm{Conv}_{k\times k}) = ND^2k^2. \tag{5}$$

The FLOPs of an MHSA module can be calculated as:

$$\mathcal{O}(\mathrm{MHSA}) = 2ND(D+D) + N^2(D+D) = 4ND^2 + 2N^2D, \tag{6}$$

where the activation function is omitted. For a fair comparison, we select the same kernel size k in LPH. The FLOPs of the proposed two exiting head are as follows:

$$\begin{aligned} \mathcal{O}(\mathrm{LPH}) &= 2ND^2 + NDk^2, \\ \mathcal{O}(\mathrm{GAH}) &= 4ND^2/s^2 + 2N^2D/s^4. \end{aligned} \tag{7}$$

We observe that the computational complexity of LPH and GAH is lower than that of standard convolution and MHSA, respectively.

$$\begin{aligned} \frac{\mathcal{O}(\mathrm{LPH})}{\mathcal{O}(\mathrm{Conv}_{k\times k})} &= (2D + k^2)/Dk^2 < 1, \\ \frac{\mathcal{O}(\mathrm{GAH})}{\mathcal{O}(\mathrm{MHSA})} &= \frac{2D + N/s^2}{2D + N} < 1. \end{aligned} \tag{8}$$

Therefore, compared to standard convolution and MHSA, our proposed LPH and GAH heads are more friendly to computational cost and convenient to implement on hardware platforms.

3.4 Two-Stage Training Strategy

To tackle the performance degradation issue caused by early exiting, we propose a novel two-stage training strategy to transfer knowledge from deeper classifiers to shallow classifiers. The process of two-stage training strategy is presented as follows.

- In the first stage, we train the backbone ViT and update the parameters of the backbone and the final classifier in an alternating strategy. As a result, the interference of multiple internal classifiers can be minimized, enabling the final classifier to reach its full potential. Similar to general training, we utilize the cross entropy function as the training loss.
- During the second stage, the backbone and final classifier are kept frozen. Only the parameters of exiting heads and internal classifiers can be updated. We introduce self-distillation to facilitate the imitation of all exiting heads from the last classifier as illustrated in the left of Figure 3.

The overall distillation loss comprises the heterogeneous distillation, homogeneous distillation and prediction distillation loss.

Heterogeneous distillation. Considering the usage of heterogeneous exiting head, the gap between different types of heads is substantial. Directly utilizing the features from the last layer as supervision for internal classifiers can lead to information loss. To tackle this issue, we propose heterogeneous distillation to facilitate learning the knowledge from heterogeneous architectures. To reduce the conflict between multiple losses, we only employ the feature of the last layer as the reference feature for the first and last exiting heads of LPH and GAH. The inductive bias and local visual representations captured by LPH can be elegantly integrated with global information extracted from self-attention. Considering the different shapes of feature maps between exiting heads and

the final block, we employ an aligning module to match the dimensions. The module consists of a depth-wise convolution, GELU and BN activation functions. The feature map of the last ViT layer $\mathbf{F}_L \in \mathbb{R}^{N\times D}$ is first reshaped to $D \times \sqrt{N} \times \sqrt{N}$ dimensions. The reshaped feature map is passed to the module to reduce dimensions, and then restored to original dimension format $N' \times D$. The loss function of heterogeneous can be formulated as:

$$\mathcal{L}_{hete} = \frac{1}{4} \sum_{m\in\mathcal{M}} \mathcal{L}_{KL}(\mathbf{F}_m, \text{Align}(\mathbf{F}_L)), \tag{9}$$

where $\mathcal{M} = \{1, M/2, M/2+1, M\}$ and $\mathcal{L}_{KL}$ is the Kullback-Leibler divergence function.

Homogeneous distillation. We propose homogeneous distillation between exiting heads with the same architectures to further improve performance. In each type of exiting heads, we employ the final heads as the teacher assistant to help the preceding homogeneous heads learn hint knowledge. For example, in all LPHs, the features of final LPH (*i.e.* at $M/2$-th exiting point) is utilized as reference feature for the preceding LPH. Given the feature maps from the first to m-th exiting heads $\mathbf{F}_m$, $(1 \le m \le M/2)$, the loss function of homogeneous distillation between LPHs is:

$$\mathcal{L}^{LPH}_{homo} = \frac{1}{M/2-1} \sum_{m=1}^{M/2-1} \mathcal{L}_{MSE}(\mathbf{F}_m, \mathbf{F}_{M/2}), \tag{10}$$

where $\mathcal{L}_{MSE}$ is the mean squared error function. Since the feature maps of GAH have different shapes, we apply dot-product operations between feature maps. Given a feature map of GAH $\mathbf{F}_m \in \mathbb{R}^{N/g^2(m)\times D}$ at the m-th exiting point, the shape can be transformed to $D \times D$ by computing $\mathbf{F}_m^T\mathbf{F}_m$. The loss function of homogeneous distillation between GAHs can be expressed as:

$$\mathcal{L}^{GAH}_{homo} = \frac{1}{M/2-1} \sum_{m=M/2+1}^{M-1} \mathcal{L}_{MSE}(\mathbf{F}_m^T\mathbf{F}_m, \mathbf{F}_M^T\mathbf{F}_M). \tag{11}$$

Therefore, the overall loss function of homogeneous distillation can be expressed as:

$$\mathcal{L}_{homo} = \mathcal{L}^{LPH}_{homo} + \mathcal{L}^{GAH}_{homo}. \tag{12}$$

Prediction distillation. In order to further improve the performance of internal classifiers, we utilize the final classifier as the reference label of $M/2$-th and M-th exiting points where last LPH and GAH are located, respectively. Given an input sample associated with label y, and assuming that the predictions at the $M/2$-th and M-th exiting points are $\hat{y}_{M/2}$ and $\hat{y}_M$, respectively, the loss function of prediction distillation can be formulated as:

$$\mathcal{L}_{pred} = \mathcal{L}_{KD}(\hat{y}_{M/2}, \hat{y}_L, y) + \mathcal{L}_{KD}(\hat{y}_M, \hat{y}_L, y), \tag{13}$$

where $\mathcal{L}_{KD}$ is the loss function of vanilla knowledge distillation:

$$\mathcal{L}_{KD}(\hat{y}^s, \hat{y}^t, y) = (1-\gamma)\mathcal{L}_{CE}(\hat{y}^s, y) + \gamma\mathcal{L}_{KL}(\hat{y}^s/T, \hat{y}^t/T). \tag{14}$$

Here, $\mathcal{L}_{CE}$ is the cross-entropy function, T is a temperature value to control the smoothness of logits, and γ is a balancing hyperparameter.

Hence, the overall loss function of our proposed method is:

$$\mathcal{L} = \alpha\mathcal{L}_{hete} + \beta\mathcal{L}_{homo} + \mathcal{L}_{pred}, \tag{15}$$

where α and β are hyperparameters.

3.5 Dynamic Inference

In this section, we first introduce the exiting metric and then depict the process of early exiting ViT inference. We employ a standard confidence metric following [28] as the exiting metric, which represents the probability of the most confident classification class. The prediction confidence c_m at the m-th exiting position is:

$$c_m(p^m) = \max_C p^m, \tag{16}$$

where p^m is the prediction distribution at m-th exiting position and C is the classification label set. During the inference process, input samples go through exits sequentially. Each sample dynamically adjusts its exiting path according to the exiting metric. If the classification confidence of a sample at the m-th exiting point exceeds a predefined threshold τ, the forward propagation of ViT will be terminated, and the prediction at the m-th exiting point will be output. The threshold τ can be adjusted according to computation cost and hardware resources to achieve an efficiency-accuracy trade-off. A low threshold may lead to a significant speed-up at the cost of a possible drop in accuracy. If the exiting condition is never reached, the ViT will revert to the standard inference process.

4 EXPERIMENT

In this section, we first introduce some implementation details and experimental settings. Then, we present the results of performance evaluations on three vision datasets and three popular ViT backbones. Finally, we conduct extensive ablation experiments to demonstrate the superiority of our methods.

4.1 Experimental Setup

Datasets. We evaluate our proposed method on three public vision datasets: CIFAR-100[11], Food-101[2], and ImageNet-1K[5]. The CIFAR-100 dataset contains 50K training images and 10K testing images, uniformly categorized into 100 classes. The Food-101 dataset consists of 101 food categories, with 750 training and 250 test images per category, making a total of 101K images. The ImageNet-1K dataset spans 1000 object classes and contains 1,281,167 training images, 50K validation images and 100K test images. We augment the training data with random crops, random horizontal flips and normalization, while the testing data is augmented with center crops and normalization.

Backbones. The proposed framework can be applied to a range of early-exit ViTs. Without loss of generality, we conduct experiments with three well-known ViT backbones, namely, ViT[7], DeiT[26], and Swin[15]. ViT is the first pure transformer structure for computer vision tasks and utilize a $[CLS]$ token to serve as the image representation for classification tasks. DeiT adds an additional distillation token to learn hard labels from the teacher model compared to ViT. Swin builds hierarchical feature maps by merging image patches in deeper layers. The $[CLS]$ token in LPH and GAH is replaced by the encoder output because it contains no $[CLS]$ token.

Baselines. We compare our dynamic early exiting methods with several representative early exiting methods. Considering most methods designed for CNNs and transformers in NLP, we transfer these methods to ViT for fair comparison.

- **SDN** [9] utilizes a weighted training strategy and employs a confidence-based criterion to decide whether to exit.

Table 1: Performance of different methods on three datasets for different ViT backbones. "Acc." represents the Top-1 classification accuracy. "#Params." represents the number of model parameters.

Methods	#Params.	Results: CIFAR-100			Results: Food-101			Results: ImageNet-1K		
		Acc.	MACs ↓	Speed-up ↑	Acc.	MACs ↓	Speed-up ↑	Acc.	MACs ↓	Speed-up ↑
ViT										
ViT-B/16	86 M	90.8 %	16.93 G	1.00 ×	89.6 %	16.93 G	1.00 ×	81.8 %	16.93 G	1.00 ×
SDN	94 M	86.5 %	10.16 G	1.64 ×	88.5 %	8.67 G	1.95 ×	79.5 %	10.95 G	1.55 ×
PABEE	94 M	85.1 %	11.48 G	1.52 ×	86.7 %	10.93 G	1.81 ×	78.6 %	12.41 G	1.36 ×
BERxiT	94 M	87.1 %	10.27 G	1.65 ×	88.3 %	8.56 G	1.98 ×	79.9 %	11.75 G	1.44 ×
ViT-EE	94 M	87.5 %	11.65 G	1.65 ×	88.2 %	10.42 G	1.91 ×	79.6 %	13.66 G	1.38 ×
PCEE	94 M	86.1 %	10.90 G	1.55 ×	88.1 %	9.50 G	1.81 ×	80.0 %	12.36 G	1.37 ×
Ours	101 M	**88.5** %	**9.76 G**	**1.87** ×	**88.6** %	**7.63 G**	**2.36** ×	**80.3** %	**10.65 G**	**1.70** ×
DeiT										
DeiT-B⚗	86 M	91.3 %	16.93 G	1.00 ×	90.3 %	16.93 G	1.00 ×	83.4 %	16.93 G	1.00 ×
SDN	94 M	87.4 %	9.65 G	1.75 ×	88.5 %	8.62 G	1.97 ×	77.5 %	11.30 G	1.50 ×
PABEE	94 M	86.4 %	11.43 G	1.48 ×	88.6 %	11.00 G	1.54 ×	78.5 %	12.40 G	1.36 ×
BERxiT	94 M	88.3 %	10.48 G	1.61 ×	88.8 %	9.16 G	1.85 ×	79.1 %	11.12 G	1.53 ×
ViT-EE	93 M	88.3 %	11.07 G	1.75 ×	88.8 %	10.26 G	1.91 ×	80.5 %	13.28 G	1.45 ×
PCEE	94 M	87.5 %	10.49 G	1.61 ×	88.6 %	9.59 G	1.76 ×	80.4 %	11.87 G	1.43 ×
Ours	102 M	**88.9** %	**9.54 G**	**1.91** ×	**89.5** %	**8.53 G**	**2.12** ×	**81.7** %	**10.90 G**	**1.67** ×
Swin										
Swin-B	87 M	92.6 %	15.13 G	1.00 ×	93.3 %	15.13 G	1.00 ×	83.5 %	15.40 G	1.00 ×
SDN	88 M	88.3 %	9.41 G	1.72 ×	90.2 %	7.64 G	2.17 ×	78.7 %	10.91 G	1.45 ×
PABEE	88 M	83.8 %	9.46 G	1.72 ×	88.8 %	8.17 G	2.01 ×	79.0 %	13.19 G	1.18 ×
BERxiT	88 M	88.4 %	9.61 G	1.68 ×	90.2 %	7.68 G	2.16 ×	80.2 %	10.35 G	1.54 ×
ViT-EE	91 M	88.1 %	9.71 G	1.82 ×	90.6 %	8.92 G	2.03 ×	82.1 %	11.20 G	1.52 ×
PCEE	88 M	88.1 %	10.68 G	1.50 ×	90.3 %	9.11 G	1.79 ×	79.8 %	11.51 G	1.37 ×
Ours	97 M	**90.7** %	**8.84 G**	**1.94** ×	**91.9** %	**7.05 G**	**2.50** ×	**82.7** %	**9.98 G**	**1.69** ×

- **PABEE** [38] employs a patience metric based on the consistency of classification decisions over several internal classifiers to make early exiting decisions.
- **BERxiT** [32] introduces an alternating training strategy to train the whole model.
- **ViT-EE** [1] utilizes a ViT encoder layer as its exiting head with the confidence criterion to decide whether to exit.
- **PCEE** [36] utilizes a patience&confidence criterion according to the enough number of confident predictions from consecutive internal classifiers.

Unless otherwise specified, the default exit architecture of baselines is a single fully connected layer that follows a pooler.

Evaluation metrics. Considering the trade-off between performance and efficiency, we employ Top-1 classification accuracy and speed-up ratio as the performance and efficiency metric, respectively. Since the measurement of runtime might not be stable, we follow [31] to calculate the speed-up ratio by comparing the actually executed layers in forward propagation and the complete layers. For an L-layer ViT, the speed-up ratio is defined as:

$$\text{Speed-up} = \frac{\sum_{i=1}^{L} L \times m^i}{\sum_{i=1}^{L} i \times m^i}, \tag{17}$$

where m^i is the number of samples that exit at the i-th layer of ViT. For clarity, we utilize the average multiply-accumulate operations (MACs) performed across the entire test dataset as a metric to assess the computational cost associated with a given model.

Implementation details. Our framework and all the compared methods are implemented using the Huggingface transformer library [29] for fair comparison. Most hyperparameters, such as learning rate, optimizer, and dropout probabilities are kept unchanged from the original backbones for fair comparison. We list different hyperparameters in the Appendix due to limited space. Each network is fine-tuned by 100 epochs on 3 NVIDIA 3090 GPUs, with a batch size of 64. There is no early stopping and the checkpoint after full fine-tuning is chosen.

Table 2: Ablation study results of main components. The "✓" mark indicating that we adopt the corresponding component. The opposite of LPH and GAH is fully connected layer with a pooler. The opposite of two-stage training is vanilla training.

LPH	GAH	Two-Stage training	Acc.	Speed-up
×	×	×	87.3 %	1.56 ×
×	×	✓	88.2 %	1.57 ×
×	✓	✓	88.3 %	1.71 ×
✓	✓	✓	**88.5** %	**1.87** ×

Table 3: Comparison of different exiting head architectures on CIFAR-100. MLP, Conv and Attention refers to utilizing a fully-connected layer, a 3×3 standard convolution layer, and a MHSA block respectively.

Exiting head	#Params.	MACs	Acc.	Speed-up
MLP [31]	91 M	10.78 G	88.2 %	1.57 ×
Conv [28]	129 M	13.86 G	87.5 %	1.72 ×
Attention [1]	105 M	12.76 G	88.0 %	1.58 ×
Ours	101 M	**9.76 G**	**88.5** %	**1.87** ×

4.2 Performance Evaluation

We conduct extensive experiments to compare our methods with the state-of-the-art methods for three ViT backbones on three vision datasets. Then we present the performance and efficient trade-off compared with other baselines.

Comparison with the state-of-the-art. We compare the performance between our methods and baselines on CIFAR-100, Food-101 and Tiny ImageNet datasets when different backbones are adopted, including ViT, DeiT, and Swin. The results are shown in Table 1. The original models for different backbones are ViT-L/16, DeiT-B and Swin-B respectively. We can find that our method can achieve approximately a 1.8 × speed-up ratio with only a 2% accuracy drop compared to the original models on most datasets, which significantly outperforms other baselines.

Performance and efficiency trade-off. To further verify the robustness and efficiency of our method, we visualize the performance and efficiency trade-off curves in Figure 1 on CIFAR-100 test set. The original backbone model is ViT-B/16. We compare five competitive baselines in the dynamic inference scenario. We can see that the performance of most early exiting methods drops dramatically when the speed-up ratio increases. This also reflects directly applying early exiting methods in ViT leads to unstable performance which cannot meet the requirements of real-time systems. However, our method is more robust to the variance of speed-up. If we set the almost same speed-up ratio, the accuracy drop of our method is 3.6 % lower than SDN method. When the accuracy is approximate to other baselines, our method can achieve faster speed-up. Moreover, our method can dynamically adjust the speed-up ratio without retraining, which is more feasible and friendly.

4.3 Ablation Study

To fully understand the impact of each part of the proposed framework, we conduct ablation study, where all experiments are evaluated on CIFAR-100 and utilize ViT as the backbone. We first study the effectiveness of the main components, and then analyze the impact of different exiting head architectures and training schemes. Finally, we verify the robustness of our methods for different numbers of predefined exiting heads.

Table 4: Comparison of different training schemes with the proposed exiting heads on CIFAR-100.

Training scheme	Accuracy	Speed-up
Normal [25]	86.9 %	1.82 ×
Weighted [9]	87.4 %	1.80 ×
Distillation [14]	86.9 %	1.83 ×
Alternating [32]	87.9 %	1.84 ×
Ours	**88.5** %	**1.87** ×

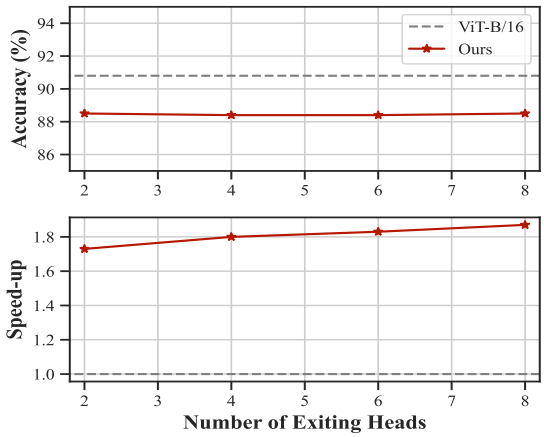

Figure 4: Results of accuracy and speed-up for different head number settings. The exiting heads are placed across all layers uniformly.

Ablation of main components. We design experiments to verify the effectiveness of the proposed LPH, GAH and two-stage training scheme. Table 2 presents the accuracy and speed-up ratio utilizing different components. The results show that for early exiting in ViT, heterogeneous exiting heads and two-stage training scheme are both significant. Specifically, we can observe that the two-stage training scheme can significantly improve accuracy. Moreover, when combined with LPH and GAH, the inference efficiency and accuracy can be further improved.

The architecture of exiting heads. In order to verify the effectiveness of the proposed heterogeneous exiting heads, we compare other competitive architectures using the same two-stage training scheme, as shown in Table 3. We can observe that the proposed heterogeneous exiting heads are crucial to achieve a speed-accuracy trade-off. Although utilizing MLP as exiting heads can gain approximate accuracy to ours, the speed-up ratio is low. The attention method can achieve a close trade-off between accuracy and speed but with high storage and computation requirements.

Training schemes. We compare our methods with four representative training schemes in early exiting methods, and they all utilize the proposed heterogeneous exiting heads. The accuracy and speed-up are shown in Table 4. Our method achieves the

highest classification accuracy and inference speed-up ratio, which significantly outperforms other training schemes.

The number of exiting heads. We analyze the influence on accuracy and speed-up when changing the number of predefined exiting heads, as shown in Figure 4. As the number of heads increases, the accuracy remains essentially consistent with only approximate 2 % accuracy drop, which shows that our method is robust to the number of heads and can tackle the overthinking problem [9]. Moreover, we can observe that the speed-up ratio can enhance with the increasing number of heads.

5 CONCLUSION

In this paper, we point out that naively applying early exiting in ViTs results in performance bottleneck due to insufficient feature representations in shallow internal classifiers and limited ability to capture target semantic information in deep internal classifiers. Based on this analysis, we propose an early exiting framework for general ViTs which combines heterogeneous exiting heads to enhance feature exploration. We also develop a novel two-stage training strategy to reduce information loss between heterogeneous exiting heads. We conduct extensive experiments for three ViT backbones on three vision datasets, demonstrating that our methods outperform other competitive counterparts. The limitation of our methods is to manually choose the exiting position and optimal exiting path. In the future, we intend to utilize Bayesian optimization to automatically perform the optimal exiting decision.

ACKNOWLEDGMENTS

This work was partially supported by the National Key Research and Development Program of China under No. 2021YFC3300200 and the National Natural Science Foundation of China under Grants No. 61971457.

REFERENCES

[1] Arian Bakhtiarnia, Qi Zhang, and Alexandros Iosifidis. 2021. Multi-Exit Vision Transformer for Dynamic Inference. In *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*. BMVA Press, 81.

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. 2014. Food-101 - Mining Discriminative Components with Random Forests. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI (Lecture Notes in Computer Science, Vol. 8694)*, David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer, 446–461.

[3] Hila Chefer, Shir Gur, and Lior Wolf. 2021. Transformer Interpretability Beyond Attention Visualization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 782–791.

[4] Richard J. Chen, Chengkuan Chen, Yicong Li, Tiffany Y. Chen, Andrew D. Trister, Rahul G. Krishnan, and Faisal Mahmood. 2022. Scaling Vision Transformers to Gigapixel Images via Hierarchical Self-Supervised Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 16123–16134.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Miami, Florida, USA). IEEE Computer Society, 248–255.

[6] Yifu Ding, Haotong Qin, Qinghua Yan, Zhenhua Chai, Junjie Liu, Xiaolin Wei, and Xianglong Liu. 2022. Towards Accurate Post-Training Quantization for Vision Transformer. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*, João Magalhães, Alberto Del Bimbo, Shin'ichi Satoh, Nicu Sebe, Xavier Alameda-Pineda, Qin Jin, Vincent Oria, and Laura Toni (Eds.). ACM, 5380–5388.

[7] 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. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

[8] Zhiwei Hao, Jianyuan Guo, Ding Jia, Kai Han, Yehui Tang, Chao Zhang, Han Hu, and Yunhe Wang. 2022. Learning efficient vision transformers via fine-grained manifold distillation. *Advances in Neural Information Processing Systems* 35 (2022), 9164–9175.

[9] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. 2019. Shallow-Deep Networks: Understanding and Mitigating Network Overthinking. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 3301–3310.

[10] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey E. Hinton. 2019. Similarity of Neural Network Representations Revisited. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 3519–3529.

[11] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images.* Technical Report. Toronto, Ontario.

[12] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[13] Woosuk Kwon, Sehoon Kim, Michael W Mahoney, Joseph Hassoun, Kurt Keutzer, and Amir Gholami. 2022. A Fast Post-Training Pruning Framework for Transformers. *arXiv preprint arXiv:2204.09656* (2022).

[14] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a Self-distilling BERT with Adaptive Inference Time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 6035–6044.

[15] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 9992–10002.

[16] Muhammad Maaz, Abdelrahman Shaker, Hisham Cholakkal, Salman H. Khan, Syed Waqas Zamir, Rao Muhammad Anwer, and Fahad Shahbaz Khan. 2022. EdgeNeXt: Efficiently Amalgamated CNN-Transformer Architecture for Mobile Vision Applications. In *Computer Vision - ECCV 2022 Workshops - Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part VII (Lecture Notes in Computer Science, Vol. 13807)*, Leonid Karlinsky, Tomer Michaeli, and Ko Nishino (Eds.). Springer, 3–20.

[17] Sachin Mehta and Mohammad Rastegari. 2022. MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

[18] Xuran Pan, Chunjiang Ge, Rui Lu, Shiji Song, Guanfu Chen, Zeyi Huang, and Gao Huang. 2022. On the Integration of Self-Attention and Convolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 805–815.

[19] Namuk Park and Songkuk Kim. 2022. How Do Vision Transformers Work?. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

[20] Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2023. On the effect of dropping layers of pre-trained transformer models. *Comput. Speech Lang.* 77 (2023), 101429.

[21] Li Shen, Yan Sun, Zhiyuan Yu, Liang Ding, Xinmei Tian, and Dacheng Tao. 2023. On Efficient Training of Large-Scale Deep Learning Models: A Literature Review. *arXiv preprint arXiv:2304.03589* (2023).

[22] Tianxiang Sun, Xiangyang Liu, Wei Zhu, Zhichao Geng, Lingling Wu, Yilong He, Yuan Ni, Guotong Xie, Xuanjing Huang, and Xipeng Qiu. 2022. A Simple Hash-Based Early Exiting Approach For Language Understanding and Generation. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, 2409–2421.

[23] Yehui Tang, Kai Han, Yunhe Wang, Chang Xu, Jianyuan Guo, Chao Xu, and Dacheng Tao. 2022. Patch Slimming for Efficient Vision Transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 12155–12164.

[24] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2023. Efficient Transformers: A Survey. *ACM Comput. Surv.* 55, 6 (2023), 109:1–109:28.

[25] Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. 2016. BranchyNet: Fast inference via early exiting from deep neural networks. In *23rd International Conference on Pattern Recognition, ICPR 2016, Cancún, Mexico, December 4-8, 2016*. IEEE, 2464–2469.

[26] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 10347–10357.

[27] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. 2021. Going deeper with Image Transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 32–42.

[28] Maciej Wolczyk, Bartosz Wójcik, Klaudia Balazy, Igor T. Podolak, Jacek Tabor, Marek Smieja, and Tomasz Trzcinski. 2021. Zero Time Waste: Recycling Predictions in Early Exit Neural Networks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (Eds.). 2516–2528.

[29] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

[30] Zhenyu Wu, Zhou Ren, Yi Wu, Zhangyang Wang, and Gang Hua. 2022. TxVAD: Improved Video Action Detection by Transformers. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*, João Magalhães, Alberto Del Bimbo, Shin'ichi Satoh, Nicu Sebe, Xavier Alameda-Pineda, Qin Jin, Vincent Oria, and Laura Toni (Eds.). ACM, 4605–4613.

[31] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic Early Exiting for Accelerating BERT Inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 2246–2251.

[32] Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. BERxiT: Early Exiting for BERT with Better Fine-Tuning and Extension to Regression. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*. Association for Computational Linguistics, 91–104.

[33] Li Yang, Mai Xu, Tie Liu, Liangyu Huo, and Xinbo Gao. 2022. TVFormer: Trajectory-guided Visual Quality Assessment on 360° Images with Transformers. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*, João Magalhães, Alberto Del Bimbo, Shin'ichi Satoh, Nicu Sebe, Xavier Alameda-Pineda, Qin Jin, Vincent Oria, and Laura Toni (Eds.). ACM, 799–808.

[34] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. 2022. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *European Conference on Computer Vision*. Springer, 191–207.

[35] Linfeng Zhang, Chenglong Bao, and Kaisheng Ma. 2022. Self-Distillation: Towards Efficient and Compact Neural Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 8 (2022), 4388–4403.

[36] Zhen Zhang, Wei Zhu, Jinfan Zhang, Peng Wang, Rize Jin, and Tae-Sun Chung. 2022. PCEE-BERT: Accelerating BERT Inference via Patient and Confident Early Exiting. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*. Association for Computational Linguistics, 327–338.

[37] Chuanyang Zheng, Zheyang Li, Kai Zhang, Zhi Yang, Wenming Tan, Jun Xiao, Ye Ren, and Shiliang Pu. 2022. SAViT: Structure-Aware Vision Transformer Pruning via Collaborative Optimization. In *NeurIPS*.

[38] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian J. McAuley, Ke Xu, and Furu Wei. 2020. BERT Loses Patience: Fast and Robust Inference with Early Exit. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

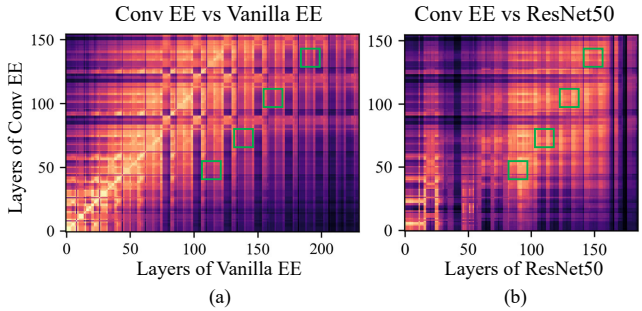

Figure 5: CKA heatmap comparing vanilla EE vs Conv EE and Conv EE vs ResNet50. Conv EE, which utilizes convolution as the exiting head, can learn different feature representations compared to vanilla EE. The layer incorporating convolution in the internal classifier closely resembles the lower half of the ResNet layers. Thus, Conv EE is more effective in capturing feature representations than vanilla EE.

A APPENDIX

A.1 Outline

In this supplementary material, we present a systematic investigation into the effectiveness of early exiting in ViTs. Besides, we provide more implementation details and experimental comparisons. The main content is summarized as follows:

- In Appendix A.2, we conduct a systematic investigation into the effectiveness of early exiting in ViTs and analyze the issues arising from the vanilla early exiting. Moreover, we obtain two observations: *1)* shallow internal classifiers cannot learn sufficient feature representation; *2)* deep internal classifiers cannot capture target semantic information.
- In Appendix A.3, we perform additional experiments to evaluate our framework. We measure the actual execution time of different methods and compare four representative training schemes with different widely-used exiting heads. Then, we ablate the effect of different exiting position schemes.

A.2 Investigation of Early Exiting in ViT

The early exiting method can halt the forward propagation of neural networks prematurely to provide a speed-accuracy trade-off, which has achieved significant performance improvements for CNNs and transformers in NLP. However, naively implementing early exiting on ViT may not yield performance gains for internal classifiers. For instance, the performance of the internal classifier on the fifth and tenth layers decreases by 21.8% and 4.0%, respectively, compared to the original classifier for ViT-B[7] on CIFAR-100[12]. Upon examining recent studies on ViT, we discover that a line of works focus on the integration of convolution and self-attention, which demonstrates that convolution operations at shallow layers can introduce additional inductive biases and capture more local information [7, 18]. Another line of works strive to exclusively employ self-attention as basic modules to construct the backbone with numerous layers for various vision tasks due to its exceptional capability in handling long-range dependencies [4, 19, 27]. However, it still remains unexplored that *1)* whether shallow internal classifiers could learn sufficient feature representation *2)* and whether deep internal classifiers could capture target semantic information. Therefore, we design the following probing experiments to answer these two questions and systematically analyze the working mechanism of early exiting methods in ViT.

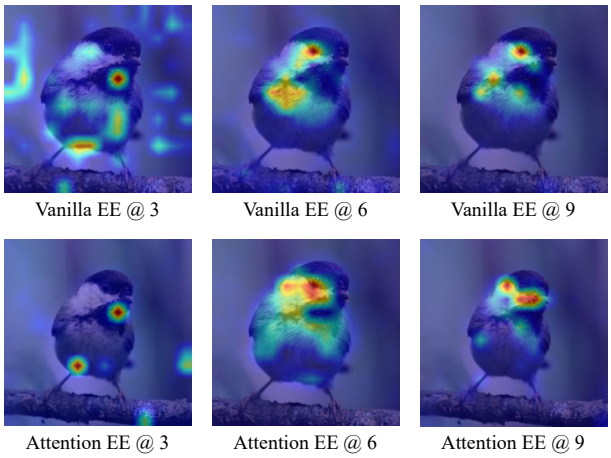

Figure 6: Comparison of attention maps for early exiting in ViT using either MLP (vanilla EE) or self-attention (Attention EE) as the exiting head. Attention EE methods are more capable of learning target semantic information by incorporating self-attention on deep internal classifiers than vanilla EE.

Regrading the first question, we initially compare the representation similarity of two early exiting (EE) architectures, namely MLP (vanilla EE) and convolution (Conv EE), and subsequently assess the similarity between Conv EE and ResNet. We employ centered kernel alignment (CKA) [10] as the similarity metric, which facilitates quantitative comparisons of representations similarities within and across neural networks. It is important to note that we compare their representation similarities using outputs from all internal classifiers and intermediate layers. The results are evaluated on the ViT-B/16 backbone using CIFAR-100 [12]. We only utilize standard 3×3 convolution in Conv EE for fair comparison. Figure 5 displays the CKA similarity results as heatmaps, with the x and y axes indexing the layers from input to output. The layers attached MLP or convolution are marked with green boxes. Lighter colors in the heatmap indicate a higher representation similarity between the corresponding layers. We observe that the layer at which the internal classifier is attached in vanilla EE differs from that in Conv EE, suggesting that they extract distinct information, as depicted in Figure 5 (a). The layer where the convolution exiting architecture is positioned exhibits high similarity to ResNet, as shown in Figure 5 (b). Consequently, the convolution exiting architecture assists ViT in capturing local information and strengthening feature representations, allowing Conv EE to learn representations akin to those of ResNet.

Concerning the second question, we compare the ability to extract target semantic information between two different early exiting architectures, namely vanilla EE and self-attention (attention EE) at different exit positions. We compute the attention map and visualize it upon the input image following [3]. The attention map highlights target pixels of the image that contribute to the dominance of the predicted label, enabling the analysis of the efficacy of

Table 6: Comparison of different training schemes with widely-used exiting heads.

Exiting heads	Training schemes	Acc.	Speed-Up
MLP	Normal	87.3 %	1.56 ×
MLP	Weighted	88.2 %	1.53 ×
MLP	Distillation	87.1 %	1.57 ×
MLP	Alternating	88.1 %	1.54 ×
MLP	**Ours**	**88.2** %	**1.57** ×
Conv	Normal	85.2 %	1.69 ×
Conv	Weighted	86.4 %	1.65 ×
Conv	Distillation	84.9 %	1.64 ×
Conv	Alternating	86.7 %	1.67 ×
Conv	**Ours**	**87.5** %	**1.72** ×
Attention	Normal	86.8 %	1.54 ×
Attention	Weighted	87.8 %	1.53 ×
Attention	Distillation	87.0 %	1.57 ×
Attention	Alternating	87.5 %	1.54 ×
Attention	**Ours**	**88.0** %	**1.58** ×
Ours	Normal	86.9 %	1.82 ×
Ours	Weighted	87.4 %	1.80 ×
Ours	Distillation	86.9 %	1.83 ×
Ours	Alternating	87.9 %	1.84 ×
Ours	**Ours**	**88.5** %	**1.87** ×

Table 5: Comparison of execution time on a RTX 3090 GPU.

Method	MACs	Acc.	Execution time
ViT-B	16.93 G	90.8 %	6.49 (±0.11) ms
SDN	10.16 G	86.5 %	5.65 (±0.23) ms
PABEE	11.48 G	85.1 %	5.86 (±0.09) ms
PCEE	10.90 G	86.1 %	6.70 (±0.24) ms
BERxiT	10.27 G	87.1 %	5.75 (±0.13) ms
ViT-EE	11.65 G	87.5 %	5.35 (±0.14) ms
Ours	**9.76 G**	**88.5** %	**5.03** (±0.16) ms

semantic information extraction from the input space. We employ DeiT-B [26] as the backbone, which comprises twelve layers in total. The attention map results for the third, sixth, and ninth layers are presented in Figure 6. It becomes more evident that the eye and beak of the bird are target parts for object identification using the attention EE method than the vanilla EE method, especially on the deeper layer, such as the ninth layer. Since the attention EE method employs self-attention as the exiting architecture, the ability to extract semantic features and spatial relationships can be further enhanced. As a result, incorporating self-attention on deep layers can help learn more semantic representations and capture richer global information than the vanilla EE method.

Based on the aforementioned analyses, we obtain the following observations:

- ***Observation 1*:** Shallow internal classifiers cannot learn sufficient feature representation.
- ***Observation 2*:** Deep internal classifiers cannot capture target semantic information.

Insight. We identify the primary reason for the poor performance resulting from directly applying the early exiting strategy in ViT. Furthermore, we discover that integrating convolution on shallow internal classifiers can enhance local information exploration, while incorporating self-attention on deep layers can improve the ability to obtain global information. Consequently, if both convolution and self-attention are employed as exiting architectures, positioned on shallow and deep layers respectively, the model would gain access to a more comprehensive combination of local and global information compared to using only MLP as the exiting architecture.

A.3 Additional Experiments

In this section, we evaluate the actual execution time of our method and compare four representative training schemes with different widely-used exiting heads. Moreover, we ablate the influences of different exiting position schemes.

Execution time. We design experiments to measure the actual execution time with batch 1 on a RTX 3090 GPU. For each method, we run it for once as a warm-up and then record the execution time with 50 runs without break for the whole testing set of CIFAR-100. The results are shown in Table 5. We can find that our method achieves the highest accuracy, the lowest running time and lowest computation cost.

Exiting heads & training schemes. We compare four representative training schemes with different widely-used exiting heads for the ViT backbone on CIFAR-100. The results of different training schemes with MLP, Conv, Attention and the proposed exiting heads are shown in Table 6. We observe that our training scheme can improve classification accuracy and accelerate inference speed with different exiting heads.

Table 7: Comparison of different exiting positions on CIFAR-100.

Position scheme	Exiting position	Acc.	Speed-Up
Shallow	{2,3,4,5}	85.3 %	1.76 ×
Deep	{8,9,10,11}	87.9 %	1.46 ×
Middle	{6,7,8,9}	87.5 %	1.74 ×
Uniform (**Ours**)	{4,6,8,10}	**88.4** %	**1.81** ×

Exiting position. In this work, all exiting heads are positioned across all layers uniformly according to an approximately equidistant computational distribution, i.e., the multiply-accumulate operations (MACs) of intermediate blocks between two adjacent exiting points remain consistent. We carefully ablate the influences of different exiting position schemes, including positioning at shallow, deep, and middle layers. The results of different exiting positions for the ViT backbone on CIFAR-100 dataset are shown in Table 7.