Optimising Vision Transformer Performance on Limited Datasets: A Multi-Gradient Approach

Mohsin Ali[1]*, Haider Raza[1], John Q. Gan[1], Muhammad Haris[2]
[1]School of Computer Science and Electronics Engineering, University of Essex
[2]Mohamed Bin Zayed University of Artificial Intelligence
{ma22159, h.raza, jqgan}@essex.ac.uk, muhammad.haris@mbzuai.ac.ae

Abstract

Vision Transformers (ViTs) are well-known for capturing the global context of images using Multi-head Self-Attention (MHSA). However, compared to Convolutional Neural Networks (CNNs), ViTs typically exhibit a reduced inductive bias and require a larger volume of training image data to learn local feature representations. While various methods like the integration of CNN features or advanced pre-training strategies have been proposed to introduce this inductive bias, they often require significant architectural modifications or rely heavily on expansive pre-training datasets. This paper introduces a novel approach for training ViTs on limited datasets without altering the ViT architecture. We propose the Multi-Gradient Image Transformer (MGiT), which utilizes a parallel training method with a compact auxiliary ViT to adaptively optimize the weights of the target ViT. This approach yields significant performance improvements across diverse datasets and training scenarios. Our findings demonstrate that MGiT enhances ViT efficiency more effectively than traditional training methods. Furthermore, the application of Jensen-Shannon (JS) Divergence validates the convergence and alignment of feature understanding between the primary and auxiliary ViTs, thereby stabilizing the training process. The code is available at https://github.com/game-sys/Multi-Gradient-Image-Transformer-MGiT-

1. Introduction

Vision Transformers (ViTs) have shown success in various mainstream Computer Vision (CV) tasks, e.g., image classification, object detection, segmentation, and generation [7, 9, 17]. ViTs process images through a grid of patches to capture global contexts, which is in contrast to the local receptive fields of CNNs, such as ResNet and EfficientNet [11, 26]. Despite their advantages, ViTs lack certain inductive biases, which limits their efficacy on smaller datasets without extensive pre-training, as opposed to CNNs that perform well on datasets such as ImageNet-1K [8], CIFAR-10 [12] and CIFAR-100 [12]. This lack of inductive bias means that ViTs are not inherently equipped to understand local patterns, for instance, recognizing relationships between neighboring pixels within an image, which are often crucial for tasks on smaller datasets [11]. To mitigate this, various methods have been proposed to enhance the inductive biases in ViTs, such as integrating CNN features or using sophisticated pre-training strategies [7, 16, 29]. However, these methods often involve significant modifications to the architecture or rely heavily on large pre-training datasets.

Moreover, plug-and-play methods such as Data-efficient Image Transformer (DeiT) and dense relative localization loss (Drloc) were developed to enhance ViTs' performance without overhauling their architecture, proving effective on medium-scale datasets [10, 15, 27]. Multi-Gradient Image Transformer (MGiT) introduces an auxiliary ViT that is similar to the original ViT proposed by Dosovitskiy [9] in architecture but less complex in terms of layers and heads without architectural changes [18, 23]. The rationale behind this approach is two-fold: 1) *Auxiliary Learning*: The auxiliary ViT, with its simpler structure, learns more generalized features that stabilize and guide the training of the more complex primary ViT. This is further illustrated in Figure 2, which shows both networks converging to similar probability distributions, effectively giving the primary ViT a significant head start in training. This can be particularly beneficial during the initial stages of training when the primary ViT is prone to overfitting on small datasets. 2) *Gradient Sharing*: By utilizing the gradients from the auxiliary ViT, which has undergone pre-training for several initial epochs, we apply a form of regularization to the training process of the primary ViT. This method uses the insights from the auxiliary ViT to guide and stabilize the primary ViT's learning, helping to improve its generalization and reduce the risk of overfitting. This pre-training equips the auxiliary ViT with preliminary knowledge about the dataset, making

its gradients less noisy and more stabilizing. When these refined gradients are used to update the primary ViT's classification layer, they help guide the primary model towards a more rapid and stable convergence by aligning it with the already acquired insights of the auxiliary model as shown in Figure 2. Subsequently, both the primary and auxiliary ViTs are trained in parallel. This parallel training allows the primary ViT to continuously benefit from the auxiliary ViT's evolving understanding of the dataset. We use Jensen-Shannon (JS) Divergence to confirm the effectiveness of our gradient-sharing method. This metric helps us verify that both the auxiliary and primary ViTs are developing a consistent understanding of the dataset as they train. This metric quantifies the similarity in the distribution of features or outputs between the two networks throughout their training, providing empirical evidence that the auxiliary ViT's guidance effectively enhances the primary ViT's learning process. Although both ViTs lack inductive biases, the simpler design of the auxiliary ViT, characterized by fewer layers, plays a crucial role in minimizing overfitting during training and provides a more controlled learning environment for the primary ViT. This innovative approach allows for auxiliary learning and gradient sharing without necessitating architectural changes or extensive pre-training, making it an efficient plug-and-play solution for enhancing ViT performance on smaller datasets. This method is detailed through empirical analysis across multiple benchmark datasets and compared with existing techniques, showing notable improvements in both scratch training and transfer learning scenarios [2, 12, 20, 21].

The main contributions of this paper are summarized as follows:

1. We introduce **MGiT** (Multi-Gradient Image Transformer), a novel plug-and-play strategy that takes advantage of gradients from an auxiliary model to update the weights of the classification layer of ViTs.
2. We provide comprehensive evaluations of MGiT on multiple benchmarks such as CIFAR-10, CIFAR-100, Pet-37, Flower-102 and Food-101. Our experimental results demonstrate that MGiT consistently outperforms existing methods, achieving performance improvements ranging from 1% to 10% across various scenarios.

2. Related Work and Background

Hybrid-ViT

ViTs have excelled in image classification, leading researchers to adapt their architecture for diverse computer vision (CV) tasks [1, 3, 6, 9, 25, 27]. The main challenge with ViTs, their lack of inductive bias in small datasets, has spurred the development of hybrid architectures that integrate CNN operations to enhance spatial understanding. Hybrid models such as SWIN [16], CVT [29], and T2T [30] modify the token transformation process to maintain the image's grid structure, introducing biases that help preserve local features. ConTraCon [24] applies these concepts to continuous learning, aiming for adaptability without redesigning the core architecture.

Plug-and-Play Methods

The DeiT model [27] employs knowledge distillation from a large CNN, such as Efficientnet [26], allowing it to perform robustly without extensive data [26]. Drloc [15] improves ViTs by introducing relative positional embeddings, enhancing the understanding of token relationships within the model. New approaches such as DropKey [14] and ES [5] optimize the self-attention mechanism by adjusting dropout placement and conceptualizing ViTs as ensemble networks with varied path lengths. Techniques such as OFDB [19] innovate data augmentation and SPT [13] address locality bias, enhancing training efficiency on smaller datasets. This breadth of plug-and-play methods demonstrates the ongoing evolution and versatility of ViT architectures.

3. Proposed Method

Section 3.1 provides an overview of the proposed method. Section 3.1.1 discusses the application of multi-gradients, and Section 3.1.2 elaborates on the proposed weighted average mechanism and early-stage training. Section 3.2 describes the architecture of the auxiliary ViT. Finally, Section 3.3 details the use of JS Divergence to validate the convergence of the primary and auxiliary ViTs.

3.1. MGiT: A Multi-Gradient Approach

The core concept proposed in this paper, Multi-Gradient Image Transformer (MGiT), is to introduce a plug-and-play methodology that seamlessly integrates with the primary architecture of any ViT, requiring no structural modifications. Central to this approach is the incorporation of a relatively compact ViT model referred to as the 'auxiliary ViT'. Both the primary and auxiliary ViTs are trained in parallel, and crucially, both models are initialized with identical weights to ensure a uniform starting point for their training. In this setup, although both models are trained on the same subset of data, the auxiliary ViT, being a simplified version of the primary, processes data differently due to its reduced complexity. This structural simplification allows the auxiliary ViT to learn more efficiently on small datasets, influencing the nature of the gradients it produces.**Generalization:** The auxiliary ViT helps in capturing generalized patterns that are beneficial for stabilizing the training process of the primary ViT. **Regularization through Gradient Sharing:** Using the gradients of the auxiliary ViT to update the primary ViT's classification layer introduces regularization,

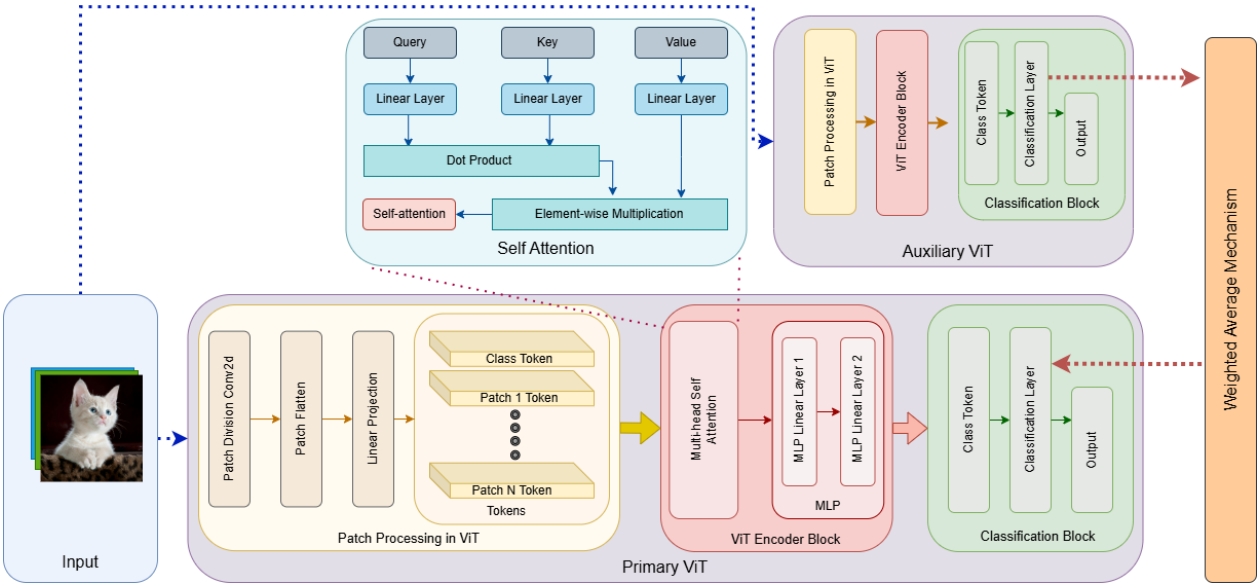

Figure 1. Overall architecture of our multi-gradient ViT training method. We begin with the parallel training of the primary ViT and auxiliary ViT models. Gradients from the Plugin ViT are transferred to the primary ViT's classification layer through a Weighted Average Mechanism, with the weight contribution of the Plugin ViT gradients reducing over time.

helping in better weight updates and reducing the risk of overfitting.

3.1.1. Multi-Gradient Approach

The key idea is to use the gradient for the classification layer obtained from the auxiliary ViT to update the classification layer of the primary ViT, as shown in Figure 1 and Equation 1. The initial motivation for our approach was highlighted during preliminary experiments, where we observed a significant improvement in accuracy when replacing the random weights in the classification layer of ViT models with pre-trained ImageNet (IN) weights. While the substitution with IN weights did enhance performance, the application of this strategy is inherently constrained. Specifically, the availability of IN weights compatible with a given ViT architecture is not always guaranteed, limiting the applicability of this direct replacement method across different models. This motivates us to develop a strategy to optimise the performance of ViT models more broadly. Instead of relying solely on the availability of IN weights, our approach focuses on refining the classification layer's weights through a methodical training process. By employing an auxiliary ViT that has been pre-trained to acquire preliminary dataset insights, we harness its more refined and stable gradients to update the primary ViT's classification layer. This not only circumvents the limitations posed by the dependency on IN weights but also ensures that the primary ViT can effectively leverage enhanced learning dynamics to improve overall performance. This strategic optimization of the classification layer opens up broader possibilities for applying ViT models across various architectures and datasets without the constraints of pre-trained weight compatibility.

$$\mathrm{UW}\left(\mathbf{W}, \mathbf{G}, \alpha\right) = \mathbf{W} - \alpha \cdot \mathbf{G} \tag{1}$$

where $\mathbf{UW}$ represents the updated weights, $\mathbf{W}$ signifies the weights of the primary ViT, $\mathbf{G}$ represents the gradient of the auxiliary ViT, and α is the learning rate. It is important to note that the primary ViT is indeed trained using its own gradients as well. The gradients from the auxiliary ViT are used specifically for the classification layer to introduce regularization and improve convergence. The rest of the layers in the primary ViT continue to be updated using their own gradients computed during the backpropagation process. This dual-gradient approach ensures that the primary ViT retains its ability to learn from the dataset while benefiting from the stabilizing influence of the auxiliary ViT's gradients.

3.1.2. Weighted Average Mechanism and Early Stage Training

Moreover, during early experiments, it was observed that directly adopting weights from the auxiliary model degrades performance, due to potential distribution mismatches between the learned features of the auxiliary ViT and those required by the primary ViT. To address this issue, a weighted average mechanism is employed as shown in Figure 1 and Equation 2. This mechanism ensures a smoother transition, mitigating distribution discrepancies, and facilitating a more seamless integration of knowledge. The update mechanism utilizes a weighted average and a technique

is applied to progressively reduce the weight of the auxiliary ViT knowledge. This reduction strategy enhances the independence and self-learning capabilities of the target ViT model. By gradually transferring knowledge from the parallel-trained, simplified auxiliary ViT, the primary ViT explores and adapts to the new task without relying on random initialization. This process effectively reduces the search space of the primary ViT, contributing to improved adaptability and performance. Further, it was noticed during the experiments that training the auxiliary ViT model for a few epochs (Early Stage Training) before utilizing its gradient to update the classification layer of the primary ViT helps to improve the performance of the primary ViT model. This approach of initial training allows the auxiliary ViT to adapt its weights according to the new dataset, facilitating effective domain adaptation.

$$\mathrm{WA}_{\mathrm{weights}}(\mathbf{UW}, \mathbf{T}, \lambda) = \lambda \cdot \mathbf{UW} + (1 - \lambda) \cdot \mathbf{T} \quad (2)$$

where **WA** denote the weighted average and **UW** denotes the updated weights obtained through the gradient of the auxiliary ViT, **T** signifies the target weights from the primary ViT, and λ serves as the weight parameter, further discussed in Section 3.2.

Parallel training was limited to 20 epochs to ensure the complexity of MGiT remains manageable for small dataset training. After 20 epochs, the auxiliary ViT no longer transfers gradients to the primary ViT, and the primary ViT continues to train using its own gradients. This strategy ensures that while the auxiliary ViT provides initial stabilization and regularization, the computational overhead is limited, and the primary ViT can adapt independently to the dataset. This methodology not only addresses the challenges of training a ViT from scratch but is also applicable to scenarios where a ViT model is trained on a small dataset with pre-existing IN weights. In conventional practices, adapting a pre-trained ViT model to a new dataset involves introducing a new classification layer with random weights. However, our proposed MGiT eliminates the need for such random initialization, streamlining the learning process. Remarkably, this novel approach demonstrates substantial performance improvements, especially in situations where pre-trained weights are unavailable. Furthermore, it proves beneficial for enhancing performance in transfer learning scenarios. By eliminating random initialization and promoting adaptability, it provides a versatile solution for improving ViT model performance and handles domain adaptation.

3.2. Auxiliary ViT

Similar to the primary ViT, the auxiliary ViT comprises a comparable architecture. However, the primary deviation in the auxiliary ViT lies in a concerted effort to reduce the complexity of parallel training. One distinctive feature of the auxiliary ViT is the deliberate reduction in the number of layers (L) compared to the primary ViT (the primary number of layers: L/α where $\alpha = 6$). In initial experiments, employing an auxiliary ViT of the same size as the primary ViT increased the method complexity without significant improvement. For further details, as shown in Table 1. This strategic reduction in the number of layers was undertaken to enhance computational efficiency while preserving competitive performance. Additionally, the number of MHSA heads (H) is reduced to further alleviate complexity (H/ω where $\omega = 2$). To integrate the designed auxiliary ViT with the primary ViT, both models are trained in parallel. The gradient of the auxiliary ViT model is utilized to update the classification block of the primary ViT. Notably, conducting an early-stage training of the ViT for a certain number of epochs before initiating the parallel training has been observed to enhance model performance. In this study, we refer to this initial training phase as 'early stage training', comprising ($epoch = 25$) epochs. After the early stage of training, a weighted average mechanism is introduced to facilitate the transfer of knowledge from the auxiliary ViT to the primary ViT. This mechanism involves a weighted average between the weights of the auxiliary ViT's classification block and the primary ViT, with the coefficient starting at $\lambda = 0.7$. Furthermore, this coefficient is gradually reduced after a certain number of epochs (step=20) with a reduction factor of ($\gamma = 0.2$). The parallel training of the auxiliary ViT is stopped after this stage when the weight becomes $\lambda <= 0.2$. These hyper-parameters are selected through experiments with multiple values. Please refer to the supplementary for details. The proposed auxiliary ViT aims to strike a balance between complexity and accuracy. By employing parallel training with the primary ViT, the auxiliary ViT demonstrates improved performance across multiple datasets, particularly in scenarios with limited data.

3.3. Validation of Auxiliary ViT Guidance via Jensen-Shannon Divergence

To quantitatively validate the effectiveness of the auxiliary ViT in guiding the primary ViT, we utilize Jensen-Shannon (JS) divergence instead of Kullback-Leibler (KL) divergence. The JS divergence, which is symmetric and bounded between 0 and 1, offers a stable and interpretable similarity measure, providing a consistent scale from perfect similarity to complete divergence. These properties make JS divergence particularly suitable for comparing the probability distributions of the features processed by both models, ensuring that the comparison is fair and not influenced by the order of the distributions. The divergence between the output layers from the primary and auxiliary ViTs is computed at each epoch during the training process, as outlined in the

following equation:

$$JS(P\|Q) = \frac{1}{2}KL(P\|M) + \frac{1}{2}KL(Q\|M) \quad (3)$$

where P and Q are the probability distributions of the outputs from the primary and auxiliary ViTs, respectively, M is the mean of P and Q, and KL denotes the Kullback-Leibler divergence. This measure provides insights into how the feature understanding between the two models converges over time. A decreasing trend in JS divergence suggests that the auxiliary ViT is becoming more effective in aligning its understanding with that of the primary ViT, thereby justifying its role in stabilizing the training process and providing useful gradients for updating the primary ViT's classification layer.

4. Experiments

Experiments are conducted in this section to show the effectiveness of the proposed method (MGiT) in improving the performance of different ViT architectures on multiple datasets. Section 4.1 describes the experimental setup, including the details of the used datasets, along with the details of different ViT architectures. Section 4.3 shows the quantitative results conducted on different datasets and ViT models by training from scratch. Similar results with pretrained IN weights are shown in Section 4.4. Furthermore, we compare the proposed method with available plug-and-play methods in Section 4.5

Table 1. Effect of Layer Reduction or α on Number of Model Parameters, GFLOPs, Accuracy on Cifar-10 and Cifar-100 Using ViT-B/32

α	Para (Millions)	GFLOPs	Cifar-10	Cifar-100
1	88	94.44	86.45	61.96
2	45	49.08	86.38	61.91
4	24	26.4	86.13	61.79
6	17	18.84	86.42	61.88
8	10	11.28	85.07	60.56

4.1. Experimental Setup

We employed the PyTorch framework [22] and used publicly available implementations of models such as ViT [9], SWIN [16], CVT [29], and T2T [30] for our experiments. These were conducted on multiple GPUs, including 2x RTX 2080Ti and 1x 3080Ti, with each system equipped with 64GB of RAM. Our research aims to enhance ViT performance on small datasets, exemplified by experiments ranging from CIFAR-10, with 5000 images per class, to Flower-102's 20 images per class. We reduced model complexity by adjusting the auxiliary ViT's layers through a scaling factor α, with $\alpha = 6$ yielding a balance of 17 million parameters and 18.84 GFLOPs, achieving 86.42% accuracy on CIFAR-10 and 61.88% on CIFAR-100, as shown in Table 1.

4.1.1. ViT Models and Complexity

To enhance ViT model performance via the plug-and-play method, we kept the core architecture unchanged. ViT models, including ViT-B [9], ViT-Ti, and ViT-S as described by Mathilde et al [4], were used in their default configurations. This approach extends to other variants such as SWIN-S, SWIN-T [16], CVT [29], and T2T [30], adhering to the original specifications set by their creators. Tables 2 and 3 compare the complexity of the MGiT and standard ViT models. It is important to note that the introduction of the auxiliary ViT does add some computational overhead. The computational overhead is primarily constrained to the initial training phase, where the auxiliary ViT operates independently for the first 25 epochs to establish a solid foundation for the training process. Following this period, both the primary and auxiliary ViTs train in parallel for an additional 20 epochs, during which the auxiliary ViT continues to support the primary ViT in setting a robust foundation. Once the initial training phase is completed, the computational load is significantly reduced. This strategy ensures that the method remains scalable even when applied to larger and more complex ViT models or datasets.

4.1.2. Training Configuration

MGiT and all other models in this study use the same augmentation techniques as those outlined in DeiT [28], ensuring equitable comparisons. We standardized the input image size to 224 x 224 for all models, with batch sizes adjusted between 32 and 128 to optimize GPU memory usage. Optimization was performed using the Adam optimizer at an initial rate of 0.0001, reduced by a weight decay factor of 0.5. Training included an early stopping mechanism, set at a patience of 20 for fine-tuning and 40 for scratch training. To verify the generalizability of the results, each experiment was conducted three times, with average accuracies reported in Tables 2, 3, 4, and 5.

4.2. Analysis of ViTs Using JS Divergence

Figure 2 illustrates the JS divergence across 20 training epochs for various configurations of the ViT models. As observed, the JS divergence values decrease significantly in the initial epochs and stabilize towards the later epochs, indicating a successful alignment of feature distributions post-initial rapid convergence. This stabilization aligns with the expected outcomes of our weighted sum approach in the classification layer, where the influence of the auxiliary ViT's gradients is reduced over time as outlined in Section 3.1.2.

Table 2. Model Performance trained without using IN weights on multiple ViT models and various datasets

Model	Para(M)	GFLOPs	C-10	C-100	Pet-37	Fr-102	Fd-101
ViT							
ViT-Ti/32 [4]	6.1	0.2	80.64	58.13	15.41	29.07	46.81
ViT-Ti/32 + MGiT	7.8	0.25	81.95	58.72	17.38	30.24	50.11
			↑ 1.31	↑ 0.59	↑ 1.97	↑ 1.17	↑ 3.30
ViT-S/32 [4]	22.8	0.76	82.88	58.36	16.43	36.28	46.19
ViT-S/32 + MGiT	27.9	0.93	83.23	61.61	20.16	41.64	50.95
			↑ 0.35	↑ 3.25	↑ 3.73	↑ 5.36	↑ 4.76
ViT-B/32 [9]	88	2.95	84.78	61.13	13.81	38.83	45.25
ViT-B/32 + MGiT	105.3	3.53	86.42	61.88	19.46	50.7	49.94
			↑ 1.64	↑ 0.75	↑ 5.65	↑ 11.87	↑ 4.69
ViT-Ti/16 [4]	5.7	0.73	85.47	62.4	22.62	39.09	57.23
ViT-Ti/16 + MGiT	6.9	0.87	85.74	63.03	27.11	39.78	60.95
			↑ 0.27	↑ 0.63	↑ 4.49	↑ 0.69	↑ 3.72
ViT-S/16 [4]	22	2.85	87.19	63.99	22.75	51.5	60.83
ViT-S/16 + MGiT	26.2	3.37	88.3	64.33	33.33	53.37	61.76
			↑ 1.11	↑ 0.34	↑ 10.58	↑ 1.87	↑ 0.93
SWIN ViT							
SWIN-T [16]	28.2	2.97	86.9	67.87	18.91	42.93	59.24
SWIN-T + MGiT	47.6	4.77	87.89	68.73	26.76	55.14	70.34
			↑ 0.99	↑ 0.86	↑ 7.85	↑ 12.21	↑ 11.10
SWIN-S [16]	49.6	5.75	56.41	48.48	7.05	35.04	54.71
SWIN-S + MGiT	72.5	8.03	84.22	54.45	7.6	49.78	69.31
			↑ 27.81	↑ 5.97	↑ 0.55	↑ 14.74	↑ 14.60
CVT							
CVT-13 [29]	17.9	3.81	93.47	75.09	39.98	63.88	77.95
CVT-13 + MGiT	21.4	5.01	93.5	75.17	40.55	64.51	77.4
			↑ 0.03	↑ 0.08	↑ 0.57	↑ 0.63	↓ 0.55
T2T							
T2T-14 [30]	4.9	9.6	84.34	61.95	38.43	56.62	60.76
T2T-14 + MGiT	7.3	17.3	86.19	63.82	37.41	59	61.85
			↑ 1.85	↑ 1.87	↑ 1.02	↑ 2.38	↑ 1.09

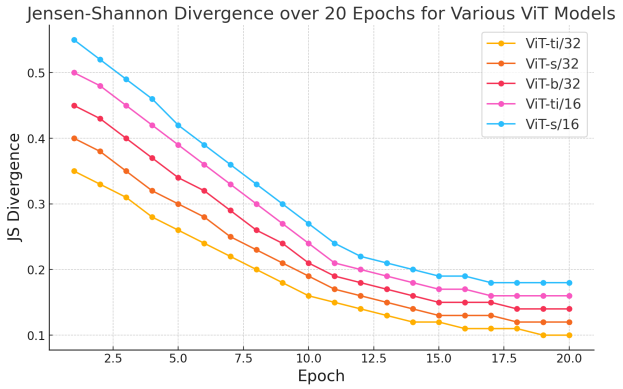

Figure 2. Jensen-Shannon Divergence across 20 epochs for various ViT configurations, demonstrating convergence in feature distribution understanding between the primary and auxiliary ViTs.

4.3. Training from Scratch

In this section, we provide a comprehensive empirical analysis of the proposed model across various datasets, ranging from small to medium-sized. The primary objective is to assess the performance of the proposed model in scenarios represented by limited data or computational resources. It is crucial to highlight that while the common practice in ViT training involves using pre-trained weights, this study also analyzes the performance of the proposed model when trained from scratch. This exploration is essential as there are situations where using pre-trained weights may not be feasible or practical. Instances of this include domain shifts, such as in medical or satellite images, or modifications to the ViT model's backbone architecture. In such cases, ViT models must undergo training from scratch on specific datasets, making it crucial to evaluate their performance under these conditions. Table 2 provides detailed results for

Table 3. Model Performance trained using IN pre-trained weights on multiple ViT models and various datasets

Model	Pr. (M)	GFLOPs	C-10	C-100	Pet-37	Fr-102	Fd-101
ViT							
ViT-Ti/32 [4]	6.1	0.2	94.87	83.03	85.9	86.23	73.1
ViT-Ti/32 + MGiT	7.8	0.25	95.28	83.78	86.33	87.11	74.25
			↑ 0.41	↑ 0.75	↑ 0.43	↑ 0.88	↑ 1.15
ViT-S/32 [4]	22.8	0.76	95.61	83.15	87.23	89.36	79.51
ViT-S/32 + MGiT	27.9	0.93	95.79	84.29	88.78	89.63	79.95
			↑ 0.18	↑ 1.14	↑ 1.55	↑ 0.27	↑ 0.44
ViT-B/32 [9]	88	2.95	97.88	87.19	88.22	91.23	81.49
ViT-B/32 + MGiT	105.3	3.53	98.57	90.03	90.16	93.94	82.52
			↑ 0.69	↑ 2.84	↑ 1.94	↑ 2.71	↑ 1.03
ViT-Ti/16 [4]	5.7	0.73	98.11	88.65	91.57	92.7	88.12
ViT-Ti/16 + MGiT	6.9	0.87	98.74	88.69	91.72	92.98	89.22
			↑ 0.63	↑ 0.04	↑ 0.15	↑ 0.28	↑ 1.10
ViT-S/16 [4]	22	2.85	98.93	89.15	91.71	94.81	89.27
ViT-S/16 + MGiT	26.2	3.37	99.1	89.66	92.21	95.37	89.64
			↑ 0.17	↑ 0.51	↑ 0.50	↑ 0.56	↑ 0.37
SWIN ViT							
SWIN T [16]	28.2	2.97	97.46	88.76	89.87	96.04	88.4
SWIN T + MGiT	47.6	4.77	98.11	90.14	89.98	97.12	87.51
			↑ 0.65	↑ 1.38	↑ 0.11	↑ 1.08	↓ 0.89
SWIN S [16]	49.6	5.75	88.45	81.26	79.46	88.72	72.7
SWIN S + MGiT	72.5	8.03	90.57	85.78	82.2	89.69	74.27
			↑ 2.12	↑ 4.52	↑ 2.74	↑ 0.97	↑ 1.57
T2T							
T2T-14 [30]	4.9	9.6	98.37	87.33	88.53	95.03	83.69
T2T-14 + MGiT	7.3	17.3	98.98	89.24	88.57	96.42	84.21
			↑ 0.61	↑ 1.91	↑ 0.04	↑ 1.39	↑ 0.52

Table 4. Comparison of performance between MGiT and other plugin methods DeiT and LDrloc when trained from scratch

Model	C-10	C-100	Fr-102
ViT-B/32 + Ldrloc	85.98	61.22	45.57
ViT-B/32 + DeiT	85.44	61.65	42.81
ViT-B/32 + SL	84.81	61.70	40.44
ViT-B/32 + SSL	83.10	60.74	41.52
ViT-B/32 + MGiT	**86.42**	**61.88**	**50.7**
Swin-T + Ldrloc	87.51	67.23	47.37
Swin-T + DeiT	87.65	68.15	45.21
Swin-T + SL	83.57	67.02	47.09
Swin-T + SSL	85.95	67.11	46.82
Swin-T + MGiT	**87.89**	**68.73**	**55.14**
T2T-ViT-14 + Ldrloc	85.56	63.03	57.98
T2T-ViT-14 + DeiT	85.23	62.14	57.11
T2T-ViT-14 + SL	84.13	61.93	56.32
T2T-ViT-14 + SSL	85.22	61.81	57.55
T2T-ViT-14 + MGiT	**86.19**	**63.82**	**59.00**

scenarios where multiple ViT architectures are trained from scratch on datasets of various sizes, utilizing the early stopping technique. The accuracy of these ViT models without employing the proposed MGiT method is first calculated across multiple datasets. It is observed that the accuracy of these models exhibits significant variations depending on the dataset size. Additionally, the architecture of the ViT model plays a crucial role, with ViT/16 models outperforming ViT/32 models by (5%-10%) accuracy. A similar trend is noted with SWIN-S and SWIN-T models, showing variations of (20%-40%) on larger datasets but similar accuracy on IN datasets (1%-3% variation). CVT stands out by outperforming all other ViT models across small and medium datasets. This is attributed to CVT's use of convolutional layers for downsampling and depth-wise convolution in the attention mechanism. After having been trained using the proposed MGiT approach on multiple datasets respectively, the performance metrics of these ViT models were recalculated and presented in Table 2, accordingly. MGiT demonstrates improved accuracy across most ViT models trained from scratch, except CVT-13, with the ex-

Table 5. Comparison of performance between MGit and other plugin methods DeiT and LDrloc when trained using pre-trained IN weights. (*) : Indicate we did not conduct the experiments ourselves and use the reported accuracy

Model	C-10	C-100	Fr-102
ViT-B/32 + Ldrloc	98.19	88.23	91.30
ViT-B/32 + DeiT	**99.10**	89.8	93.40
ViT-B/32 + SL	95.53	86.27	91.84
ViT-B/32 + SSL	96.41	88.17	90.11
ViT-B/32 + DeiT + ES *	98.6	87.00	-
ViT-B/32 + OFDB *	97.2	85.3	**98.3**
ViT-B/32 + MGiT	98.57	**90.03**	93.94
Swin-T + Ldrloc	98.37	89.4	97.21
Swin-T + DeiT	**98.78**	89.55	**97.33**
Swin-T + SL	95.52	87.24	96.52
Swin-T + SSL	94.71	88.42	95.91
Swin-T + MGiT	98.11	**90.14**	97.12
T2T-ViT-14 + Ldrloc	98.52	88.08	96.2
T2T-ViT-14 + DeiT	98.89	88.79	96.39
T2T-ViT-14 + SL	96.33	83.22	93.75
T2T-ViT-14 + SSL	97.91	85.30	94.34
T2T-ViT-14 + DropKey *	97.6	79.2	-
T2T-ViT-14 + MGiT	**98.93**	**89.24**	**96.42**

tent of improvement dependent on the ViT architecture and dataset size. SWIN-S, in particular, exhibits the highest improvement, ranging from (5%-27%), except for the Pet-37 dataset, where the SWIN-S improvement is 0.55%. The lowest improvement is observed with the CVT model, varying from (0.03%-0.63%). Other ViT models, such as ViT and T2T, also show improvements ranging from (1%-10%) when trained with the MGiT method.

4.4. Fine-tuning

In this section, we focus on the analysis of ViT models (excluding CVT due to the absence of pre-trained weights and resource constraints for training CVT on IN) to understand the impact of MGiT when using pre-trained weights from IN as shown in Table 3. Utilizing pre-trained weights from IN generally provides a significant boost in accuracy for all ViT models. Further, it also reduces the variation between different ViT models. Similar to training from scratch, in this scenario, we exclusively employ the MGiT method to update the weights of the classification layer. Without this approach, the classification layer would typically use random weights. It is observed that similar to training from scratch, MGiT improves the accuracy of most models when pre-trained weights are utilized. However, the magnitude of improvement is not as significant compared to training from scratch. The most notable impact is observed in the SWIN-S ViT model, with improvements ranging from 1% to 4%. However, it's important to note that, similar to training from scratch, SWIN-S underperforms when trained on small datasets and also using IN pre-trained weights.

4.5. Comparison with Other Methods

In our experimental comparisons, we evaluated MGiT against other plug-and-play methods, specifically DeiT [27] SL [13], SSL [10], Drloc [15], ES [5], OFDB [19], and DropKey [14]. DeiT employs knowledge distillation to allow ViT to leverage CNN knowledge, especially beneficial for large datasets such as IN. On the other hand, Ldrloc uses a self-supervised technique to capture spatial information between patches. Similarly, [15] uses SPA and LSA increasing the locality inductive bias and [10] utilizes a self-supervised method for weight initialization. Our comparisons were conducted on three datasets: CIFAR-10, CIFAR-100, and Flower-102, due to the limited resources. Table 4 and Table 5 showcase the results for training from scratch and transfer learning. Results showed that MGiT outperforms in a train-from-scratch scenario, particularly highlighted in the Flower-102 dataset where MGiT achieved notably higher scores with various models such as ViT-B/32 and Swin-T. However, in transfer learning scenarios using IN weights, MGiT faced challenges, with SWIN-T and ViT-B models, where DeiT's performance, leveraging CNN knowledge, was superior. This indicates MGiT's strength in training from scratch, while DeiT may have advantages in some transfer learning scenarios.

5. Conclusion

We extensively investigate the performance of ViT models, highlighting the distinctive factors that influence their performance compared to CNNs. ViTs performance variability is attributed to factors such as model architecture and dataset size. Moreover, we propose a straightforward yet highly efficient technique called MGiT tailored for training ViTs on smaller datasets, applicable across both initial training and transfer learning scenarios. Our method leverages auxiliary ViT structures with a specific emphasis on refining weight updates within the classification layer. This focused approach aims to narrow the parameter search process, offering a notable enhancement over random weight initialization techniques. Our study demonstrates MGiT's effectiveness in optimizing transformer-based architectures for computer vision. While it has been tested on small to medium datasets, further research is needed to assess its performance on larger datasets and with large CNNs.

Acknowledgments

IADS Essex University and the Economic and Social Research Council (ESRC) provided high-performance computing resources through the Business and Local Government Data Research Centre under Grant ES/S007156/1.

References

[1] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. *arXiv preprint arXiv:2012.09958*, 2020. 2

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. 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 13*, pages 446–461. Springer, 2014. 2

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 5, 6, 7

[5] Shuning Chang, Pichao Wang, Hao Luo, Fan Wang, and Mike Zheng Shou. Revisiting vision transformer from the view of path ensemble. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19889–19899, 2023. 2, 8

[6] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021. 2

[7] Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Patch n'pack: Navit, a vision transformer for any aspect ratio and resolution. *arXiv preprint arXiv:2307.06304*, 2023. 1

[8] 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*, pages 248–255. Ieee, 2009. 1

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 5, 6, 7

[10] Hanan Gani, Muzammal Naseer, and Mohammad Yaqub. How to train vision transformer on small-scale datasets? *arXiv preprint arXiv:2210.07240*, 2022. 1, 8

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1, 2

[13] Seung Hoon Lee, Seunghyun Lee, and Byung Cheol Song. Vision transformer for small-size datasets. *arXiv preprint arXiv:2112.13492*, 2021. 2, 8

[14] Bonan Li, Yinhan Hu, Xuecheng Nie, Congying Han, Xiangjian Jiang, Tiande Guo, and Luoqi Liu. Dropkey for vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22700–22709, 2023. 2, 8

[15] Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco Nadai. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems*, 34:23818–23830, 2021. 1, 2, 8

[16] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 1, 2, 5, 6, 7

[17] Imad Eddine Marouf, Enzo Tartaglione, and Stéphane Lathuilière. Mini but mighty: Finetuning vits with mini adapters. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1732–1741, 2024. 1

[18] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 1

[19] Ryo Nakamura, Hirokatsu Kataoka, Sora Takashima, Edgar Josafat Martinez Noriega, Rio Yokota, and Nakamasa Inoue. Pre-training vision transformers with very limited synthesized images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20360–20369, 2023. 2, 8

[20] 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*, pages 722–729. IEEE, 2008. 2

[21] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 2

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 5

[23] Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. Parallel training of dnns with natural gradient and parameter averaging. *arXiv preprint arXiv:1410.7455*, 2014. 1

[24] Anurag Roy, Vinay K Verma, Sravan Voonna, Kripabandhu Ghosh, Saptarshi Ghosh, and Abir Das. Exemplar-free continual transformer with convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5897–5907, 2023. 2

[25] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7262–7272, 2021. 2

[26] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International*

conference on machine learning, pages 6105–6114. PMLR, 2019. 1, 2

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 1, 2, 8

[28] Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *European Conference on Computer Vision*, pages 516–533. Springer, 2022. 5

[29] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22–31, 2021. 1, 2, 5, 6

[30] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 558–567, 2021. 2, 5, 6, 7