Enhanced Normalization in Vision Transformers: The Dual PatchNorm Approach

Abstract

This study introduces Dual PatchNorm, a modification for Vision Transformers that incorporates two Layer Normalization layers (LayerNorms) positioned before and after the patch embedding layer. The effectiveness of Dual PatchNorm is demonstrated through its superior performance compared to alternative LayerNorm placement strategies within the Transformer block, as determined through extensive testing. Experimental results across various tasks, including image classification, contrastive learning, semantic segmentation, and transfer learning on downstream classification datasets, consistently show that this simple adjustment leads to improved accuracy over well-optimized standard Vision Transformers, without any negative impact.

1 Introduction

Layer Normalization is essential for the successful and stable training of Transformer models, enabling high performance across diverse tasks. This normalization technique is equally vital in Vision Transformers (ViTs), which largely adhere to the standard architecture of the original Transformer model.

This research investigates whether a different arrangement of LayerNorms can enhance ViT models. Initially, we evaluate five ViT architectures on ImageNet-1k and find that an exhaustive search for optimal LayerNorm placements within the Transformer block's components does not yield improvements in classification accuracy. This suggests that the pre-LN approach in ViTs is nearly optimal. Further investigation reveals that alternative LayerNorm placements, such as NormFormer and Sub-LN, also do not surpass the performance of robust ViT classification models when used independently.

A significant finding of this study is the observation that the addition of LayerNorms before and after the standard ViT-projection layer, termed Dual PatchNorm (DPN), can substantially improve performance over well-tuned baseline ViTs. Experiments conducted on image classification across three datasets with varying sample sizes, as well as contrastive learning, confirm the effectiveness of DPN. Notably, qualitative analysis indicates that the LayerNorm scale parameters assign greater weight to pixels located at the center and corners of each patch.

2 Related Work

Prior research has explored modifications to the patch-embedding layer in ViTs. For instance, one study demonstrated that adding a LayerNorm after patch-embedding enhances ViT's resilience to image corruptions on smaller datasets. Another study replaced the standard Transformer stem with a series of stacked stride-two 3x3 convolutions with batch normalizations, resulting in improved sensitivity to optimization hyperparameters and increased final accuracy.

Further analysis of LayerNorm has shown that the derivatives of the mean and variance significantly contribute to performance, as opposed to forward normalization. Alternative strategies like Image-LN and Patch-LN have been considered for efficiently training a single model across different patch sizes. Some researchers have added extra LayerNorms before the final dense projection in the self-attention block and the non-linearity in the MLP block, employing a different initialization strategy. Others have proposed adding LayerNorms after the final dense projection in the self-attention block, along with a LayerNorm after the non-linearity in the MLP block.

In contrast to previous studies, our work demonstrates that applying LayerNorms both before and after the embedding layer consistently enhances performance in classification and contrastive learning tasks. While other research has focused on incorporating convolutional inductive biases into Vision Transformers, our study exclusively and thoroughly examines LayerNorm placements within the standard ViT architecture.

3 Methodology

3.1 Patch Embedding Layer in Vision Transformer

Vision Transformers consist of a patch embedding layer (PE) followed by multiple Transformer blocks. The PE layer first transforms an image $x \in \mathbb{R}^{H \times W \times 3}$ into a sequence of patches $x_p \in \mathbb{R}^{P^2 \times P^2 HW}$, where P is the patch size. Each patch is then independently projected using a dense layer, creating a sequence of "visual tokens" $x_t \in \mathbb{R}^{HWP^2 \times D}$. The patch size P determines the trade-off between the granularity of the visual tokens and the computational demands of subsequent Transformer layers.

3.2 Layer Normalization

When applied to a sequence of N patches $x \in \mathbb{R}^{N \times D}$, LayerNorm in ViTs involves two steps:

$$x = \frac{x - \mu(x)}{\sigma(x)} \tag{1}$$

$$y = \gamma x + \beta \tag{2}$$

where $\mu(x) \in \mathbb{R}^N$, $\sigma(x) \in \mathbb{R}^N$, $\gamma \in \mathbb{R}^D$, and $\beta \in \mathbb{R}^D$.

First, Equation 3.1 normalizes each patch $x_i \in \mathbb{R}^D$ in the sequence to have zero mean and unit standard deviation. Then, Equation 3.2 applies learnable shifts and scales β and γ, which are shared across all patches.

3.3 Alternate LayerNorm placements:

Following established practices, ViTs typically place LayerNorms before each self-attention and MLP layer, known as the pre-LN strategy. We assess three different strategies for each self-attention and MLP layer: placing LayerNorm before (pre-LN), after (post-LN), and both before and after (pre+post-LN). This results in nine distinct combinations.

3.4 Dual PatchNorm

Instead of adding LayerNorms within the Transformer block, we propose applying them to the stem alone, both before and after the patch embedding layer. Specifically, we replace:

$$x = PE(x) \tag{3}$$

with

$$x = LN(PE(LN(x))) \tag{4}$$

while keeping the rest of the architecture unchanged. We refer to this approach as Dual PatchNorm (DPN).

4 Experiments

4.1 Setup

We utilize the standard Vision Transformer formulation, which has demonstrated broad applicability across various vision tasks. We train ViT architectures, both with and without DPN, in a supervised manner on three datasets with varying numbers of examples: ImageNet-1k (1M), ImageNet-21k (21M), and JFT (4B). In our experiments, we apply DPN directly to the baseline ViT recipes without any additional hyperparameter tuning. We divide the ImageNet training set into training and validation subsets and use the validation set to finalize the DPN recipe.

For ImageNet-1k, we train five architectures: Ti/16, S/16, S/32, B/16, and B/32 using a standard recipe for 93,000 steps with a batch size of 4,096. We report the accuracy on the official ImageNet validation split. Additionally, we evaluate an S/16 baseline (S/16+) with extensive hyperparameter tuning on ImageNet. We also apply DPN to the base and small DeiT variants.

On ImageNet-21k, we use a similar setup as ImageNet-1k and report ImageNet 25-shot accuracies in two training regimes: 93K and 930K steps.

For JFT, we evaluate the ImageNet 25-shot accuracies of three variants (B/32, B/16, and L/16) in two training regimes (220K and 1.1M steps) with a batch size of 4,096, without additional data augmentation or mixup regularization.

We report the 95% confidence interval across at least three independent runs on ImageNet-1k. Due to the computational cost of training on ImageNet-21k and JFT, we train each model once and report the mean 25-shot accuracy with a 95% confidence interval across three random seeds.

4.2 DPN versus alternate LayerNorm placements

Each Transformer block in ViT includes a self-attention (SA) and an MLP layer. Following the pre-LN strategy, LN is placed before both the SA and MLP layers. We first demonstrate that the default pre-LN strategy in ViT models is nearly optimal by evaluating alternative LN placements on ImageNet-1k. We then compare this with the performance of NormFormer, Sub-LN, and DPN.

For each SA and MLP layer, we evaluate three LN placements: Pre, Post, and Pre+Post, resulting in nine total LN placement configurations. Additionally, we assess the LayerNorm placements in NormFormer and Sub LayerNorm, which add extra LayerNorms within the self-attention and MLP layers in the transformer block. Figure 1 shows that none of these placements significantly outperform the default Pre-LN strategy, indicating that the default strategy is close to optimal. NormFormer provides some improvements on ViT models with a patch size of 32. However, DPN consistently enhances performance across all five architectures.

Figure 1: This plot illustrates the accuracy gains achieved by various LayerNorm placement strategies over the default pre-LN strategy. Each blue point represents a different LN placement within the Transformer block. None of the alternative placements surpass the default Pre-LN strategy on ImageNet-1k. The application of DPN (represented by the black cross) consistently improves performance across all five architectures.

4.3 Comparison to ViT

Table 1 (left) shows that DPN improved the accuracy of B/16, the best ViT model, by 0.7, while S/32 achieved the maximum accuracy gain of 1.9. The average gain across all architectures is 1.4. On top of DeiT-S and DeiT-B, DPN provides improvements of 0.3 and 0.2, respectively. Furthermore, we fine-tune B/16 and B/32 models with and without DPN on high-resolution ImageNet (384x384) for 5,000 steps with a batch size of 512. Applying DPN improves the high-resolution, fine-tuned B/16 and B/32 by 0.6 and 1.0, respectively.

DPN enhances all architectures trained on ImageNet-21k (Table 1, right) and JFT (Table 2) in shorter training regimes, with average gains of 1.7 and 0.8, respectively. In longer training regimes, DPN improves the accuracy of the best-performing architectures on JFT and ImageNet-21k by 0.5 and 0.4, respectively.

In three cases (Ti/16 and S/32 with ImageNet-21k, and B/16 with JFT), DPN matches or slightly underperforms compared to the baseline. Nevertheless, across a large proportion of ViT models, simply applying DPN out-of-the-box on top of well-tuned ViT baselines leads to significant improvements.

Table 1: Left: ImageNet-1k validation accuracies of five ViT architectures with and without Dual PatchNorm after 93,000 steps. Right: Training ViT models on ImageNet-21k in two regimes (93k and 930k steps) with a batch size of 4,096, showing ImageNet 25-shot accuracies with and without Dual PatchNorm.

ViT AugReg			ImageNet-21k		
Arch	Base	DPN	Arch	Base	DPN
S/32	72.1 ± 0.07	**74.0 ± 0.09**	93K Steps		
Ti/16	72.5 ± 0.07	**73.9 ± 0.09**	Ti/16	52.2 ± 0.07	**53.6 ± 0.07**
B/32	74.8 ± 0.06	**76.2 ± 0.07**	S/32	54.1 ± 0.03	**56.7 ± 0.03**
S/16	78.6 ± 0.32	**79.7 ± 0.2**	B/32	60.9 ± 0.03	**63.7 ± 0.03**
S/16+	79.7 ± 0.09	**80.2 ± 0.03**	S/16	64.3 ± 0.15	**65.0 ± 0.06**
B/16	80.4 ± 0.06	**81.1 ± 0.09**	B/16	70.8 ± 0.09	**72.0 ± 0.03**
DeiT			930K Steps		
S/16	80.1 ± 0.03	**80.4 ± 0.06**	Ti/16	61.0 ± 0.03	61.2 ± 0.03
B/16	81.8 ± 0.03	**82.0 ± 0.05**	S/32	63.8 ± 0.00	**65.1 ± 0.12**
AugReg + 384x384 Finetune			B/32	72.8 ± 0.03	**73.1 ± 0.07**
B/32	79.0 ± 0.00	**80.0 ± 0.03**	S/16	72.5 ± 0.1	72.5 ± 0.1
B/16	82.2 ± 0.03	**82.8 ± 0.00**	B/16	78.0 ± 0.06	**78.4 ± 0.03**

4.4 Finetuning on ImageNet with DPN

We fine-tune four models trained on JFT-4B with two resolutions on ImageNet-1k: (B/32, B/16) × (220K, 1.1M) steps at resolutions 224x224 and 384x384. For B/32, we observe consistent improvement across all configurations. With L/16, DPN outperforms the baseline in three out of four configurations.

Table 2: Left: Training three ViT models on JFT-4B in two regimes (200K and 1.1M steps) with a batch size of 4,096, showing ImageNet 25-shot accuracies with and without DPN. Right: Corresponding full fine-tuning results on ImageNet-1k.

JFT-4B			ImageNet-1k Finetuning				
Arch	Base	DPN	Arch	Resolution	Steps	Base	DPN
220K steps			B/32	224	220K	77.6 ± 0.06	**78.3 ± 0.00**
B/32	63.8 ± 0.03	**65.2 ± 0.03**	B/32	384	220K	81.3 ± 0.09	**81.6 ± 0.00**
B/16	72.1 ± 0.09	**72.4 ± 0.07**	B/32	224	1.1M	80.8 ± 0.1	**81.3 ± 0.00**
L/16	77.3 ± 0.00	**77.9 ± 0.06**	B/32	384	1.1M	83.8 ± 0.03	**84.1 ± 0.00**
1.1M steps			L/16	224	220K	84.9 ± 0.06	**85.3 ± 0.03**
B/32	70.7 ± 0.1	71.1 ± 0.09	L/16	384	220K	86.7 ± 0.03	**87.0 ± 0.00**
B/16	76.9 ± 0.03	76.6 ± 0.03	L/16	224	1.1M	86.7 ± 0.03	**87.1 ± 0.00**
L/16	80.9 ± 0.03	**81.4 ± 0.06**	L/16	384	1.1M	88.2 ± 0.00	**88.3 ± 0.06**

5 Experiments on Downstream Tasks

5.1 Finetuning on VTAB

We fine-tune ImageNet-pretrained B/16 and B/32 models, both with and without DPN, on the Visual Task Adaptation Benchmark (VTAB), which consists of 19 datasets categorized as Natural, Specialized, and Structured. Natural datasets contain images captured with standard cameras, Specialized datasets have images from specialized equipment, and Structured datasets require scene comprehension. We use the VTAB training protocol, which defines a standard training split of 800 examples and a validation split of 200 examples per dataset. We perform a lightweight sweep across three learning rates for each dataset and select the best model based on the mean validation accuracy across three seeds. The corresponding mean test scores across three seeds are reported in Table 3.

On Natural datasets, which are most similar to the source dataset ImageNet, B/32 and B/16 with DPN significantly outperform the baseline in 7 out of 7 and 6 out of 7 datasets, respectively. The only exception is Sun397, where DPN performs worse. However, additional experiments show that DPN is beneficial when B/16 is trained from scratch on Sun397. On Structured datasets, applying DPN improves accuracy in 4 out of 8 datasets and remains neutral in 2 for both B/16 and B/32. On Specialized datasets, DPN improves performance in 1 out of 4 datasets and is neutral in 2. In conclusion, DPN offers the most significant improvements when fine-tuned on Natural datasets. For Structured and Specialized datasets, DPN serves as a lightweight alternative that can enhance or at least not harm performance in most cases.

Table 3: Evaluation of DPN on VTAB. When fine-tuned on Natural datasets, B/32 and B/16 with DPN significantly outperform the baseline in 7 out of 7 and 6 out of 7 datasets, respectively. On Structured datasets, DPN improves both B/16 and B/32 in 4 out of 8 datasets and remains neutral in 2. On Specialized datasets, DPN improves performance in 1 out of 4 datasets and is neutral in 2.

	Natural							Specialized		
	Caltech101	CIFAR-100	DTD	Flowers102	Pets	Sun397	SVHN	Camelyon	EuroSAT	Re
B/32	87.1	53.7	56.0	83.9	87.2	32.0	76.8	77.9	94.8	
+ DPN	**87.7**	**58.1**	**60.7**	**86.4**	**88.0**	35.4	**80.3**	**78.5**	**95.0**	
B/16	86.1	35.5	60.1	90.8	90.9	33.9	76.7	81.3	95.9	
+ DPN	**86.6**	**51.4**	**63.1**	**91.3**	**92.1**	32.5	**78.3**	80.6	**95.8**	

	Structured							
	Clevr-Count	Clevr-Dist	DMLab	dSpr-Loc	dSpr-Ori	KITTI-Dist	sSNORB-Azim	sNORB-Elev
B/32	58.3	52.6	39.2	71.3	59.8	73.6	20.7	47.2
+ DPN	**62.5**	**55.5**	40.7	60.8	**61.6**	73.4	**20.9**	34.4
B/16	65.2	59.8	39.7	72.1	61.9	**81.3**	18.9	50.4
+ DPN	**73.7**	48.3	**41.0**	**72.4**	**63.0**	80.6	**21.6**	36.2

5.2 Contrastive Learning

We apply DPN to image-text contrastive learning. Each minibatch consists of image and text pairs. We train a text and image encoder to map an image to its correct text over all other texts in the minibatch. Specifically, we adopt a method where we initialize and freeze the image encoder from a pretrained checkpoint and train the text encoder from scratch. To evaluate zero-shot ImageNet accuracy, we represent each ImageNet class by its text label, which the text encoder maps into a class embedding. For a given image embedding, the prediction is the class corresponding to the nearest class embedding.

We evaluate four frozen image encoders: two architectures (B/32 and L/16) trained with two schedules (220K and 1.1M steps). We reuse standard hyperparameters and train only the text encoder using a contrastive loss for 55,000 steps with a batch size of 16,384. Table 4 shows that on B/32, DPN improves over the baselines in both setups, while on L/16, DPN provides improvement when the image encoder is trained with shorter training schedules.

Table 4: Zero-Shot ImageNet accuracy in the contrastive learning setup.

Arch	Steps	Base	DPN
B/32	220K	61.9 ± 0.12	**63.0 ± 0.09**
B/32	1.1M	67.4 ± 0.07	**68.0 ± 0.09**
L/16	220K	75.0 ± 0.11	**75.4 ± 0.00**
L/16	1.1M	78.7 ± 0.05	78.7 ± 0.1

5.3 Semantic Segmentation

We fine-tune ImageNet-pretrained B/16 models, with and without DPN, on the ADE-20K 512x512 semantic segmentation task. Following established methods, a single dense layer maps the ViT features into per-patch output logits. A bilinear upsampling layer then transforms the output distribution into the final high-resolution 512x512 semantic segmentation output. We fine-tune the entire ViT backbone with a standard per-pixel cross-entropy loss. Table 5 reports the mean mIOU across 10 random seeds and different fractions of training data. The improvement in IoU is consistent across all setups.

Table 5: Fine-tuning ImageNet pretrained B/16 models with and without DPN on the ADE20K Semantic Segmentation task, with varying fractions of ADE20K training data. The table reports the mean IoU across ten random seeds. Applying DPN improves IoU across all settings.

Fraction of Train Data	1/16	1/8	1/4	1/2	1
B/16	27.3 ± 0.09	32.6 ± 0.09	36.9 ± 0.13	40.8 ± 0.1	45.6 ± 0.08
+DPN	**28.0 ± 0.21**	**33.7 ± 0.11**	**38.0 ± 0.11**	**41.9 ± 0.09**	**46.1 ± 0.11**

6 Ablations

Is normalizing both the inputs and outputs of the embedding layer optimal? In Eq 4, DPN applies LN to both the inputs and outputs of the embedding layer. We evaluate three alternative strategies: Pre, Post, and Post PosEmb. Pre applies LayerNorm only to the inputs, Post applies it only to the outputs, and Post PosEmb applies it to the outputs after they are summed with positional embeddings.

Table 6 shows the accuracy gains with these alternative strategies. Pre is unstable on B/32, leading to a significant drop in accuracy, and it also results in minor accuracy drops on S/32 and Ti/16. Post and Post PosEmb perform worse on smaller models (B/32, S/32, and Ti/16). Our experiments demonstrate that applying LayerNorm to both inputs and outputs of the embedding layer is necessary for consistent accuracy improvements across all ViT variants.

Table 6: Ablations of various components of DPN. Pre: LayerNorm only to the inputs of the embedding layer. Post: LayerNorm only to the outputs of the embedding layer. No learnable: Per-patch normalization without learnable LayerNorm parameters. Only learnable: Learnable scales and shifts without standardization.

	B/16	S/16	B/32	S/32	Ti/16
Pre	-0.1	0.0	-2.6	-0.2	-0.3
Post	0.0	-0.2	-0.5	-0.7	-1.1
Post PosEmb	0.0	-0.1	-0.4	-0.9	-1.1
Only learnable	-0.8	-0.9	-1.2	-1.6	-1.6
RMSNorm	0.0	-0.1	-0.4	-0.5	-1.7
No learnable	-0.5	0.0	-0.2	-0.1	-0.1

Normalization vs. Learnable Parameters: As seen in Sec. 3.2, LayerNorm involves a normalization operation followed by learnable scales and shifts. We also ablate the effect of each of these operations in DPN.

Applying only learnable scales and shifts without normalization significantly decreases accuracy across all architectures (See: Only learnable in Table 6). Additionally, removing the learnable parameters leads to unstable training on B/16 (No learnable in Table 6). Finally, removing the centering and bias parameters, as done in RMSNorm, reduces the accuracy of B/32, S/32, and Ti/16. We conclude that while both normalization and learnable parameters contribute to the success of DPN, normalization has a greater impact.

7 Analysis

7.1 Gradient Norm Scale

We present per-layer gradient norms for B/16, both with and without DPN. Figure 2 (Left) displays the mean gradient norm of the last 1000 training steps as a function of depth. Notably, the gradient norm of the base ViT patch embedding (black) is disproportionately large compared to other layers. Applying DPN (red) scales down the gradient norm of the embedding layer. Figure 2 (Right) further shows that the gradient norm of the embedding layer is reduced not only before convergence but also throughout the training process. This characteristic is consistent across ViT architectures of different sizes.

7.2 Visualizing Scale Parameters

The first LayerNorm in Eq. 4 is applied directly to patches, i.e., raw pixels. Thus, the learnable parameters (biases and scales) of the first LayerNorm can be visualized directly in pixel space. Figure 3 shows the scales of our smallest and largest models: Ti/16 trained on ImageNet for 90,000 steps and L/16 trained on JFT for 1.1M steps, respectively. Since the absolute magnitude of the scale parameters varies across the R, G, and B channels, we visualize the scale separately for each channel. Interestingly, for both models, the scale parameter increases the weight of the pixels in the center of the patch and at the corners.

8 Conclusion

We propose a straightforward modification to standard ViT models