DUAL PATCHNORM

Manoj Kumar, Mostafa Dehghani, Neil Houlsby
Google Research, Brain Team
{mechcoder,dehghani,neilhoulsby}@google.com

ABSTRACT

We propose Dual PatchNorm: two Layer Normalization layers (LayerNorms), before and after the patch embedding layer in Vision Transformers. We demonstrate that Dual PatchNorm outperforms the result of exhaustive search for alternative LayerNorm placement strategies in the Transformer block itself. In our experiments, incorporating this trivial modification, often leads to improved accuracy over well-tuned Vision Transformers and never hurts.

1 INTRODUCTION

Layer Normalization (Ba et al., 2016) is key to Transformer's success in achieving both stable training and high performance across a range of tasks. Such normalization is also crucial in Vision Transformers (ViT) (Dosovitskiy et al., 2020) which closely follow the standard recipe of the original Transformer model.

Following the "pre-LN" strategy in Baevski & Auli (2019) and Xiong et al. (2020), ViTs place LayerNorms before the self-attention layer and MLP layer in each Transformer block. We explore the following question: Can we improve ViT models with a different LayerNorm ordering? First, across five ViT architectures on ImageNet-1k (Russakovsky et al., 2015), we demonstrate that an exhaustive search of LayerNorm placements between the components of a Transformer block does not improve classification accuracy. This indicates that the pre-LN strategy in ViT is close to optimal. Our observation also applies to other alternate LayerNorm placements: NormFormer (Shleifer et al., 2021) and Sub-LN (Wang et al., 2022), which in isolation, do not improve over strong ViT classification models.

Second, we make an intriguing observation: placing additional LayerNorms before and after the ViT-projection layer, which we call Dual PatchNorm (DPN), improves over well tuned ViT baselines. Experiments across three different datasets with varying number of examples, demonstrate the efficacy of DPN. Interestingly, our qualitative experiments show that the LayerNorm scale parameters upweight the pixels at the center and corners of each patch.

```
hp, wp = patch_size[0], patch_size[1]
x = einops.rearrange(
    x, "b (ht hp) (wt wp) c -> b (ht wt) (hp wp c)", hp=hp, wp=wp)
x = nn.LayerNorm(name="ln0")(x)
x = nn.Dense(output_features, name="dense")(x)
x = nn.LayerNorm(name="ln1")(x)
```

Dual PatchNorm consists of a 2 line change to the standard ViT-projection layer.

2 RELATED WORK

Kim et al. (2021) add a LayerNorm after the patch-embedding and show that this improves the robustness of ViT against corruptions on small-scale datasets. Xiao et al. (2021) replace the standard Transformer stem with a small number of stacked stride-two 3×3 convolutions with batch normalizations and show that this improves the sensitivity to optimization hyperparameters and final accuracy. Xu et al. (2019) analyze LayerNorm and show that the derivatives of mean and variance have a greater contribution to final performance as opposed to forward normalization. Beyer et al.

(2022a) consider Image-LN and Patch-LN as alternative strategies to efficiently train a single model for different patch sizes. Wang et al. (2022) add extra LayerNorms before the final dense projection in the self-attention block and the non-linearity in the MLP block, with a different initialization strategy. Shleifer et al. (2021) propose extra LayerNorms after the final dense projection in the self-attention block instead with a LayerNorm after the non-linearity in the MLP block.

3 Background

3.1 Patch Embedding Layer in Vision Transformer

Vision Transformers (Dosovitskiy et al., 2020) consist of a patch embedding layer (PE) followed by a stack of Transformer blocks. The PE layer first rearranges the image $x \in \mathcal{R}^{H \times W \times 3}$ into a sequence of patches $x_p \in \mathcal{R}^{\frac{HW}{P^2} \times P^2}$ where P denotes the patch size. It then projects each patch independently with a dense projection to constitute a sequence of "visual tokens" $\mathbf{x_t} \in \mathcal{R}^{\frac{HW}{P^2} \times D}$ P controls the trade-off between granularity of the visual tokens and the computational cost in the subsequent Transformer layers.

3.2 Layer Normalization

Given a sequence of N patches $\mathbf{x} \in \mathcal{R}^{N \times D}$, LayerNorm as applied in ViTs consist of two operations:

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

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

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

First, Eq. 1 normalizes each patch $\mathbf{x_i} \in \mathcal{R}^D$ of the sequence to have zero mean and unit standard deviation. Then, Eq 2 applies learnable shifts and scales β and γ which are shared across all patches.

4 Methods

4.1 Alternate LayerNorm placements:

Following Baevski & Auli (2019) and Xiong et al. (2020), ViTs incorporate LayerNorm before every self-attention and MLP layer, commonly known as the pre-LN strategy. For each of the self-attention and MLP layer, we evaluate 3 strategies: place LayerNorm before (pre-LN), after (post-LN), before and after (pre+post-LN) leading to nine different combinations.

4.2 Dual PatchNorm

Instead of adding LayerNorms to the Transformer block, we also propose to apply LayerNorms in the stem alone, both before and after the patch embedding layer. In particular, we replace

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

with

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

and keep the rest of the architecture fixed. We call this Dual PatchNorm (DPN).

5 Experiments

5.1 Setup

We train ViT architectures (with and without DPN) in a supervised fashion on 3 different datasets with varying number of examples: ImageNet-1k (1M), ImageNet-21k (21M) and JFT (4B) (Zhai et al., 2022a). In our experiments, we apply DPN directly on top of the baseline ViT recipes without additional hyperparamter tuning. We split the ImageNet train set into a train and validation split, and use the validation split to arrive at the final DPN recipe.

ImageNet 1k: We train 5 architectures: Ti/16, S/16, S/32, B/16 and B/32 using the AugReg (Steiner et al., 2022) recipe for 93000 steps with a batch size of 4096 and report the accuracy on the official ImageNet validation split as is standard practice. We additionally evaluate a S/16 baseline (S/16+) with more optimal hyperparameters on ImageNet (Beyer et al., 2022b).

ImageNet 21k: We adopt a similar setup as in ImageNet 1k. We report ImageNet 25 shot accuracies in two training regimes: 93K and 930K steps.

JFT: We evaluate the ImageNet 25 shot accuracies of 3 variants (B/32, B/16 and L/16) on 2 training regimes: (220K and 1.1M steps) with a batch size of 4096. In this setup, we do not use any additional data augmentation or mixup regularization.

On ImageNet-1k, we report the 95% confidence interval across 3 independent runs. On ImageNet-21k and JFT, because of expensive training runs, we train each model once and report the mean 25 shot accuracy with 95% confidence interval across 3 random seeds.

5.2 DPN versus alternate LayerNorm placements

Each Transformer block in ViT consists of a self-attention (SA) and MLP layer. Following the pre-LN strategy (Xiong et al., 2020), LN is inserted before both the SA and MLP layers. We first show that the default pre-LN strategy in ViT models is close to optimal by evaluating alternate LN placements on ImageNet-1k. We then contrast this with the performance of NormFormer, Sub-LN and DPN.

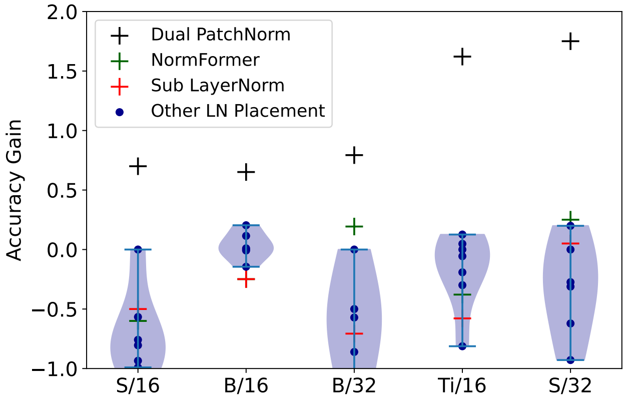


Figure 1: The plot displays the accuracy gains of different LayerNorm placement strategies over the default pre-LN strategy. Each blue point (**Other LN placement**) corresponds to a different LN placement in the Transformer block. None of the placements outperform the default Pre-LN strategy on ImageNet-1k (Russakovsky et al., 2015). Applying DPN (black cross) provides consistent improvements across all 5 architectures.

For each SA and MLP layer, we evaluate three LN placements: Pre, Post and Pre+Post, that leads to nine total LN placement configurations. Additionally, we evaluate the LayerNorm placements in NormFormer (Shleifer et al., 2021) and Sub LayerNorm (Wang et al., 2022) which add additional LayerNorms within each of the self-attention and MLP layers in the transformer block. Figure 1 shows that none of the placements outperform the default Pre-LN strategy significantly, indicating that the default pre-LN strategy is close to optimal. NormFormer provides some improvements on ViT models with a patch size of 32. DPN on the other-hand provides consistent improvements across all 5 architectures.

5.3 Comparison to ViT

Arch	Base	DPN
S/32	72.1 ± 0.07	**74.0** ± 0.09
Ti/16	72.5 ± 0.07	**73.9** ± 0.09
B/32	74.8 ± 0.06	**76.2** ± 0.07
S/16	78.6 ± 0.32	**79.7** ± 0.2
S/16+	79.7 ± 0.09	**80.2** ± 0.03
B/16	80.4 ± 0.06	**81.1** ± 0.09

Arch	Base	DPN
93K Steps		
Ti/16	52.2 ± 0.07	**53.6** ± 0.07
S/32	54.1 ± 0.03	**56.7** ± 0.03
B/32	60.9 ± 0.03	**63.7** ± 0.03
S/16	64.3 ± 0.15	**65.0** ± 0.06
B/16	70.8 ± 0.09	**72.0** ± 0.03
930K Steps		
Ti/16	**61.0** ± 0.03	**61.2** ± 0.03
S/32	63.8 ± 0.00	**65.1** ± 0.12
B/32	72.8 ± 0.03	**73.1** ± 0.07
S/16	**72.5** ± 0.1	**72.5** ± 0.1
B/16	78.0 ± 0.06	**78.4** ± 0.03

Table 1: **Left:** ImageNet-1k validation accuracies of five ViT architectures with and without dual patch norm after 93000 steps. **Right:** We train ViT models on ImageNet-21k in two training regimes: 93k and 930k steps with a batch size of 4096. The table shows their ImageNet 25 shot accuracies with and without Dual PatchNorm

In Table 1 left, DPN improved the accuracy of B/16, the best ViT model by 0.7 while S/32 obtains the maximum accuracy gain of 1.9. The average gain across all architecture is 1.4.

DPN improves all architectures trained on ImageNet-21k (Table 1 Right) and JFT (Table 2) on shorter training regimes with average gains of 1.7 and 0.8 respectively. On 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 leads to marginally worse results than the baseline. Nevertheless, across a large fraction of ViT models, simply employing DPN out-of-the-box on top of well-tuned ViT baselines lead to significant improvements.

5.4 Finetuning with DPN

We finetune four models trained on JFT-4B with two resolutions on ImageNet-1k: (B/32, B/16) × (220K, 1.1M) steps on resolutions 224×224 and 384×384. On B/32, we observe a consistent improvement across all configurations. On L/16, the baselines without DPN match the transfer performance with DPN.

5.5 Contrastive Learning

We adopt the LiT contrastive-learning setup (Zhai et al., 2022b) and evaluate models trained with DPN on zero-shot ImageNet accuracy. We evalute 4 frozen image encoders: 2 architectures (B/32 and L/16) trained with 2 schedules (220K and 1.1M steps). We resue standard hyperparameters and train only the text encoder using a contrastive loss for 55000 steps with a batch-size of 16384. Table 3 shows that on B/32, DPN improves over the baselines on both the setups while on L/16 DPN provides improvement when the image encoder is trained with shorter training schedules.

Arch	Base	DPN
220K steps		
B/32	63.8 ± 0.03	**65.2** ± 0.03
B/16	72.1 ± 0.09	**72.4** ± 0.07
L/16	77.3 ± 0.00	**77.9** ± 0.06
1.1M steps		
B/32	70.7 ± 0.1	**71.1** ± 0.09
B/16	**76.9** ± 0.03	76.6 ± 0.03
L/16	80.9 ± 0.03	**81.4** ± 0.06

Arch	Resolution	Steps	Base	DPN
B/32	224	220K	77.6 ± 0.06	**78.3** ± 0.00
B/32	384	220K	81.3 ± 0.09	**81.6** ± 0.00
B/32	224	1.1M	80.8 ± 0.1	**81.3** ± 0.00
B/32	384	1.1M	83.8 ± 0.03	**84.1** ± 0.00
L/16	224	220K	**84.6** ± 0.06	**84.5** ± 0.13
L/16	384	220K	**86.4** ± 0.00	**86.4** ± 0.03
L/16	224	1.1M	**86.6** ± 0.03	**86.5** ± 0.13
L/16	384	1.1M	**87.8** ± 0.08	**87.9** ± 0.01

Table 2: **Left:** We train 3 ViT models on JFT-4B in two training regimes: 200K and 1.1M steps with a batch size of 4096. The table displays their ImageNet 25 shot accuracies with and without DPN. **Right:** Corresponding full finetuneing results on ImageNet-1k.

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

Table 3: Zero Shot ImageNet accuracy on the LiT (Zhai et al., 2022b) contrastive learning setup.

5.6 Ablations and Analysis

Is normalizing both the inputs and outputs optimal? In Eq 4, DPN applies LN to both the inputs and outputs to the embedding layer. We assess two alternate strategies: **Pre** and **Post**. **Pre** applies LayerNorm only to the inputs while **Post** only to the outputs.

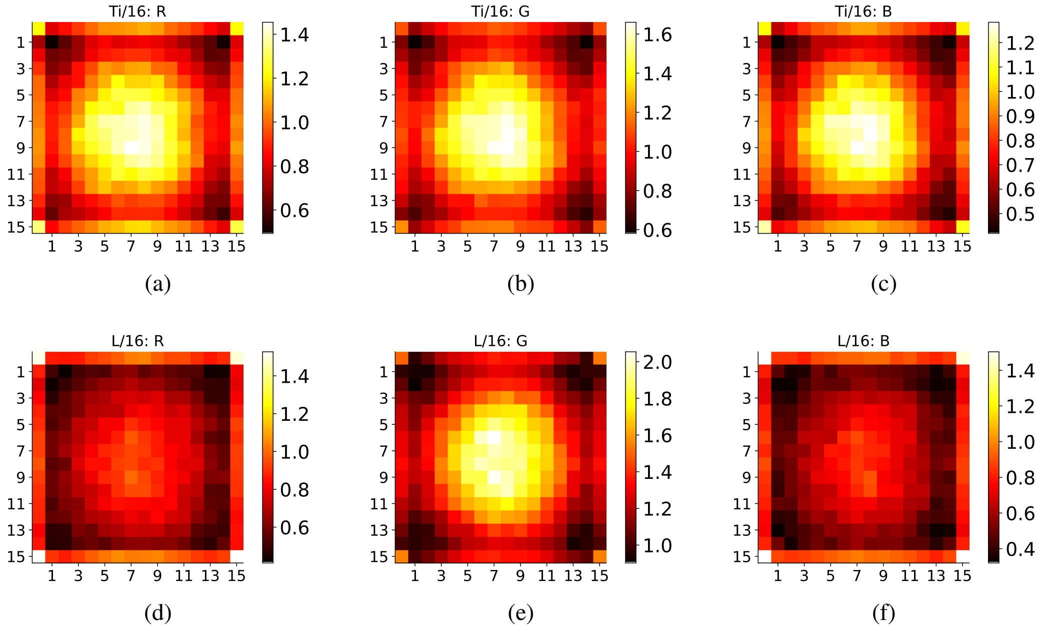
Table 4 displays the accuracy gains with two alternate strategies: **Pre** is unstable on B/32 leading to a significant drop in accuracy. Additionally, **Pre** obtains minor drops in accuracy on S/32 and Ti/16. **Post** achieves worse performance on smaller models B/32, S/32 and Ti/16. Our experiments show that applying LayerNorm to both inputs and outputs of the embedding layer is necessary to obtain consistent improvements in accuracy.

	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
No learnable	-0.5	0.0	-0.2	-0.1	-0.1
Only learnable	-0.8	-0.9	-1.2	-1.6	-1.6

Table 4: Ablations of various components of Patch Layer Normalization. **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.

Normalization vs Learnable Parameters: As seen in Sec. 3.2, LayerNorm constitutes 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 leads to a significant decrease in accuracy across all architectures. (See: **Only learnable** in Table 4). Additionally, removing the learnable parameters leads to unstable training on B/16 (**No learnable** in Table 4). We conclude that while both normalization and learnable parameters contribute to the success of DPN, normalization has a higher impact.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 2: Visualization of scale parameters of the first LayerNorm. **Top:** Ti/16 trained on ImageNet 1k. **Bottom:** L/16 trained on JFT-4B

5.7 Visualizing Scale Parameters

Note that the first LayerNorm in Eq. 4 is applied directly on patches, that is, to raw pixels. Thus, the learnable parameters (biases and scales) of the first LayerNorm can be visualized directly in pixel space. Fig. 2 shows the scales of our smallest model and largest model which are: Ti/16 trained on ImageNet for 90000 steps and L/16 trained on JFT for 1.1M steps respectively. Since the absolute magnitude of the scale parameters vary across the R, G and B channel, 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.

Acknowledgments

We would like to thank Lucas Beyer for detailed feedback on the draft and help with implementation details. We additionally thank Xiahoua Zhai, Abhishek Kumar, Jonathan Heek, Alexander Kolesnikov and Alexey Gritsenko for fruitful dicussions throughout the project.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. *ICLR*, 2019.

Lucas Beyer, Pavel Izmailov, Alexander Kolesnikov, Mathilde Caron, Simon Kornblith, Xiaohua Zhai, Matthias Minderer, Michael Tschannen, Ibrahim Alabdulmohsin, and Filip Pavetic. Flexivit: One model for all patch sizes. *arXiv preprint arXiv:2212.08013*, 2022a.

Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Better plain vit baselines for imagenet-1k. *arXiv preprint arXiv:2205.01580*, 2022b.

Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Big vision. `https://github.com/google-research/big_vision`, 2022c.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Bum Jun Kim, Hyeyeon Choi, Hyeonah Jang, Dong Gu Lee, Wonseok Jeong, and Sang Woo Kim. Improved robustness of vision transformer via prelayernorm in patch embedding. *arXiv preprint arXiv:2111.08413*, 2021.

Alex Rogozhnikov. Einops: Clear and reliable tensor manipulations with einstein-like notation. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=oapKSVM2bcj`.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Sam Shleifer, Jason Weston, and Myle Ott. Normformer: Improved transformer pretraining with extra normalization. *arXiv preprint arXiv:2110.09456*, 2021.

Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022. URL `https://openreview.net/forum?id=4nPswr1KcP`.

Hongyu Wang, Shuming Ma, Shaohan Huang, Li Dong, Wenhui Wang, Zhiliang Peng, Yu Wu, Payal Bajaj, Saksham Singhal, Alon Benhaim, et al. Foundation transformers. *arXiv preprint arXiv:2210.06423*, 2022.

Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. *Advances in Neural Information Processing Systems*, 34:30392–30400, 2021.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and improving layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *CVPR*, 2022a.

Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18123–18133, 2022b.

A Initial Project Idea

We arrived at the Dual PatchNorm solution via another project that explored adding whitened (decorrelated) patches to the inputs of ViT. Our initial prototype had a LayerNorm right after the decorrelated patches, to ensure that they are at an appropriate scale. This lead to improvements across multiple benchmarks, suggesting that whitened patches can improve image classification. Via ablations, later found out that just LayerNorm is sufficient and adding whitened patches on their own could degrade performance.

B Code

We perform all our experiments in the big-vision (Beyer et al., 2022c) library. Since the first LayerNorm of DPN is directly applied on pixels, we replace the first convolution with a patchify operation implemented with the einops (Rogozhnikov, 2022) library and a dense projection.

C ViT Projection Layer: Gradient Norm

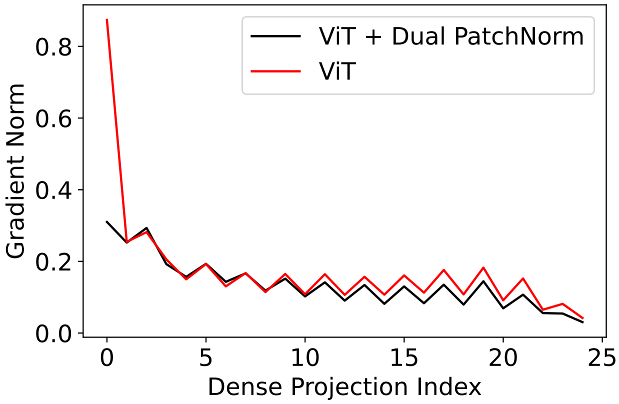


Figure 3: Gradient norm of each dense projection layer as a function of depth with and without DPN. Index 0 corresponds to the ViT patch embedding layer. DPN makes the gradient norm of the ViT-projection layer to be of the same scale as the other dense projections.

We analyse the gradient norms of different dense projections in a ViT B/32 model as a function of depth with and without DPN. Interestingly, Fig. 3 shows that DPN makes the gradient norm of the patch embedding to be of the same scale as the other dense projections within the Transformer blocks. However, we were unable to leverage this insight, to train models with DPN at higher learning rates. Without any other change in hyperparameters, ViTs with and without DPN, start to diverge roughly at the same learning rates.