RAW IMAGE PATCHES IN THE RESIDUAL STREAM: A DATA-EFFICIENT ISOTROPIC VISION TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

In cases where data or compute is constrained, state-of-the-art computer vision models have tended to be anisotropic, i.e. architecturally constrained to regard a different granularity of image detail in different layers. These are Convolutional Neural Networks (CNNs) or Transformer-based models incorporating CNN stems or inductive biases inspired by CNNs. This contrasts with the original ViT, which is isotropic in the sense that every layer represents the image as the same number of constant-size patches. Here we present an ablation study leading to a modernised, post-norm ViT that is isotropic and yet competitive with anisotropic Transformers in data- and compute-constrained settings. In particular, we match or exceed previous Transformer-based models when training for 300 epochs on CIFAR-100 and are competitive with similarly-sized SOTA models when training for 300 epochs on ImageNet-1k. The strength of our model comes partly from incorporating ideas proven effective since the original ViT paper (Dosovitskiy et al., 2020) and partly from three novel architectural choices. First, our model uses a novel re-weighting of the add-and-norm step of Transformers, allowing stable training across a wide range of depths. Second, we use a Transformer feed-forward block as a stem in place of the usual patch embedding and, third, we pass RMSNormed image patches directly into the residual stream.

1 INTRODUCTION

Vision Transformer (Dosovitskiy et al., 2020) (ViT) models apply the scalable sequence-processing ability of transformers (Vaswani et al., 2017) to images by pre-processing an image into a sequence of tokens. ViTs are amenable to predictable improvements from model scaling in much the same way that has been demonstrated for large language models (Zhai et al., 2022; Dehghani et al., 2023; Alabdulmohsin et al., 2023). However, due to their less image-specific inductive biases (Cordonnier et al., 2020; Raghu et al., 2021), they may be outperformed by convolutional neural networks (CNNs) in situations where training data is scarce or compute is somehow restricted (Kaya et al., 2025), for example when restricted to training from scratch for a fixed number of epochs, using a fixed parameter budget. Thus it has been shown that hybrid models mixing both convolutional layers and Transformer blocks (Wu et al., 2021; Vasu et al., 2023), or featuring a convolutional stem leading into a Transformer (Xiao et al., 2021; Hassani et al., 2022), are more data efficient than Transformer-only models. Accordingly, efforts have also been made to incorporate CNN-like features into Transformer-like architectures, for example using more convolution-like attention (d'Ascoli et al., 2021; Liu et al., 2021) or pooling (Marin et al., 2023). The common feature shared by CNNs, hybrid architectures and the best-performing Transformer architectures of the past 5 years has been that they are anisotropic. That is, they treat the image as a large number of small spatial patches in early layers, and feature a pyramid-like computational graph that explicitly combines these small patches into aggregate representations of larger areas of the image.

In this paper, we present an isotropic vision Transformer model that is data-efficient and competitive with state-of-the-art (SOTA) models in compute-constrained settings. This is of practical interest for two reasons. First, improvements in the stability and data-efficiency of isotropic Transformers may be applicable to the large-scale settings where these are often applied (Siméoni et al., 2026; Grattafiori et al., 2024). Second, isotropic vision transformers have fewer inductive biases than for example more vision-specialised models and an isotropic Transformer that works well on small data sets is a good baseline choice for data modalities for which the expedient inductive biases are

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

not well known. Motivated by this, we present an ablation study (see section 4.2) in the spirit of NeoBERT (Le Breton et al., 2025). The baseline of our ablation study is a post-norm Transformer, stabilised in two main ways. First we apply a novel variant of DeepNorm (Wang et al., 2024) comparable to but distinct from the work of Liu et al. (2020), described in section 3.1.2 and justified mathematically in section 3.1.3. From this starting point we apply a number of architectural improvements described in section 4.2, culminating in a 14M-parameter ViT that matches or exceeds all previous Transformer-based models when training for 300 epochs on CIFAR-100. We then scale our model to 63M parameters and show that it can also perform competitively with SOTA Transformer models when trained for 300 epochs on ImageNet-1k (Deng et al., 2009). In all cases we retain the same learning rate and weight decay hyperparameters, which we do not claim are optimal.

2 Related Work

2.1 Safely Scaling Post-Norm Transformers

The closest existing work to ours in terms of stabilising post-norm transformers is Liu et al. (2020), who re-weight the add-and-norm step to favour the residual stream as we do. In their work, the residual is weighted based on its observed behaviour at initialisation. We weight the branches, which is equivalent since the sum is normed, and our method permits the same scaling factor for every branch as described in sections 3.1.2 and 3.1.3. Another nearby paper is Wang et al. (2023), who introduce a variant of DeepNorm (Wang et al., 2024) applied to a pre-norm Transformer architecture with additional LayerNorms. Our method originated as an adaptation of DeepNorm to incorporate new norm steps, but our new norms are a superset of those in Wang et al. (2023) and we apply them to a post-norm architecture.

Wang et al. (2024) successfully train a 1000-layer post-norm Transformer by making the residual stream relatively large in early training, bringing the initialised model closer to a stable identity function. A similar idea is used in our work, Wang et al. (2023) mentioned above, ReZero (Bachlechner et al., 2021), LayerScale (Touvron et al., 2021b) and others (e.g. Chen & Wei, 2026). ReZero, LayerScale and Keel Chen & Wei (2026) use learned multiplicative parameters to weight the summed residual and block components, which we also incorporate via the learned gain of the RMSNorms on our branch outputs, as discussed in section 3.1.2.

Several prior works demonstrate that additional norm layers can stabilise training (Shleifer et al., 2021; Wang et al., 2023; Henry et al., 2020) and we add a large number of additional norm steps to our architecture as described in section 3.1.1.

2.2 FF-first Architecture

Most architectures do not prescribe putting an FF block at the start of a computer vision model. Press et al. (2020) experiment with FF-first transformers but come to the conclusion, in contrast to our findings, that this is worse than the usual practice of leading with attention blocks. Nevertheless, we are aware of two exceptions that support the idea. MLP-only computer vision architectures (Tolstikhin et al., 2021) are built almost entirely from Transformer-style MLP (i.e. FF) blocks and the hMLP stem of Touvron et al. (2022a) includes multiple blocks of patch aggregation followed by linear, LayerNorm, and activation steps analogous to the first half of our FF block stem but lacking the distinctive inverse bottleneck of a Transformer FF block.

Geva et al. (2021) show that the up-projection step of a FF block can be conceptualised as producing an unnormalised "key" vector for each input token, representing weights over some codebook of "memories" represented by the weights of the down-projection layer. Further, Ghiasi et al. (2022) show that the first layers of Transformer FF blocks learn to act like filter banks, similar to how the first layer of a CNN network performs. Thus, it seems sensible to have a FF block at the start of a ViT and in this work we show that indeed this works well.

3 METHODS

3.1 OUR BASELINE VIT

In recent literature pre-norm Transformer architectures, i.e. those where the output of feed-forward (FF) and attention blocks is added to the residual stream without a following normalisation, the inputs to those branch functions already having been normalised by LayerNorm (Ba et al., 2016) or RMSNorm (Zhang & Sennrich, 2019), have been more popular as a result of their stability, especially early in training (Xiong et al., 2020). Branch outputs in pre-norm transformers are calculated like this:

$$x_t = x_{t-1} + f(n(x_{t-1})) \tag{1}$$

where f is the branch function (i.e. an attention or FF block) and n is a norm function.

In contrast, in post-norm architectures the sum of the residual stream and a branch output is normalised:

$$x_t = n(x_{t-1} + f(x_{t-1})) \tag{2}$$

These are known to have the potential for better performance in an ideal setting (Liu et al., 2020; Wang et al., 2024), but are less stable in early training and increasingly so at greater depth if no mitigating measures are taken.

3.1.1 ADDITIONAL NORMS

We use RMSNorm (Zhang & Sennrich, 2019) rather than LayerNorm (Ba et al., 2016) throughout our models. We refer to these functions as "norms" in this section, short for normalising functions, not to be confused with e.g. the L2 norm of a tensor.

We apply a large number of additional norms to our architecture compared to in a standard Transformer, finding that this improves stability without hurting performance. The output of almost every weight tensors in our architecture is RMSNormed, so that its output is invariant to the norm of its weights. This means that only the direction of each weight tensor is important, in a way reminiscent of Loshchilov et al. (2025). Furthermore, the outputs of all residual stream branches lie on a hypersphere at initialisation as discussed in section 3.1.3, drawing a further parallel with the latter work.

We normalise the query and key of every attention block (Q/K norm) (Henry et al., 2020). We also add output norms to attention blocks as used in Shleifer et al. (2021) and normalisation between the two linear components of our FF blocks, as seen in the latter paper and also in Wang et al. (2023). We norm the output of the first linear projection of our FF blocks both before and after activation, normalising the gate and value elements of the pre-activation tensor separately in cases where GLU activations (Shazeer, 2020) are used. Note however that the baseline model at the beginning of the ablation uses GELU. The post-activation normalisation is observable in previous works (Shleifer et al., 2021; Wang et al., 2023). The pre-activation normalisation is unique to our architecture as far as we are aware, and ensures that weight decay has a similar meaning for the first linear projection as for the second, i.e. affecting the equilibrium effective learning rate (Kosson et al., 2024). Finally we add an output norm to all our FF blocks, analogous to the attention block output norm. All of our added norms retain the default characteristic of applying a learned affine transform after the norming operation, except in the case of the Q/K norm where this would potentially undermine the protection those provide against sending too-large logits into the softmax function, and analogously in the case of the pre-activation norm of the first linear layer of FF blocks. For a diagram of the placement of our norms, see Appendix A.

3.1.2 RECONCILING DEEPNORM WITH SCALE-INVARIANT LAYERS

We use a variant of DeepNorm (Wang et al., 2024), which allows trivial depth scaling as demonstrated in our experiments and visualised in Appendix B. As originally conceived, DeepNorm calls for FF block layers, value-projection layers and attention output projections in Transformers to have their initial weights multiplied by an attenuating coefficient $\beta < 1$, which has the effect of shrinking the branch output relative to the residual stream, while the latter is multiplied by a coefficient $\alpha > 1$. Our additional normalisation functions, discussed above, are added to some layers whose

weight initialisations are scaled in DeepNorm. This causes the output of those layers to be invariant to their initialisation, undermining the desired effect. Additionally, with our additional norm functions applied, the size of the initialised weights in these layers sets their initial effective learning rate (Van Laarhoven, 2017). When using an adaptive optimiser such as the AdamW variant we use (see section 4.1) the effective learning rate will ultimately reach an equilibrium (Kosson et al., 2024). However, as the weight initialisation is calculated based on depth, the starting effective learning rate will be different depending on the depth of the network, which is a property we would like to avoid.

Thus, we initially calculated α and β in exactly the same way as Wang et al. (2024) but instead used them to directly weight the residual and branch elements, respectively, in the add-and-norm operation used after every Transformer block. This is a similar strategy to that presented in Liu et al. (2020), but with an analytic re-weighting of every add-and-norm rather than a different one per branch, facilitated by the fact that all of our branch outputs have the same norm, see section 3.1.3. This gives us equation 3 rather than what we showed in equation 2.

$$x_t = n(\alpha \cdot x_{t-1} + \beta \cdot f(x_{t-1})) \tag{3}$$

Ultimately we use a different approach to calculating α and β, as described in section 3.1.3 below. Note that because all branch outputs are RMSNormed and the RMSNorm steps have a learned affine transform, the latter's learned gain parameters can inflate the contribution of each branch to its subsequent add-and-norm step, away from its starting point $\propto \beta$.

3.1.3 Why We Weight Our Branch Outputs with a Coefficient $\rho \propto 1/\sqrt{S}$

We present reasoning for how our architecture behaves near initialisation, and our argument particularly relates to the Transformer stack of our model, which is where depth can vary and is a stack of S RMSNormed branches contributing to a residual stream. At initialisation every RMSNorm gain is one, so the input to the stack x_0, each residual-stream state x_i and each branch output $\hat{b}_i$ has norm $\sqrt{d_{\text{model}}}$, which we take as our unit of length.

Because RMSNorm ignores the scale of its argument, equation 3 becomes

$$x_i = n(x_{i-1} + \rho\,\hat{b}_i), \qquad \rho = \beta/\alpha, \tag{4}$$

which differs from the standard post-norm of equation 2 only by the factor ρ on the branch.

We assume that each branch's output is orthogonal to its input, $\hat{b}_i \perp x_{i-1}$, which holds approximately for randomly initialised high-dimensional branches. Each add-and-norm then divides by $\|x_{i-1} + \rho\,\hat{b}_i\| = \sqrt{1+\rho^2}$, and unrolling equation 4 gives

$$x_S = \sigma_{input}\, x_0 + \sum_{i=1}^{S} \rho\, A_i\, \hat{b}_i, \qquad A_i = (1+\rho^2)^{-(S-i+1)/2}, \qquad \sigma_{input} = (1+\rho^2)^{-S/2}. \tag{5}$$

Here A_i is the factor by which the norm of branch i's contribution $\rho\,\hat{b}_i$ is attenuated on its way to the output, and σ_{input} is the norm of the input that survives there, so that σ_{input}^2 is the input's share of the output's energy (squared norm). The input passes through the same S add-and-norms as the first branch, so $\sigma_{input} = A_1$.

We will come to choose $\rho = c/\sqrt{S}$ for a constant c, a well-established choice for unnormalised and pre-norm residual networks (Hayou et al., 2021; Yang et al., 2024; Bordelon et al., 2024). We show below why it suits our post-norm architecture. If ρ is chosen this way, small deviations from orthogonality do not accumulate with depth: each cosine $\langle \hat{b}_i, x_{i-1} \rangle$ is of order $d_{\text{model}}^{-1/2}$ and enters the divisor as $\sqrt{1+\rho^2+2\rho\langle \hat{b}_i, x_{i-1}\rangle}$, shifting $\log \sigma_{input}$ by about $\rho\langle \hat{b}_i, x_{i-1}\rangle$. Over S branches the total shift has standard deviation $\rho\sqrt{S/d_{\text{model}}} = c/\sqrt{d_{\text{model}}}$, or 4.4% for $c = 1$ at $d_{\text{model}} = 512$.

If ρ were fixed, σ_{input} and the A_i of early branches would decay exponentially with depth, and the output would forget the input and the earliest branches, consistent with the known difficulty of training deep post-norm Transformers (Xiong et al., 2020; Liu et al., 2020). With $\rho = c/\sqrt{S}$, however, because $(1 + c^2/S)^S$ increases with S towards e^{c^2},

$$A_i \geq \sigma_{input} > e^{-c^2/2} \quad \text{for every } i \text{ and every } S, \tag{6}$$

with equality on the left for the first branch. The contribution of every branch, and of the input, thus reaches the output with more than $e^{-c^2/2}$ of its original norm and more than e^{-c^2} of its energy, however deep the model.

We choose the parsimonious $c = 1$, whereby each branch joins the stream with $1/S$ of the stream's energy and retains at least $\frac{1}{e}$ of its energy in the output of the model. This is different to the ρ obtained by applying the α and β suggested by Wang et al. (2024) directly to our architecture, but has worked well in practice and a visual check of its suitability is shown in Appendix B.

That the contribution of every branch survives to the output according to these floors is what we expect to keep deep versions of our architecture trainable with AdamW. AdamW normalises each parameter's update by the magnitude of its gradient, so the attenuation of gradients along the residual path, by the same factors A_i, matters little in our setting. By equation 5, an update to branch i moves the output by A_i times the change in that branch's contribution, so that no branch's update is lost in the output however deep the model. The component of each branch's gradient that arrives directly along the residual path, carrying $\partial L/\partial x_S$, likewise retains at least $e^{-c^2/2}$ of its norm. This is consistent with Appendix B, where a 100-layer model with $\rho = 1$ fails to train, while with $\rho = 1/\sqrt{S}$ it trains at the same learning rate and weight decay as our shallower models.

3.2 FF-First Architecture

We find that an initial FF block can replace the patch embedding of a ViT without loss of accuracy, and that when this is in place there is no need to embed image patches before passing them into the Transformer. Instead, we simply flatten 3-dimensional image patch tensors by a space-to-depth operation. This is not always possible to do at the full patch size, for example in the case of ImageNet-1k (see section 4.3 below) where it is customary to use 16×16 image patches in three colours, leading to a flattened patch size of 768, which may be bigger than the desired d_{model} as it was in our case. Thus we define our method as featuring an optional resize step whereby each patch may be resized as an image before being flattened and added to the residual stream, per figure 1. In cases where patches are resized before being added to the residual stream, the model runs the first FF block on the full sized patch and the patch is resized only for the purposes of injecting it into the smaller d_{model}-sized tokens. Where the image patch added to the residual stream is smaller than d_{model}, as is always the case after resizing, we pad it with trailing zeros.

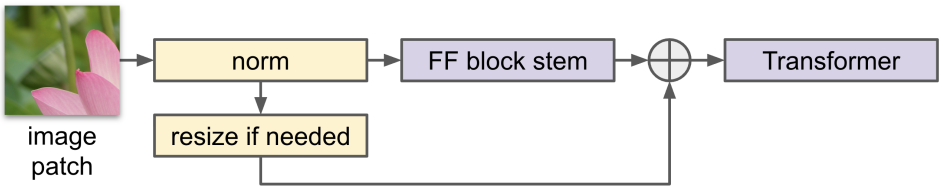

Figure 1: We allow the initial FF block of our model to operate on full-size image patches, but these may be too big to add to the resultant token. If so, we resize them.

Using an FF block as a ViT stem and treating the stem as a branch, i.e. adding normed image patches to the residual stream, is intuitively appealing in two ways. First, the FF block acts as a key value store, as described in Geva et al. (2021), whereby the columns of the weights of the second linear projection are a codebook of learned embeddings and the output of the block is a weighted sum of these. This provides a more sophisticated embedding than the linear projection typically used in ViT models. Second, as all Transformer layers have access to the output of earlier layers and to the input by design (Weiss et al., 2021), any subset of alternating attention and FF branches can in principle function as a valid ViT.

Regarding the former benefit, the behaviour of the first linear projection in the FF block stem of our design is demonstrably that of a filter bank leading into a code book of embeddings. As shown in figure 2, the learned filters are markedly like the edge-detection and colour opposition learned by early CNN layers (Zeiler & Fergus, 2014; Harris et al., 2021) and we propose that this facility is

what allows an initial FF block to replace the usual patch embedding without loss of accuracy, as shown in our ablation study in section 4.2.

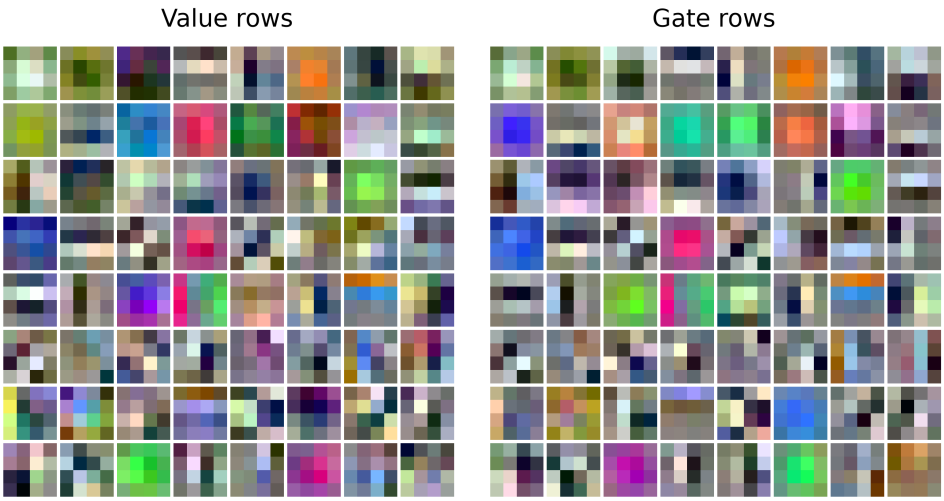

Figure 2: As we use a SwiGLU activation in our ViT, the model learns pairs of filters, each with a value filter dictating the disposition of the output and a gate filter dictating whether the filter pair is active. Sometimes, as in e.g. the 4th filter from the left in the top row, which detects a horizontal edge, the value and gate are very similar. At other times, as in e.g. the 3rd filter from the left in the row beneath the top row, the value filter looks different to the gate filter. In the latter example, the gate only activates for orange patches that are lighter in the middle, and when this happens the product of the patch and the (blue) value filter dictates the output, which in this case would be an attenuated response in the outline (orange multiplied by blue, which are approximately orthogonal) and positive in the middle (blue multiplied by white). The rule of thumb is that a patch must match the gate in order to produce a response from the filter pair, and the response is shaped by the value.

4 Experiments

We conduct an ablation study on CIFAR-100 (Krizhevsky et al., 2009), showing the progression from our baseline post-norm ViT model to our final model configuration. We then present the results of scaling the same model to 63M parameters and training it on ImageNet-1k (Deng et al., 2009).

4.1 General setup

In all our experiments we forgo the standard practice of standardising images by the data set mean and standard deviation, finding that applying an initial batch normalisation (Ioffe & Szegedy, 2015) to input image channels confers a similar benefit.

We use stochastic depth (Huang et al., 2016) in all models presented in this paper. Within each sample, each attention/FF pair is dropped with a probability that increases linearly such that the first pair is dropped with probability 0.0 and the final pair with probability 0.2. We do not rescale outputs when layers are dropped as in our model every layer rotates its input without rescaling.

Maximum learning rate (LR) is set to 0.006 and maximum weight decay (WD) to 0.05 for all experiments presented in this paper. RMSNorm affine transforms, all $<$ 2-dimensional tensors, including biases, and all embeddings are excluded from weight decay completely. We warm-up both LR and WD to their maxima from a minimum set to $\frac{1}{60}$ of maximum for CIFAR-100 and $\frac{1}{100}$ of maximum for ImageNet-1k. The LR and WD then decay back to their minima using a cosine schedule over the remaining epochs. Schedules are applied per training step. We warm up and decay both LR and WD, in contrast to more common recipes that only do so with LR, as the principal effect of WD in our highly normed network is to contribute to the equilibrium effective learning rate (Kosson et al., 2024). We use 5 warm-up epochs for CIFAR-100 and 10 for ImageNet-1k.

All models are trained with MuAdamW, a variant of AdamW provided by Microsoft (2022), which automatically applies μP (Yang et al., 2022) when provided a base model configuration, with the goal of achieving monotonic improvements of model performance with increasing model width. We use $\epsilon = 10^{-8}$ and $\beta_1, \beta_2 = 0.9, 0.999$, which are Pytorch default hyperparameters of the AdamW algorithm and also default in MuAdamW. We clip gradients to 1.0.

When scaling our models we follow the approach outlined in Alabdulmohsin et al. (2023) to find model geometries nearby the parameter counts we want and make small adjustments to make the parameter counts reach our intended targets while keeping a whole number of Transformer heads with $d_{head} = 64$. This approach is not necessarily as effective for our architecture as it was for the significantly different models in the latter work, so the model geometries we use in this work could potentially be improved.

In every case we report the accuracy of a model represented by the exponential moving average (EMA) of model weights across update steps (Polyak & Juditsky, 1992). We use decay rate 0.9999 for ImageNet-1k, the default related to this data set in the well-used `timm` library (Wightman, 2019), and 0.9995 for CIFAR-100.

4.2 Ablation Study on CIFAR-100

We use CIFAR-100 (Krizhevsky et al., 2009), using the standard train-test split with no additional validation split. Our results for the ablation study shown in figure 3 do not report a best epoch but rather the mean and standard deviation accuracy after 300 epochs for 5 seeds.

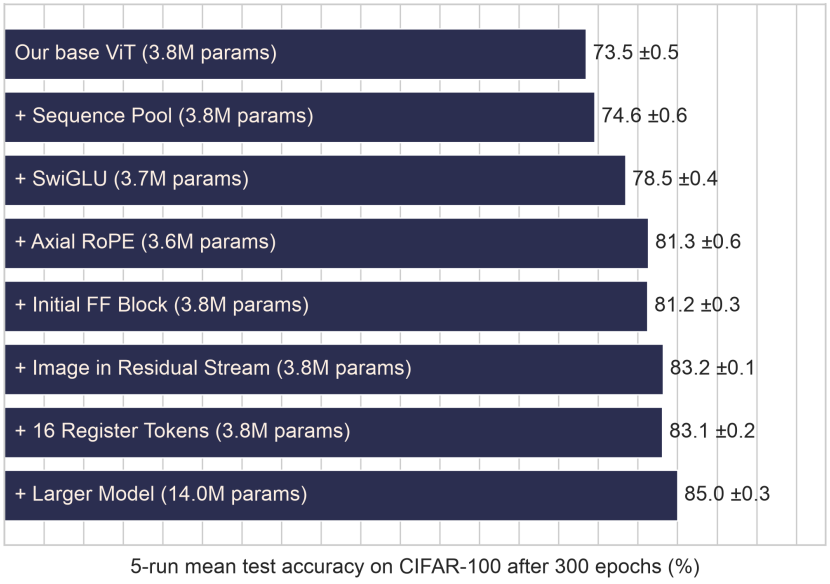

Figure 3: Our ablation study: mean and standard deviation of model variants trained for 300 epochs on CIFAR-100.

The ablation study begins with our baseline model described in section 3.1. For this we use patch embedding per the original ViT (Dosovitskiy et al., 2020), which we apply as a single CNN layer with kernel size = stride = 2, with no activation function and no pooling. The image class is projected from a learned class token, per the original ViT (Dosovitskiy et al., 2020).

We then progress to using the SequencePool classification head of Hassani et al. (2022), which calculates a single token for the purpose of projection by scoring each token in the output sequence by a query function and taking a weighted sum of the tokens based on the score.

The next stages add SwiGLU (Shazeer, 2020) activations rather than GELU, and then to add axial RoPE (Fang et al., 2024) instead of learned absolute position embeddings. In particular we use fractional axial RoPE (Khan et al., 2026), following the implementation provided by Wang (2025).

We then add our second and third architectural innovations, our baseline model already having contained our new method of add-and-norm-scaling. We add a FF block stem instead of a ViT-like embedding step and then introduce RMSNormed image patches to the residual stream. When we introduce the FF block stem, we additionally update the patch production process: it now has size 4, stride 2 and no projection. We simply RMSNorm the patch and perform a space-to-depth operation to turn it into a token, as described in section 3.2.

Finally, we add the register tokens recommended by Darcet et al. (2024). These provide less than 10% of sequence length in either CIFAR-100 or ImageNet-1k, and significantly less than 1% of parameters of the model. While they do not change accuracy in our ablation, we retain them following Darcet et al. (2024). This is the final change that provides the architecture we use for comparison in table 1, in 3.8M and 14.0M parameter sizes. As shown in table 1, our architecture outperforms previous SOTA Transformer-based models of similar and larger parameter counts.

Table 1: Results of Transformer-based models trained from scratch for 300 epochs on CIFAR-100. Our model performs the best among similarly-sized models, outperforming SOTA anisotropic Transformer models. * = reported in Zhang & Zhang (2025).

Model	Type	Parameters	Top-1 Acc.
MSCVIT-T (Zhang & Zhang, 2025)	Anisotropic	3.8M	80.1
CCT-7/3×1 (Hassani et al., 2022)	Anisotropic	3.8M	80.9
Ours (tiny)	Isotropic	3.8M	**83.1**
PVT-T* (Wang et al., 2021)	Anisotropic	13.2M	69.6
Swin-T* (Liu et al., 2021)	Anisotropic	27.5M	78.1
ESwin-T (Yao & Shao, 2024)	Anisotropic	18.4M	83.8
MSCVIT-S (Zhang & Zhang, 2025)	Anisotropic	14.0M	84.7
Ours (small)	Isotropic	14.0M	**85.0**

4.3 IMAGENET-1K

We scale up the best model from our CIFAR-100 ablation to 63M parameters using the same scaling heuristic used in that experiment (Alabdulmohsin et al., 2023). We then train one seed for 300 epochs on ImageNet-1k (Deng et al., 2009), on 2 A100 GPUs with combined batch size 1024. As is customary and in keeping with the comparators shown, we report the best accuracy achieved on the validation set at any epoch. As shown in table 2, our model performs comparably with similarly-sized SOTA Transformer-based models. No additional training data, pretraining or model distillation is used. Other than not standardising the input images, as discussed in section 4.1, our image preprocessing and augmentation approach is exactly that used in Touvron et al. (2021a). We use patch size 16 and stride 16 for our ImageNet-1k experiment.

5 CONCLUSION

Here we have presented an isotropic, post-norm Transformer that exhibits outstanding data efficiency. The stability of our model at depth is backed up by a novel mathematical argument presented in section 3.1.3 and is generally improved by the liberal application of additional normalisation steps. Atop these innovations, our ablation study shows that an FF block can replace the patch embedding of a ViT without loss of accuracy, and that with this stem in place, passing RMSNormed image patches directly into the residual stream improves accuracy by around two percentage points on CIFAR-100 (Krizhevsky et al., 2009).

We achieve strong results on data sets of significantly different sizes, CIFAR-100 and ImageNet-1k, transferring most of the hyperparameters of our models without tuning. We thus recommend our approach as a strong and forgiving baseline for any computer vision task, and particularly where data efficiency is desirable such as in medical or edge computing applications. To this end, we open source our code for the benefit of the community.

Table 2: Results of CNN and Transformer models trained from scratch on ImageNet-1k for 300 epochs. We give model performance as reported in the related publication and do not attempt to reproduce results from those works. Our model is competitive with the best current Transformer-based models, which still fall short of the best CNN models in this compute-constrained regime. Results for CaiT are at standard resolution for 300 epochs as reported in Table 8 of Touvron et al. (2021b)

Model	Type	Parameters	Top-1 Acc.
SW-small (Li et al., 2025)	CNN	56M	**83.9**
UniRepLKNet-S (Ding et al., 2024)	CNN	56M	**83.9**
ConvNeXt-S (Liu et al., 2022)	CNN	50M	83.1
PVT v2-B4 (Wang et al., 2022)	Anisotropic Transformer	63M	**83.6**
Ours	Isotropic Transformer	63M	83.5
Swin-B (Liu et al., 2021)	Anisotropic Transformer	88M	83.5
PVT v2-B3 (Wang et al., 2022)	Anisotropic Transformer	45M	83.2
CaiT-S* (Touvron et al., 2021b)	Isotropic Transformer	68M	83.2
Swin-S (Liu et al., 2021)	Anisotropic Transformer	50M	83.0
DeiT-III-B (Touvron et al., 2022b)	Isotropic Transformer	87M	82.8

6 Limitations

As our work is carried out without access to large-scale compute, our innovations are only demonstrated to work at a relatively small scale and on a narrow range of problems. We claim our demonstration in this paper is sufficient to establish that the novel approaches presented here are promising and worthy of robust testing by other groups.

Each of the design decisions leading to our new architecture was chosen based on its benefit to test set performance on CIFAR-100 and further testing on a wider range of tasks will be required to fully rule out benefits specific to that data set. However, since many of the added elements are well-explored elsewhere and none at all involve tuning continuous parameters, we believe the elements of our final architecture will stand up to further testing.

We consider that the architecture described in this paper has important differences to the isotropic transformers most commonly used today, and so we would caution against making assumptions about the applicability of techniques refined on other Transformer architectures. On this basis we encourage those building on our work to question whether the heuristics we have used, for example μP and the model scaling approach of Alabdulmohsin et al. (2023), that nevertheless worked well, are well-justified for this architecture.

7 Ethics Statement

Although the ethics of AI is of increasing societal importance, and our work presented here could potentially be applied to malevolent purposes or inherently risky frontier models, we consider that presenting our work here is of low marginal risk. We favourably weigh this against the benefit of having stronger baseline models to use in data-poor applications of wide benefit to society, such as e.g. in supporting interpretation of medical images.

8 Reproducibility Statement

We make the code to reproduce our experiments available in the following three repositories:

- Code and configs for running experiments: [link obfuscated]
- Model code: [link obfuscated]
- Code related to learning rate schedules and optimisation: [link obfuscated]

9 AI DISCLOSURE

In this work, we used generative AI tools for retrieval and discovery of relevant existing research, for ideation, and to support the production of accurate code. We have not used generative AI tools to draft or refine written content or to prove mathematical claims, although generative AI was used for ideation around mathematical argumentation. The disclosure required by ICLR in cases where generative AI helps to generate synthetic data is not relevant to this paper.

REFERENCES

Ibrahim M Alabdulmohsin, Xiaohua Zhai, Alexander Kolesnikov, and Lucas Beyer. Getting vit in shape: Scaling laws for compute-optimal model design. *Advances in Neural Information Processing Systems*, 36:16406–16425, 2023.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in artificial intelligence*, pp. 1352–1361. PMLR, 2021.

Blake Bordelon, Lorenzo Noci, Mufan Li, Boris Hanin, and Cengiz Pehlevan. Depthwise hyperparameter transfer in residual networks: Dynamics and scaling limit. In *International Conference on Learning Representations*, volume 2024, pp. 22088–22127, 2024.

Chen Chen and Lai Wei. Post-layernorm is back: Stable, expressive, and deep, 2026. URL https://arxiv.org/abs/2601.19895.

Jean Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *8th International Conference on Learning Representations, ICLR 2020*, 2020.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International Conference on Learning Representations (ICLR)*, 2024.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International conference on machine learning*, pp. 7480–7512. PMLR, 2023.

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

Xiaohan Ding, Yiyuan Zhang, Yixiao Ge, Sijie Zhao, Lin Song, Xiangyu Yue, and Ying Shan. Unireplknet: A universal perception large-kernel convnet for audio video point cloud time-series and image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5513–5524, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International conference on machine learning*, pp. 2286–2296. PMLR, 2021.

Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 149:105171, 2024.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021.

Amin Ghiasi, Hamid Kazemi, Eitan Borgnia, Steven Reich, Manli Shu, Micah Goldblum, Andrew Gordon Wilson, and Tom Goldstein. What do vision transformers learn? a visual exploration, 2022. URL https://arxiv.org/abs/2212.06727.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Ethan Harris, Daniela Mihai, and Jonathon Hare. How convolutional neural network architecture biases learned opponency and color tuning. *Neural Computation*, 33(4):858–898, 2021.

Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the big data paradigm with compact transformers, 2022. URL https://arxiv.org/abs/2104.05704.

Soufiane Hayou, Eugenio Clerico, Bobby He, George Deligiannidis, Arnaud Doucet, and Judith Rousseau. Stable resnet. In *International Conference on Artificial Intelligence and Statistics*, pp. 1324–1332. PMLR, 2021.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4246–4253, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.379. URL https://aclanthology.org/2020.findings-emnlp.379/.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pp. 646–661. Springer, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Alon Kaya, Igal Bilik, and Inna Stainvas. A comparative study of vision transformers and cnns for few-shot rigid transformation and fundamental matrix estimation. *arXiv preprint arXiv:2510.04794*, 2025.

Mohammad Aflah Khan, Krishna P Gummadi, Manish Gupta, and Abhilasha Ravichander. Fractional rotation, full potential? investigating performance and convergence of partial rope. *arXiv preprint arXiv:2603.11611*, 2026.

Atli Kosson, Bettina Messmer, and Martin Jaggi. Rotational equilibrium: How weight decay balances learning across neural networks. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=MQirNNU2pC.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Lola Le Breton, Quentin Fournier, John Xavier Morris, Mariam El Mezouar, and Sarath Chandar. Neobert: A next generation bert. *Transactions on Machine Learning Research*, 2025.

Dachong Li, Li Li, Zhuangzhuang Chen, and Jianqiang Li. Shiftwiseconv: small convolutional kernel with large kernel effect. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25281–25291, 2025.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. Understanding the difficulty of training transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5747–5763, 2020.

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*, pp. 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. ngpt: Normalized transformer with representation learning on the hypersphere. In *International Conference on Learning Representations*, volume 2025, pp. 74014–74038, 2025.

Dmitrii Marin, Jen-Hao Rick Chang, Anurag Ranjan, Anish Prabhu, Mohammad Rastegari, and Oncel Tuzel. Token pooling in vision transformers for image classification. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 12–21, 2023.

Microsoft. Maximal update parametrization and hyperparameter transfer (v1.0.0). `https://github.com/microsoft/mup`, 2022.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

Ofir Press, Noah A Smith, and Omer Levy. Improving transformer models by reordering their sublayers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2996–3005, 2020.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

Noam Shazeer. GLU variants improve transformer. *CoRR*, abs/2002.05202, 2020. URL `https://arxiv.org/abs/2002.05202`.

Sam Shleifer, Jason Weston, and Myle Ott. Normformer: Improved transformer pretraining with extra normalization, 2021. URL `https://arxiv.org/abs/2110.09456`.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michael Ramamonjisoa, Francisco Massa, Daniel HAZIZA, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Herve Jegou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL `https://openreview.net/forum?id=2NlGyqNjns`. Featured Certification.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 24261–24272. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/cba0a4ee5ccd02fda0fe3f9a3e7b89fe-Paper.pdf`.

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*, pp. 10347–10357. PMLR, 2021a.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 32–42, 2021b.

Hugo Touvron, Matthieu Cord, Alaaeldin El-Nouby, Jakob Verbeek, and Hervé Jégou. Three things everyone should know about vision transformers. In *European Conference on Computer Vision*, pp. 497–515. Springer, 2022a.

Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *European conference on computer vision*, pp. 516–533. Springer, 2022b.

Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. Fastvit: A fast hybrid vision transformer using structural reparameterization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5785–5795, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Hongyu Wang, Shuming Ma, Shaohan Huang, Li Dong, Wenhui Wang, Zhiliang Peng, Yu Wu, Payal Bajaj, Saksham Singhal, Alon Benhaim, et al. Magneto: A foundation transformer. In *International conference on machine learning*, pp. 36077–36092. PMLR, 2023.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6761–6774, 2024.

Phil Wang. Rotary embeddings - pytorch. https://github.com/lucidrains/rotary-embedding-torch, 2025.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 548–558, 2021. doi: 10.1109/ICCV48922.2021.00061.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. *Computational visual media*, 8(3):415–424, 2022.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In *International Conference on Machine Learning*, pp. 11080–11090. PMLR, 2021.

Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

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*, pp. 22–31, 2021.

Tete Xiao, Piotr Dollar, Mannat Singh, Eric Mintun, Trevor Darrell, and Ross Girshick. Early convolutions help transformers see better. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Lpfh1Bpqfk.

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.

Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer, 2022. URL https://arxiv.org/abs/2203.03466.

Greg Yang, Dingli Yu, Chen Zhu, and Soufiane Hayou. Tensor programs vi: Feature learning in infinite depth neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 55099–55150, 2024.

Dazhi Yao and Yunxue Shao. A data efficient transformer based on swin transformer: D. yao, y. shao. *The Visual Computer*, 40(4):2589–2598, 2024.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12104–12113, 2022.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Bowei Zhang and Yi Zhang. Mscvit: A small-size vit architecture with multi-scale self-attention mechanism for tiny datasets. *Neural networks*, 188:107499, 2025.

A Placement of Additional RMSNorms in Our Model

In section 3.1.1 we describe the addition of several new RMSNorms to our architecture. For clarity we include a diagram of where these appear in figure 4.

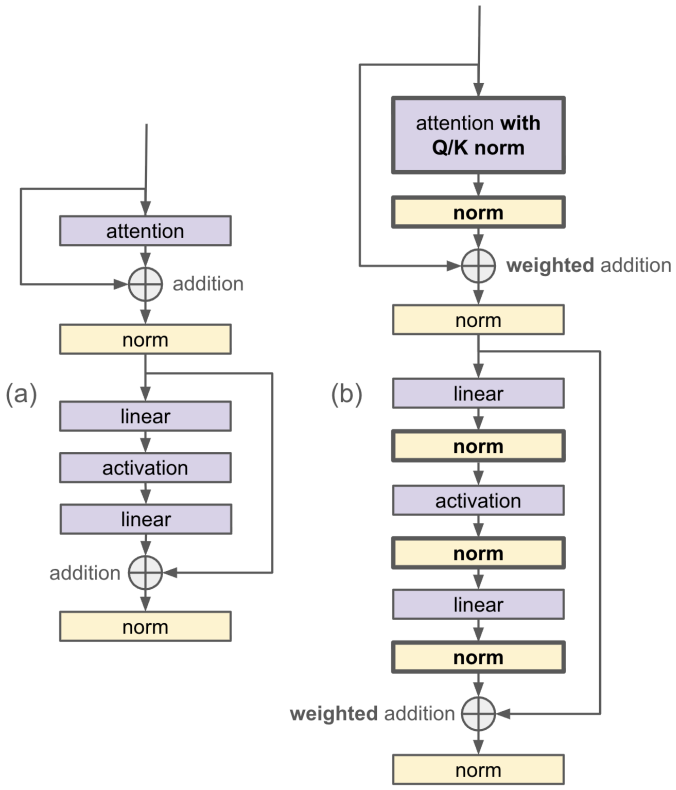

Figure 4: The normalisation scheme used in our network (b) compared with a standard post-norm Transformer architecture. Boxes in bold on the right-hand diagram indicate the addition of a norm not included on the left. The weighted addition refers to the use of α and β as discussed in section 3.1.2

B Visualising The Effects of Our New Add-and-Norm Weighting

As described in section 3.1.3, we multiply each branch's contribution to the add-and-norm steps in out model by $\rho = 1/\sqrt{S}$, where S is the number of branches in the model. The standard Transformer (Vaswani et al., 2017) and most models derived from it, use an implicit ρ of 1. In figure 5, we show that this branch weighting allows training successfully at 100 layers (200 branches), whereas our architecture cannot train at all on CIFAR-100 at this depth with $\rho = 1$.

Choices of ρ that bound the size of a model update to $\mathcal{O}(1)$ with respect to depth are in the family $c/\sqrt{S}$. Applying the α and β calculated in Wang et al. (2024) directly to our residual and branch

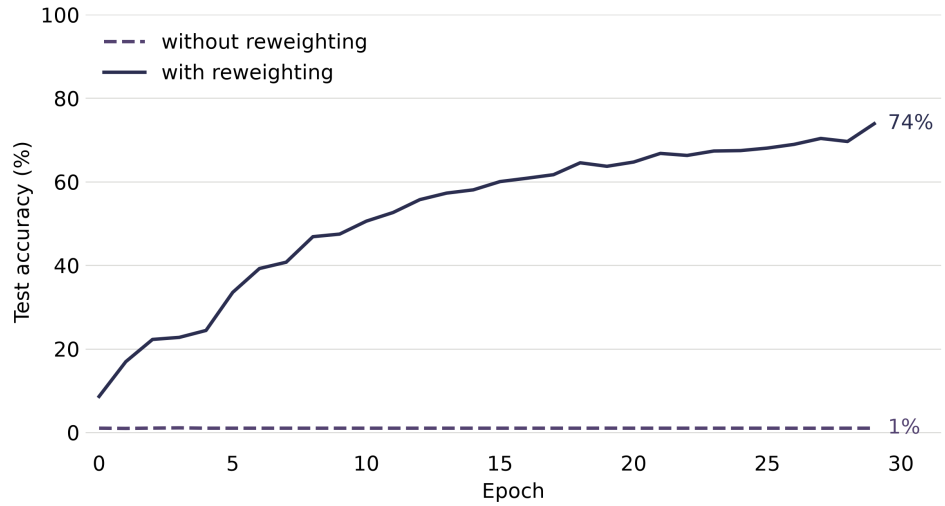

Figure 5: By multiplying each branch's contribution to the add-and-norm steps in out model by $\rho = 1/\sqrt{S}$, we allow a model to stably train on CIFAR-100 at 100 layers while using the same hyperparameters as we use for our 7-, 11-, and 21-layer models in this paper. 30 epochs shown here for illustrative purposes. The model without re-weighting ($\rho = 1$) stays close to 1% accuracy throughout the run.

terms, respectively, is equivalent to choosing $\rho = 1/\sqrt{2S}$, i.e. $c = 1/\sqrt{2}$. As we show in figure 6, the contribution of each branch to the energy (squared norm) of the model output increases during training, which it is able to do due to the learned gain within the RMSNorm on its output. The figure provides a visual check that our chosen operating point of $c = 1$ is in keeping with the trained state of the model visualised. $c = 1$ initialises branch contributions to be louder than they would be with $c = 1/\sqrt{2}$, but still consistently below where we expect them to move to through training.

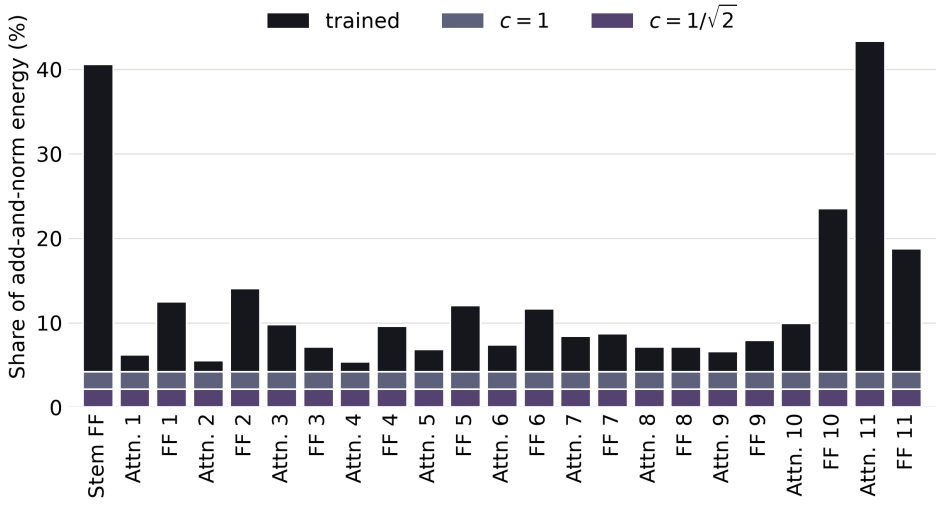

Figure 6: In the add-and-norm sum following each branch, the branch itself contributes a certain share of the energy (squared norm) depending on the chosen c, see section 3.1.3. When we use the α and β provided by the formulas of Wang et al. (2024), we start at $c = 1/\sqrt{2}$ and a large version of our CIFAR-100 models finishes at a learned state where the branch contribution is always higher. This provides a suitability check for our operating point of $c = 1$. Indeed it starts between the two values and we assume that this allows the model to reach the optimal contribution to each branch with no difficulty over and above that posed by $c = 1/\sqrt{2}$.