Model	Res.	Params	Speed	Flops	IN-1k	IN-C	IN-A	IN-R	FGSM	PGD
Transformers										
ViT-B/16‡	224	86 M	182	16.9	77.9	52.2	7.0	21.9	30.6	14.3
ViT-L/16‡	224	307 M	55	59.7	76.5	49.3	6.1	17.9	27.8	13.0
ViT-B/16 21k‡	224	86 M	182	16.9	84.0	65.8	26.7	38.0	31.3	10.3
ViT-L/16 21k‡	224	307 M	55	59.7	**85.1**	**70.0**	28.1	40.6	40.5	16.2
DeiT-S†	224	22 M	544	4.6	79.9	55.4	18.9	31.0	40.7	16.7
DeiT-B†	224	87 M	182	17.6	82.0	60.7	27.4	34.6	46.4	21.3
ConViT-S†	224	28 M	296	5.4	81.5	59.5	24.5	34.0	41.0	17.2
ConViT-B†	224	87 M	139	17.7	82.4	61.9	29.0	36.9	51.8	20.8
RVT-S†	224	23.3 M	-	4.7	81.9	-	25.7	47.7	51.8	28.2
RVT-B†	224	91.8 M	-	17.7	82.6	-	28.5	48.7	53.0	29.9
CNNs										
ResNet50‡	224	25 M	736	4.1	76.8	46.1	4.2	21.5	-	-
ResNet101‡	224	45 M	435	7.85	78.0	50.2	6.3	23.0	14.7	2.0
ResNet101x3‡	224	207 M	62	69.6	80.3	53.4	9.1	24.5	23.6	7.3
ResNet152x4‡	224	965 M	18	183.1	80.4	54.5	11.6	25.8	33.3	10.5
ResNet50-RS	160	36 M	938	4.6	78.8	36.8	5.7	39.1	28.7	18.4
ResNet101-RS	192	64 M	674	12.1	80.3	44.1	11.8	44.8	32.9	18.8
ResNet152-RS	256	87 M	304	31.2	81.2	49.9	23.4	45.9	41.6	28.5
ResNet200-RS	256	93 M	225	40.4	82.8	49.3	25.4	48.1	40.4	24.6
ResNet270-RS	256	130 M	152	54.2	83.8	53.6	26.6	48.7	44.7	30.3
ResNet350-RS	288	164 M	89	87.5	84.0	53.9	34.9	49.7	48.3	34.6
Our Transformed CNNs										
T-ResNet50-RS	224	38 M	447	17.6	81.0	48.0	18.7	42.9	47.2	33.9
T-ResNet101-RS	224	66 M	334	25.1	82.4	52.9	27.7	47.8	50.3	34.2
T-ResNet152-RS	320	89 M	128	65.8	83.7	54.5	39.8	50.6	57.3	36.8
T-ResNet200-RS	320	96 M	105	80.2	84.0	57.0	41.2	51.1	58.3	36.4
T-ResNet270-RS	320	133 M	75	107.2	84.3	58.6	43.7	51.4	**59.0**	**36.6**
T-ResNet350-RS	320	167 M	61	130.5	84.5	59.2	**44.8**	**53.8**	53.4	36.4

Table 2: **Accuracy of our models on various benchmarks.** Throughput is the number of images processed per second on a V100 GPU at batch size 32. For ImageNet-C, we keep a resolution of 224 at test time to avoid distorting the corruptions; this disadvantages our large models, which are trained at higher resolutions. †: reported from (Mao et al., 2021) (we recalculated ImageNet-C accuracies, as the original paper reports MCE). ‡: reported from (Bhojanapalli et al., 2021) (in their setup, PGD uses 8 steps with a stepsize of 1/8).

A Performance table

In Tab. 2, we display the characteristics and the performance of our T-ResNet-RS models and compare them to the original ResNet-RS models as well as several other strong baselines reported in Bhojanapalli et al. (2021); Mao et al. (2021).

B When should one start learning the self-attention layers?

We have demonstrated the benefits of initializing T-CNNs from pre-trained CNNs, a very compelling procedure given the wide availability of pretrained models. But one may ask: how does this compare to training a hybrid model from scratch? More generally, given a computational budget, how long should the SA layers be trained compared to the convolutional backbone?

Transformed CNN versus hybrid models To answer the first question, we consider a ResNet-50 trained on ImageNet for 400 epochs. We use SGD with momentum 0.9 and a batch size of 1024,

Name	t_1	t_2	Train time	Top-1
Vanilla CNN	400	0	2.0k mn	79.04
Vanilla CNN↑320	450	0	2.4k mn	**79.78**
T-CNN	400	50	2.3k mn	79.88
T-CNN↑320	400	50	2.7k mn	**80.84**
Vanilla hybrid	0	400	2.8k mn	79.95
T-CNN*	100	300	2.6k mn	**80.44**
T-CNN*	200	200	2.4k mn	80.28
T-CNN*	300	100	2.2k mn	79.28

Table 3: **The benefit of late reparametrization.** We report the top-1 accuracy of a ResNet-50 on ImageNet reparameterized at various times t_1 during training. ↑320 stands for fine-tuning at resolution 320. The models with a $\star$ keep the same optimizer after reparametrization, in contrast with the usual T-CNNs.

warming up the learning rate for 5 epochs before a cosine decay. To achieve a strong baseline, we use the same augmentation scheme as in Touvron et al. (2020) for the DeiT. Results are reported in Tab. 3. In this modern training setting, the vanilla ResNet50 reaches a solid performance of 79.04% on ImageNet, well above the 77% usually reported in litterature.

The T-CNN obtained by fine-tuning the ResNet for 50 epochs at same resolution obtains a top-1 accuracy of 79.88%, with a 15% increase in training time, and 80.84 as resolution 320, with a 35% increase in training time. In comparison, the hybrid model trained for 400 epochs in the same setting only reaches 79.95%, in spite of a 40% increase in training time. Hence, fine-tuning yields better results than training the hybrid model from scratch.

What is the best time to reparametrize? We now study a scenario between the two extreme cases: what happens if we reparametrize halfway through training? To investigate this question in a systematic way, we train the ResNet50 for t_1 epochs, then reparametrize and resume training for another t_2 epochs, ensuring that $t_1 + t_2 = 400$ in all cases. Hence, $t_1 = 400$, amounts to the vanilla ResNet50, whereas $t_1 = 0$ corresponds to the hybrid model trained from scratch. To study how final performance depends on t_1 in a fair setting, we keep the same optimizer and learning rate after the reparametrization, in contrast with the fine-tuning procedure which uses fresh optimizer.

Results are presented in Tab. 3. Interestingly, the final performance evolves non-monotonically, reaching a maximum of 80.44 for $t_1 = 100$, then decreasing back down as the SA layers have less and less time to learn. This non-monotonicity is remarkably similar to that observed in d'Ascoli et al. (2019), where reparameterizing a CNN as a FCN in the early stages of training enables the FCN to outperform the CNN. Crucially, this result suggests that reparametrizing during training not only saves time, but also helps the T-CNN find better solutions.

C Changing the learning rate

We have shown that the learning dynamics decompose into two phases: the learning rate warmup phase, where the test loss drops, then the learning rate decay phase, where the test loss increases again. This could lead one to think that the maximal learning rate is too high, and the dip could be avoided by choosing a lower learning rate. Yet this is not the case, as shown in Fig. 6. Reducing the maximal learning rate indeed reduces the dip, but it also slows down the increase in the second phase of learning. This confirms that the model needs to "unlearn" the right amount to find better solutions.

D Changing the test resolution

One advantage of the GPSA layers introduced by d'Ascoli et al. (2021) is how easily they adapt to different image resolutions. Indeed, the positional embeddings they use are fixed rather than learnt. They simply consist in 3 values for each pair of pixels: their euclidean distance $\|\boldsymbol{\delta}\|$, as well as their

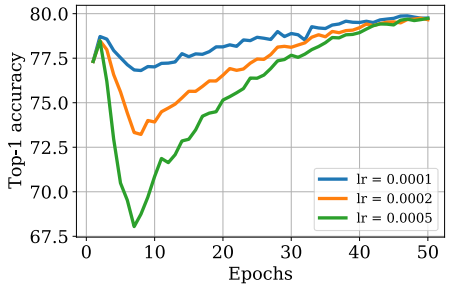


Figure 6: **The larger the learning rate, the lower the test accuracy dips, but the faster it climbs back up.** We show the dynamics of the ResNet50, fine-tuned for 50 epochs at resolution 224, for three different values of the maximal learning rate.

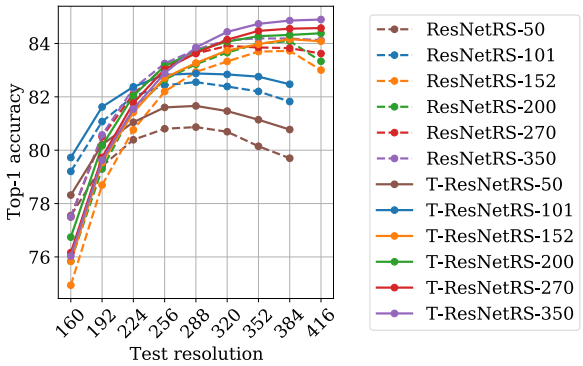


Figure 7: **Performance at different test-time resolutions, for the finetuned models with and without SA.** The ResNet50-RS and ResNet101-RS models are finetuned at resolution 224, and all other models are finetuned at resolution 320.

coordinate distance $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2$ (see Eq. 4). Our implementation automatically adjusts these embeddings to the input image, allowing us to change the test resolution seamlessly.

In Fig. 7, we show how the top-1 accuracies of our T-ResNet-RS models compares to those of the ResNet-RS models finetuned at same resolution but without SA. At test resolution 416, our T-ResNetRS-350 reaches an impressive top-1 accuracy of 84.9%, beyond those of the best EfficientNets and BotNets Srinivas et al. (2021).

E CHANGING THE NUMBER OF EPOCHS

In Tab. 4, we show how the top-1 accuracy of the T-ResNet-RS model changes with the number of fine-tuning epochs. As expected, performance increases significantly as we fine-tune for longer, yet we chose to set a maximum of 50 fine-tuning epochs to keep the computational cost of fine-tuning well below that of the original training.

F CHANGING THE ARCHITECTURE

Our framework, which builds on the timm package, makes changing the original CNN architecture very easy. We applied our fine-tuning procedure to the ResNet-D models He et al. (2019) with the exact same hyperparameters, and observed substantial performance gains, similar to the ones obtained for ResNet-RS, see Tab. 5. This suggests the wide applicability of our method.

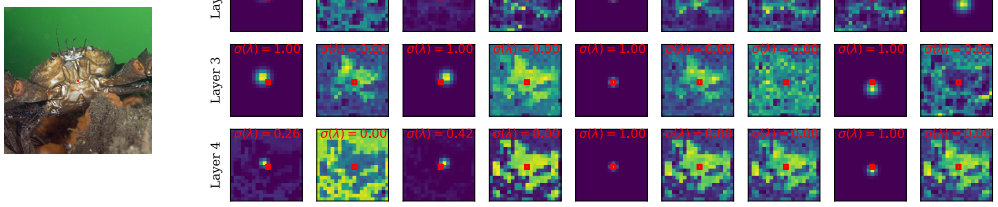
Model	Epochs	Top-1 acc
ResNet50-RS	0	79.91
T-ResNet50-RS	10	80.11
T-ResNet50-RS	20	80.51
T-ResNet50-RS	50	**81.02**
ResNet101-RS	0	81.70
T-ResNet101-RS	10	81.54
T-ResNet101-RS	20	81.90
T-ResNet101-RS	50	**82.39**

Table 4: **Longer fine-tuning increases final performance.** We report the top-1 accuracies of our models on ImageNet-1k at resolution 224.

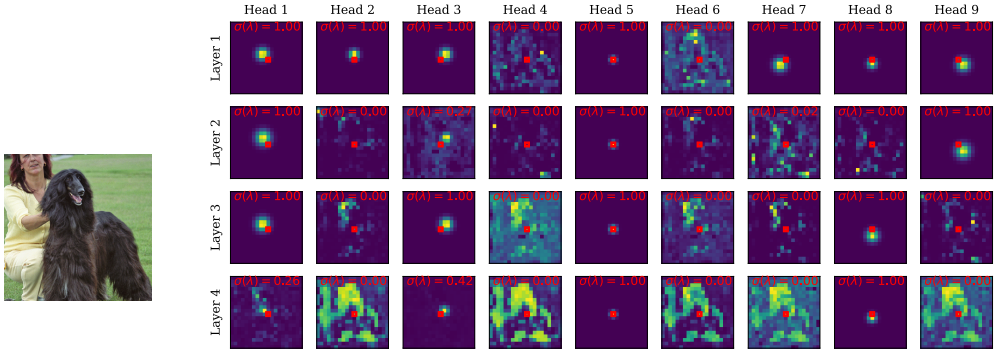
Model	Original res.	Original acc.	Fine-tune res.	Fine-tune acc.	Gain
T-ResNet50-D	224	80.6	320	81.6	+1.0
T-ResNet101-D	320	82.3	384	83.1	+0.8
T-ResNet152-D	320	83.1	384	83.8	+0.7
T-ResNet200-D	320	83.2	384	**83.9**	+0.7
T-ResNet50-RS	160	78.8	224	81.0	+2.8
T-ResNet101-RS	192	81.2	224	82.4	+1.2
T-ResNet152-RS	256	83.0	320	83.7	+0.7
T-ResNet200-RS	256	83.4	320	**84.0**	+0.6

Table 5: **Comparing the performance gains of the ResNet-RS and ResNet-D architectures.** Top-1 accuracy is measured on ImageNet-1k validation set. The pre-trained models are all taken from the timm library Wightman (2019).

G MORE ATTENTION MAPS



(a) Attention maps



(b) Attention maps

Figure 8: **GPSA layers combine local and global attention in a complementary way.** We depicted the attention maps of the four GPSA layers of the T-ResNet270-RS, obtained by feeding the image on the left through the convolutional backbone, then selecting a query pixel in the center of the image (red box). For each head h, we indicate the value of the gating parameter $\sigma(\lambda_h)$ in red (see Eq. 7). ($\sigma(\lambda_h) = 0$).