TRANSFORMED CNNS: RECASTING PRE-TRAINED CONVOLUTIONAL LAYERS WITH SELF-ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision Transformers (ViT) have recently emerged as a powerful alternative to convolutional networks (CNNs). Although hybrid models attempt to bridge the gap between these two architectures, the self-attention layers they rely on induce a strong computational bottleneck, especially at large spatial resolutions. In this work, we explore the idea of reducing the time spent training these layers by initializing them from pre-trained convolutional layers. This enables us to transition smoothly from any pre-trained CNN to its functionally identical hybrid model, called Transformed CNN (T-CNN). With only 50 epochs of fine-tuning, the resulting T-CNNs demonstrate significant performance gains over the CNN as well as substantially improved robustness. We analyze the representations learnt by the T-CNN, providing deeper insights into the fruitful interplay between convolutions and self-attention.

INTRODUCTION

Since the success of AlexNet in 2012 (Krizhevsky et al., 2017), the field of Computer Vision has been dominated by Convolutional Neural Networks (CNNs) (LeCun et al., 1998; 1989). Their local receptive fields give them a strong inductive bias to exploit the spatial structure of natural images (Scherer et al., 2010; Schmidhuber, 2015; Goodfellow et al., 2016), while allowing them to scale to large resolutions seamlessly. Yet, this inductive bias limits their ability to capture long-range interactions.

In this regard, self-attention (SA) layers, originally introduced in language models (Bahdanau et al., 2014; Vaswani et al., 2017; Devlin et al., 2018), have gained interest as a building block for vision Ramachandran et al. (2019); Zhao et al. (2020). Recently, they gave rise to a plethora of Vision Transformer (ViT) models, able to compete with state-of-the-art CNNs in various tasks Dosovitskiy et al. (2020); Touvron et al. (2020); Wu et al. (2020); Touvron et al. (2021); Liu et al. (2021); Heo et al. (2021) while demonstrating better robustness (Bhojanapalli et al., 2021; Mao et al., 2021). However, capturing long-range dependencies necessarily comes at the cost of quadratic complexity in input size, a computational burden which many recent directions have tried to alleviate (Bello, 2021; Wang et al., 2020; Choromanski et al., 2020; Katharopoulos et al., 2020). Additionally, ViTs are generally harder to train (Zhang et al., 2019; Liu et al., 2020), and require vast amounts of pre-training (Dosovitskiy et al., 2020) or distillation from a convolutional teacher (Hinton et al., 2015; Jiang et al., 2021; Graham et al., 2021) to match the performance of CNNs.

Faced with the dilemma between efficient CNNs and powerful ViTs, several approaches have aimed to bridge the gap between these architectures. On one side, hybrid models append SA layers onto convolutional backbones (Chen et al., 2018; Bello et al., 2019; Graham et al., 2021; Chen et al., 2021; Srinivas et al., 2021), and have already fueled successful results in a variety of tasks (Carion et al., 2020; Hu et al., 2018; Chen et al., 2020; Locatello et al., 2020; Sun et al., 2019). Conversely, a line of research has studied the benefit of introducing convolutional biases in Transformer architectures to ease learning (d'Ascoli et al., 2021; Wu et al., 2021; Yuan et al., 2021). Despite these interesting compromises, modelling long-range dependencies at low computational cost remains a challenge for practitioners.

Contributions At a time when pre-training on vast datasets has become common practice, we ask the following question: does one need to train the SA layers during the whole learning process? Could

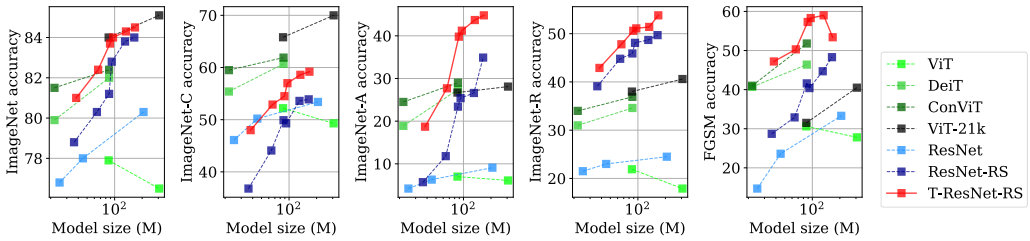


Figure 1: **Transformed ResNets strike a strong accuracy-robustness balance.** Our models (red) significantly outperform the original ResNet-RS models (dark blue) they were initialized from when evaluated on ImageNet-1k. They also exhibit solid performances on various robustness benchmarks (ImageNet-C, A and R, and FGSM adversarial attacks from left to right).

one instead learn cheap components such as convolutions first, leaving the SA layers to be learnt at the end? In this paper, we take a step in this direction by presenting a method to fully reparameterize any pre-trained convolutional layer as a *Gated Positional Self-Attention* (GPSA) layer (d'Ascoli et al., 2021). The latter is initialized to reproduce the mapping of the convolutional layer, but is then encouraged to learn more general mappings which are not accessible to the CNN by adjusting positional gating parameters.

We leverage this method to reparametrize pre-trained CNNs as functionally equivalent hybrid models. After only 50 epochs of fine-tuning, the resulting Transformed CNNs (T-CNNs) boast significant performance and robustness improvements as shown in Fig. 1, demonstrating the practical relevance of our method. We analyze the inner workings of the T-CNNs, and show that they learn more robust representations by combining convolutional heads and attentional heads in a complementary way.

Related work Our work mainly builds on two pillars. First, the idea that SA layers can express any convolution, introduced by Cordonnier et al. (2019). This idea was recently leveraged by d'Ascoli et al. (2021), which initialize the SA layers of the ViT as *random* convolutions and observe performance gains compared to the standard initialization, especially in the low-data regime where inductive biases are most useful. Our approach is a natural follow-up of this idea: what happens if the SA layers are instead initialized as *trained* convolutions?

Second, we exploit the following learning paradigm: train a simple and fast model, then reparameterize it as a more complex model for the final stages of learning. This approach was studied from a scientific point of view in d'Ascoli et al. (2019), which shows that reparameterizing a CNN as a fully-connected network (FCN) halfway through training can lead the FCN to outperform the CNN. Yet, the practical relevance of this method is limited by the vast increase in number of parameters required by the FCN to functionally represent the CNN. In contrast, our reparameterization hardly increases the parameter count of the CNN, making it easily applicable to any state-of-the-art CNN. Note that these reparameterization methods can be viewed an informed version of dynamic architecture growing algorithms such as AutoGrow (Wen et al., 2020).

In the context of hybrid models, various works have studied the performance gains obtained by introducing MHSA layers in ResNets with minimal architectural changes (Srinivas et al., 2021; Graham et al., 2021; Chen et al., 2021). However, the MHSA layers used in these works are initialized randomly and need to be trained from scratch. Our approach is different, as it makes use of GPSA layers, which can be initialized to represent the same function as the convolutional layer it replaces. We emphasize that the novelty in our work is not in the architectures used, but in the unusual way they are blended together.

1 Background

Multi-head self-attention The SA mechanism is based on a trainable associative memory with (key, query) vector pairs. To extract the semantic interpendencies between the L elements of a sequence $\boldsymbol{X} \in \mathbb{R}^{L \times D_{in}}$, a sequence of "query" embeddings $\boldsymbol{Q} = \boldsymbol{W}_{qry}\boldsymbol{X} \in \mathbb{R}^{L \times D_h}$ is matched against another sequence of "key" embeddings $\boldsymbol{K} = \boldsymbol{W}_{key}\boldsymbol{X} \in \mathbb{R}^{L \times D_h}$ using inner products. The

result is an attention matrix whose entry (ij) quantifies how semantically relevant $\boldsymbol{Q}_i$ is to $\boldsymbol{K}_j$:

$$\boldsymbol{A} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{D_h}}\right) \in \mathbb{R}^{L\times L}. \tag{1}$$

Multi-head SA layers use several SA heads in parallel to allow the learning of different kinds of dependencies:

$$\text{MSA}(\boldsymbol{X}) := \sum_{h=1}^{N_h} [\text{SA}_h(\boldsymbol{X})]\, \boldsymbol{W}_{out}^h, \qquad \text{SA}_h(\boldsymbol{X}) := \boldsymbol{A}^h \boldsymbol{X} \boldsymbol{W}_{val}^h, \tag{2}$$

where $\boldsymbol{W}_{\text{val}}^h \in R^{D_{in}\times D_v}$ and $\boldsymbol{W}_{\text{out}}^h \in R^{D_v \times D_{out}}$ are two learnable projections.

To incorporate positional information, ViTs usually add absolute position information to the input at embedding time, before propagating it through the SA layers. Another possibility is to replace the vanilla SA with positional SA (PSA), by including a position-dependent term in the softmax (Ramachandran et al., 2019; Shaw et al., 2018). Although there are several ways to parametrize the positional attention, we use encodings $\boldsymbol{r}_{ij}$ of the relative position of pixels i and j as in (Cordonnier et al., 2019):

$$\boldsymbol{A}_{ij}^h := \text{softmax}\left(\boldsymbol{Q}_i^h \boldsymbol{K}_j^{h\top} + \boldsymbol{v}_{pos}^{h\top} \boldsymbol{r}_{ij}\right). \tag{3}$$

Each attention head learns an embedding $\boldsymbol{v}_{pos}^h \in \mathbb{R}^{D_{pos}}$, and the relative positional encodings $\boldsymbol{r}_{ij} \in \mathbb{R}^{D_{pos}}$ only depend on the distance between pixels i and j, denoted denoted as a two-dimensional vector $\boldsymbol{\delta}_{ij}$.

Self-attention as a generalized convolution Cordonnier et al. (2019) shows that a multi-head PSA layer (Eq. 3) with N_h heads and dimension $D_{pos} \geq 3$ can express any convolutional layer of filter size $\sqrt{N}_h$, with D_{in} input channels and $\min(D_v, D_{out})$ output channels, by setting the following:

$$\begin{cases} \boldsymbol{v}_{pos}^h := -\alpha^h \left(1, -2\Delta_1^h, -2\Delta_2^h, 0, \dots 0\right) \\ \boldsymbol{r}_{\boldsymbol{\delta}} := \left(\|\boldsymbol{\delta}\|^2, \delta_1, \delta_2, 0, \dots 0\right) \\ \boldsymbol{W}_{qry} = \boldsymbol{W}_{key} := \boldsymbol{0} \end{cases} \tag{4}$$

In the above, the *center of attention* $\boldsymbol{\Delta}^h \in \mathbb{R}^2$ is the position to which head h pays most attention to, relative to the query pixel, whereas the *locality strength* $\alpha^h > 0$ determines how focused the attention is around its center $\boldsymbol{\Delta}^h$. When α^h is large, the attention is focused only on the pixel located at $\boldsymbol{\Delta}^h$; when α^h is small, the attention is spread out into a larger area. Thus, the PSA layer can achieve a convolutional attention map by setting the centers of attention $\boldsymbol{\Delta}^h$ to each of the possible positional offsets of a $\sqrt{N_h} \times \sqrt{N_h}$ convolutional kernel, and sending the locality strengths α^h to some large value.

2 Approach

In this section, we introduce our method for mapping a convolutional layer to a functionally equivalent PSA layer with minimal increase in parameter count. To do this, we leverage the GPSA layers introduced in d'Ascoli et al. (2021).

Loading the filters We want each head h of the PSA layer to functionally mimic the pixel h of a convolutional filter $\boldsymbol{W}_{\text{filter}} \in \mathbb{R}^{N_h \times D_{in} \times D_{out}}$, where we typically have $D_{out} \geq D_{in}$. Rewriting the action of the MHSA operator in a more explicit form, we have

$$\text{MHSA}(\boldsymbol{X}) = \sum_{h=1}^{N_h} \boldsymbol{A}^h \boldsymbol{X} \underbrace{\boldsymbol{W}_{\text{val}}^h \boldsymbol{W}_{\text{out}}^h}_{\boldsymbol{W}^h \in \mathbb{R}^{D_{in}\times D_{out}}} \tag{5}$$

In the convolutional configuration of Eq. 4, $\boldsymbol{A}^h \boldsymbol{X}$ selects pixel h of $\boldsymbol{X}$. Hence, we need to set $\boldsymbol{W}^h = \boldsymbol{W}_{\text{filter}}^h$. However, as a product of matrices, the rank of $\boldsymbol{W}_h$ is bottlenecked by D_v. To avoid

this being a limitation, we need $D_v \geq D_{in}$ (since $D_{out} \geq D_{in}$). To achieve this with a minimal number of parameters, we choose $D_v = D_{in}$, and simply set the following initialization:

$$\boldsymbol{W}^h_{\text{val}} = \boldsymbol{I}, \qquad \boldsymbol{W}^h_{\text{out}} = \boldsymbol{W}^h_{\text{filter}}. \tag{6}$$

Note that this differs from the usual choice made in SA layers, where $D_v = \lfloor D_{in}/N_h \rfloor$. However, to keep the parameter count the same, we share the same $\boldsymbol{W}^h_{val}$ across different heads h, since it plays a symmetric role at initialization.

Note that this reparameterization introduces three additional matrices compared to the convolutional filter: $\boldsymbol{W}_{qry}, \boldsymbol{W}_{key}, \boldsymbol{W}_{val}$, each containing $D_{in} \times D_{in}$ parameters. However, since the convolutional filter contains $N_h \times D_{in} \times D_{out}$ parameters, where we typically have $N_h = 9$ and $D_{out} \in \{D_{in}, 2D_{in}\}$, these additional matrices are much smaller than the filters and hardly increase the parameter count. This can be seen from the model sizes in Tab. 2.

Gated Positional self-attention Recent work (d'Ascoli et al., 2021) has highlighted an issue with standard PSA: the fact that the content and positional terms in Eq. 3 are potentially of very different magnitudes, in which case the softmax ignores the smallest of the two. This can typically lead the PSA to adopt a greedy attitude: choosing the form of attention (content or positional) which is easiest at a given time then sticking to it.

To avoid this, the authors suggest to sum the content and positional terms *after* the softmax, with their relative importances governed by a learnable *gating* parameter λ_h (one for each attention head). The resulting Gated Positional Self-Attention (GPSA) layers are parametrized as follows:

$$\boldsymbol{A}^h_{ij} := (1 - \sigma(\lambda_h))\,\text{softmax}\left(\boldsymbol{Q}^h_i \boldsymbol{K}^{h\top}_j\right) + \sigma(\lambda_h)\,\text{softmax}\left(\boldsymbol{v}^{h\top}_{pos}\boldsymbol{r}_{ij}\right), \tag{7}$$

where $\sigma : x \mapsto {}^1\!/\!{}_{(1+e^{-x})}$ is the sigmoid function. In the positional part, the encodings $\boldsymbol{r}_{ij}$ are fixed rather than learnt (see Eq. 4), which makes changing input resolution straightforward (see SM. D) and leaves only 3 learnable parameters per head: $\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2$ and α[1].

How convolutional should the initialization be? The convolutional initialization of GPSA layers involves two parameters, determining how strictly convolutional the behavior is: the initial value of the *locality strength* α, which determines how focused each attention head is on its dedicated pixel, and the initial value of the *gating parameters* λ, which determines the importance of the positional information versus content. If $\lambda_h \gg 0$ and $\alpha \gg 1$, the T-CNN will perfectly reproduce the input-output function of the CNN, but may want to greedily stay in the convolutional configuration. Conversely, if $\lambda_h \ll 0$ and $\alpha \ll 1$, the T-CNN will forget about the input-output function of the CNN. Hence, we choose $\alpha = 1$ and $\lambda = 1$ to lie in between these two extremes, encouraging the T-CNNs to escape locality throughout training.

Architectural details To make our setup as canonical as possible, we focus on ResNet architectures (He et al., 2016), which contain 5 stages, with spatial resolution halfed and number of channels doubled at each stage. Our method involves reparameterizing 3×3 convolutions as GPSA layers with 9 attention heads. However, global SA is too costly in the first layers, where the spatial resolution is large. We therefore only reparameterize the last stage of the architecture, while replacing the first stride-2 convolution by a stride-1 convolution, exactly as in (Srinivas et al., 2021). We also add explicit padding layers to account for the padding of the original convolutions.

3 Performance of the Transformed CNNs

In this section, we apply our reparametrization to state-of-the-art CNNs, then fine-tune the resulting T-CNNs to learn better representations. This method allows to fully disentangle the training of the SA layers from that of the convolutional backbone, which is of practical interest for two reasons. First, it minimizes the time spent training the SA layers, which typically have a slower throughput. Second, it separates the algorithmic choices of the CNN backbone from those of the SA layers, which are typically different; for example, CNNs are typically trained with SGD whereas SA layers perform

[1] Since α represents the temperature of the softmax, its value must stay positive at all times. To ensure this, we instead learn a rectified parameter $\tilde{\alpha}$ using the softplus function: $\alpha = \frac{1}{\beta}\log(1 + e^{-\beta\tilde{\alpha}})$, with $\beta = 5$.

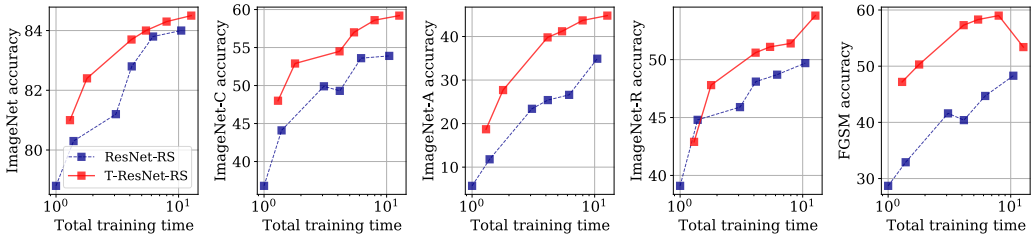


Figure 2: **T-CNNs reach much higher performance and robustness at equal training time.** Total training time (original training + finetuning) is normalized by the total training time of the ResNet50-RS.

much better with adaptive optimizers such as Adam (Zhang et al., 2019), an incompatibility which may limit the performance of usual hybrid models.

Note that our reparametrization can also be applied halfway through training: this scenario is investigated in SM. B. Results suggest that reparametrizing at intermediate times is optimal in terms of speed-performance trade-offs. However, to achieve solid baselines, and for easier reproducibility, we focus on reparametrizing the fully pre-trained CNNs available in the `timm` package (Wightman, 2019).

Training details We applied our method to pre-trained ResNet-RS (Bello et al., 2021) models, using the weights provided by the `timm` package (Wightman, 2019). These models are derived from the original ResNet (He et al., 2016), but use improved architectural features and training strategies, enabling them to reach better speed-accuracy trade-offs than EfficientNets.

To minimize computational cost, we restrict our fine-tuning to 50 epochs[2]. Following (Zhang et al., 2019), we use the AdamW optimizer, with a batch size of 1024[3]. The learning rate is warmed up to 10^{-4} then annealed using a cosine schedule. To encourage the T-CNN to escape the convolutional configuration and learn content-based attention, we use a larger learning rate of 0.1 for the gating parameters of Eq. 7 (one could equivalently decrease the temperature of the sigmoid function).

We use the same data augmentation scheme as the DeiT (Touvron et al., 2020), as well as rather large stochastic depth coefficients d_r reported in Tab. 1. Hoping that our method could be used as an alternative to the commonly used practice of fine-tuning models at higher resolution, we also increase the resolution during fine-tuning (Touvron et al., 2019). In this setting, a ResNet50 requires only 6 hours of fine-tuning on 16 V100 GPUs, compared to 33 hours for the original training. For our largest model (ResNet350-RS), the fine-tuning lasts 50 hours.

Performance improvements Results are presented in Tab. 1, where we also report the baseline improvement of fine-tuning in the same setting but without SA. In all cases, our fine-tuning improves top-1 accuracy, with a significant gap over the baseline. To demonstrate the wide applicability of our method, we report similar improvements for ResNet-D architectures in SM. F.

Despite the extra fine-tuning epochs and their slower throughput, the resulting T-CNNs also outperforming the original CNNs on the ImageNet validation set at equal training budget, as shown in the leftmost panel of Fig. 2[4]. However, the major benefit of the reparametrization is in terms of robustness, as shown in Fig. 2(b) and explained below.

Robustness improvements Recent work has shown that Transformer-based architectures offer better robustness and out-of-domain generalization than convolutional architectures (Bhojanapalli et al., 2021; Mao et al., 2021; Mahmood et al., 2021; Shao et al., 2021). To investigate whether our

[2]We study how performance depends on the number of fine-tuning epochs in SM. E.

[3]Confirming the results of (Zhang et al., 2019), we obtained worse results with SGD.

[4]We estimated the training times of the original ResNet-RS models based on their throughput, for the same hardware as used for the T-ResNet-RS.

Backbone	Training				Fine-tuning					
							Without SA		With SA	
	Res.	d_r	TTT	Top-1	Res.	d_r	TTT	Top-1	TTT	Top-1
ResNet50-RS	160	0.0	1 (ref.)	78.8	224	0.1	1.16	80.4	1.30	**81.0**
ResNet101-RS	192	0.0	1.39	80.3	224	0.1	1.65	81.9	1.79	**82.4**
ResNet152-RS	256	0.0	3.08	81.2	320	0.2	3.75	83.4	4.13	**83.7**
ResNet200-RS	256	0.1	4.15	82.8	320	0.2	5.04	83.7	5.42	**84.0**
ResNet270-RS	256	0.1	6.19	83.8	320	0.2	7.49	83.9	7.98	**84.3**
ResNet350-RS	288	0.1	10.49	84.0	320	0.2	12.17	84.1	12.69	**84.5**

Table 1: **Statistics of the models considered, trained from scratch on ImageNet.** Top-1 accuracy is measured on ImageNet-1k validation set. "TTT" stands for total training time (including fine-tuning), normalized by the total training time of the ResNet50-RS. d_r is the stochastic depth coefficient used for the various models.

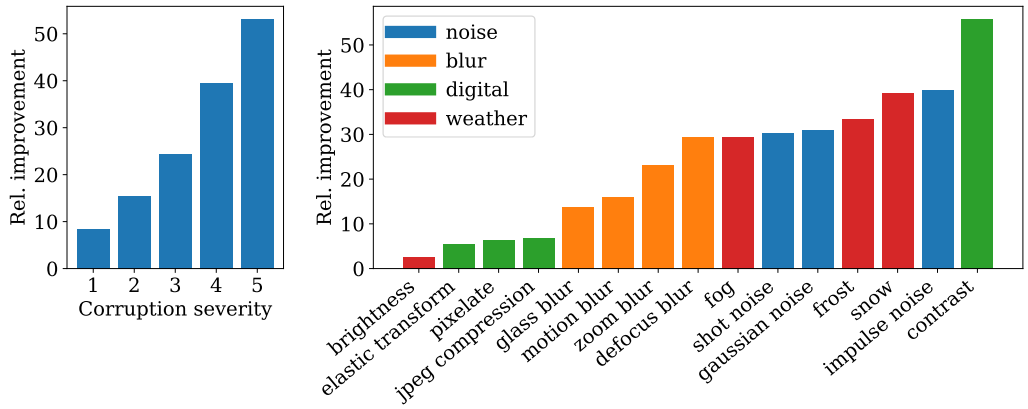


Figure 3: **Robustness is most improved for strong and blurry corruption categories.** We report the relative improvement between the top-1 accuracy of the T-ResNet50-RS and that of the ResNet50-RS on ImageNet-C, averaging over the different corruption categories (left) and corruption severities (right).

fine-tuning procedure is enough to imbue CNNs with these advantages, we evaluate our T-CNNs on various benchmarks:

- **Common corruptions**: we use ImageNet-C (Hendrycks & Dietterich, 2019), a dataset containing 15 sets of randomly generated corruptions, grouped into 4 categories: 'noise', 'blur', 'weather', and 'digital'. Each corruption type has five levels of severity, resulting in 75 distinct corruptions. Note that to avoid distorting the corruptions, which are often pixel-based, we keep a resolution of 224 at inference, which disadvantages the large models trained at higher resolutions.
- **Adversarial robustness**: following (Mao et al., 2021), we evaluate the accuracy of our models under two white-bow attacks[5]: (i) single-step FGSM (Goodfellow et al., 2014) and (ii) multi-step L_∞-PGD (Madry et al., 2017) with $t = 5$ steps of size $\alpha = 0.5$. Both attackers perturb the input image with max magnitude $\epsilon = 1$. We also evaluate our models on ImageNet-A (Hendrycks et al., 2021), a dataset containing naturally "adversarial" examples from ImageNet. Note however that since this dataset is built from the flaws of a ResNet, it is potential unfair to CNNs.
- **Distribution shifts**: we use ImageNet-R (Hendrycks et al., 2020), a dataset with various stylized "renditions" of ImageNet images ranging from paintings to embroidery, which strongly modify the local image statistics.

[5]We use the toolkit provided by https://github.com/bethgelab/foolbox.

The full table of results is presented in Tab. 2 of SM. A, and illustrated in Figs. 1 and 2. The T-ResNet-RS substantially outperforms the ResNet-RS on all robustness benchmarks. For example, our T-ResNet101-RS, which is 50% faster than ResNet200-RS, reaches similar or better results all robustness tasks, despite its lower top-1 accuracy on ImageNet-1k. This demonstrates that SA improves robustness more than it improves classification accuracy. The most striking improvement is in terms of adversarial robustness, where the smallest T-ResNet-RS is on par with the largest ResNet-RS despite requiring 5 times less compute.

To better understand where the benefits come from, we decompose the improvement of the T-ResNet50-RS over the various corruption severeties and categories of ImageNet-C in Fig. 3. We observe that improvement increases almost linearly with corruption severity. Although performance is higher in all corruption categories, there is a strong variability: the T-CNN shines particularly in tasks where the objects in the image are less sharp due to lack of contrast, bad weather or blurriness. We attribute this to the ability of SA to distinguish shapes in the image, as investigated in Sec 4.

4 Dissecting the Transformed CNNs

In this section, we analyze various observables to understand how the representations of a T-ResNet270-RS evolve from those of the ResNet270-RS throughout training.

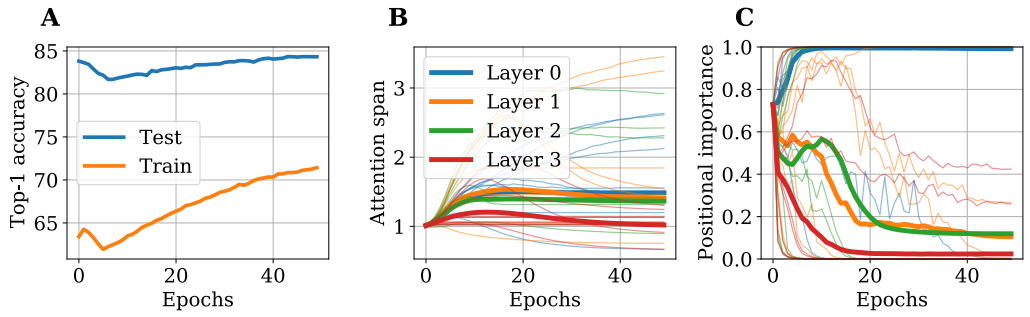
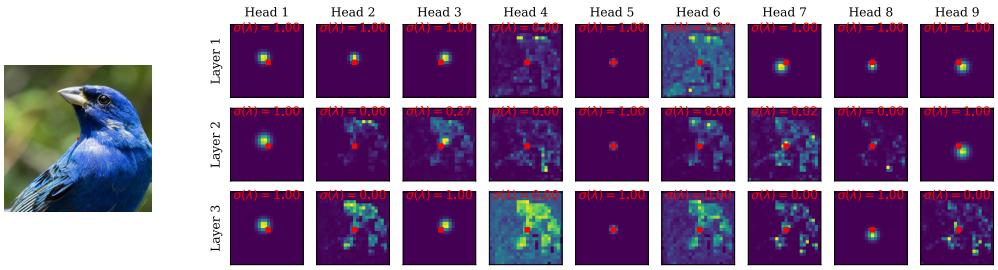


Figure 4: **The later layers effectively escape the convolutional configuration. A:** top-1 accuracy throughout the 50 epochs of fine-tuning of a T-ResNet270-RS. **B:** size of the receptive field of the various heads h (thin lines), calculated as α_h^{-1} (see Eq. 3). Thick lines represent the average over the heads. **C:** depicts how much attention the various heads h (thin lines) pay to positional information, through the value of $\sigma(\lambda_h)$ (see Eq. 7). Thick lines represent the average over the heads.



(a) Input image (b) Attention maps

Figure 5: **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). In each layer, at least one of the heads learns to perform content-based attention ($\sigma(\lambda_h) = 0$).

Unlearn to better relearn In Fig. 4A, we display the train and test accuracy throughout training[6]. The dynamics decompose into two distinct phases: accuracy dips down during the learning rate warmup phase (first 5 epochs of training), then increases back up as the learning rate is decayed.

As shown in SM. C, the depth of the dip depends on the learning rate. For too small learning rates, the dip is small, but the test accuracy increases too slowly after the dip; for too large learning rates, the test accuracy increases rapidly after the dip, but the dip is too deep to be compensated for. This suggests that the T-CNN needs to "unlearn" to some extent, a phenomenon reminiscent of the "catapult" mechanism of Lewkowycz et al. (2020) which propels models out of sharp minima to land in wider minima.

Escaping the convolutional representation In Fig. 4B, we show the evolution of the "attention span" $1/\alpha_h$ (see Eq. 4), which reflects the size of the receptive field of attention head h. On average (thick lines), this quantity increases in the first three layers, showing that the attention span widens, but variability exists among different attention heads (thin lines): some broaden their receptive field, whereas others contract it.

In Fig. 4C, we show the evolution of the gating parameters λ^h of Eq. 7, which reflect how much attention head h pays to position versus content. Interestingly, the first layer stays strongly convolutional on average, as $\mathbb{E}_h \sigma(\lambda_h)$ rapidly becomes close to one (thick blue line). The other layers strongly escape locality, with most attention heads focusing on content information at the end of fine-tuning.

In Fig. 5, we display the attention maps after fine-tuning. A clear divide appears between the "convolutional" attention heads, which remain close to their initialization, and the "content-based" attention heads, which learn more complex dependencies. Notice that the attention head initially focusing on the query pixel (head 5) stays convolutional in all layers. Throughout the layers, the edges of the central object is more and more clearly visible, as observed in (Caron et al., 2021). This supports the hypothesis that robustness gains obtained for blurry corruptions (see Fig. 3) are partly due to the ability of the SA layers to isolate objects from the background.

DISCUSSION

In this work, we showed that complex building blocks such as self-attention layers need not be trained from start. Instead, one can save in compute time while gaining in performance and robustness by initializing them from pre-trained convolutional layers. At a time where energy savings and robustness are key stakes, we believe this finding is important.

On the practical side, our fine-tuning method offers an interesting new direction for practitioners. One limitation of our method is the prohibitive cost of reparametrizing the early stages of CNNs. This cost could however be alleviated by using linear attention methods (Wang et al., 2020), an important direction for future work. Note also that while our T-CNNs significantly improve the robustness of CNNs, they do not always reach the performance of end-to-end Transformers such as the DeiT (for example on ImageNet-C, see Fig. 1). Bridging this gap is an important next step for hybrid models.

[6]The train accuracy is lower than the test accuracy due to the heavy data augmentation used during fine-tuning.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Irwan Bello. Lambdanetworks: Modeling long-range interactions without attention. *arXiv preprint arXiv:2102.08602*, 2021.

Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3286–3295, 2019.

Irwan Bello, William Fedus, Xianzhi Du, Ekin D Cubuk, Aravind Srinivas, Tsung-Yi Lin, Jonathon Shlens, and Barret Zoph. Revisiting resnets: Improved training and scaling strategies. *arXiv preprint arXiv:2103.07579*, 2021.

Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. *arXiv preprint arXiv:2103.14586*, 2021.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision*, pp. 104–120. Springer, 2020.

Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A2-nets: Double attention networks. *arXiv preprint arXiv:1810.11579*, 2018.

Zhengsu Chen, Lingxi Xie, Jianwei Niu, Xuefeng Liu, Longhui Wei, and Qi Tian. Visformer: The vision-friendly transformer, 2021.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. *arXiv preprint arXiv:1911.03584*, 2019.

Stéphane d'Ascoli, Levent Sagun, Giulio Biroli, and Joan Bruna. Finding the needle in the haystack with convolutions: on the benefits of architectural bias. In *Advances in Neural Information Processing Systems*, pp. 9334–9345, 2019.

Stéphane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. *arXiv preprint arXiv:2103.10697*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. *arXiv preprint arXiv:2104.01136*, 2021.

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

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 558–567, 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021.

Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv preprint arXiv:2103.16302*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3588–3597, 2018.

Zihang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Xiaojie Jin, Anran Wang, and Jiashi Feng. Token labeling: Training a 85.4% top-1 accuracy vision transformer with 56m parameters on imagenet, 2021.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pp. 5156–5165. PMLR, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. Understanding the difficulty of training transformers. *arXiv preprint arXiv:2004.08249*, 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. *arXiv preprint arXiv:2103.14030*, 2021.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. *arXiv preprint arXiv:2104.02610*, 2021.

Xiaofeng Mao, Gege Qi, Yuefeng Chen, Xiaodan Li, Shaokai Ye, Yuan He, and Hui Xue. Rethinking the design principles of robust vision transformer. *arXiv preprint arXiv:2105.07926*, 2021.

Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019.

Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. In Konstantinos Diamantaras, Wlodek Duch, and Lazaros S. Iliadis (eds.), *Artificial Neural Networks – ICANN 2010*, Lecture Notes in Computer Science, pp. 92–101, Berlin, Heidelberg, 2010. Springer. ISBN 978-3-642-15825-4. doi: 10.1007/978-3-642-15825-4_10.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61: 85–117, January 2015. ISSN 0893-6080. doi: 10.1016/j.neunet.2014.09.003. URL http://www.sciencedirect.com/science/article/pii/S0893608014002135.

Rulin Shao, Zhouxing Shi, Jinfeng Yi, Pin-Yu Chen, and Cho-Jui Hsieh. On the adversarial robustness of visual transformers. *arXiv preprint arXiv:2103.15670*, 2021.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck Transformers for Visual Recognition. *arXiv e-prints*, art. arXiv:2101.11605, January 2021.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7464–7473, 2019.

Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. *arXiv preprint arXiv:1906.06423*, 2019.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and H Linformer Ma. Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Wei Wen, Feng Yan, Yiran Chen, and Hai Li. Autogrow: Automatic layer growing in deep convolutional networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 833–841, 2020.

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

Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual Transformers: Token-based Image Representation and Processing for Computer Vision. *arXiv:2006.03677 [cs, eess]*, July 2020. URL http://arxiv.org/abs/2006.03677. arXiv: 2006.03677.

Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *arXiv preprint arXiv:2103.11816*, 2021.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank J Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *arXiv preprint arXiv:1912.03194*, 2019.

Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10076–10085, 2020.