https://arxiv.org/pdf/2103.17239.pdf

Going deeper with Image Transformers

Abstract the interplay of architecture and optimization of such dedicated transformers

two transformers architecture changes
leads us to produce models whose performance does not saturate early with more depth

Introduction we add a learnable diagonal matrix on output of each residual block, initialized close to (but not at) 0
class-attention layers

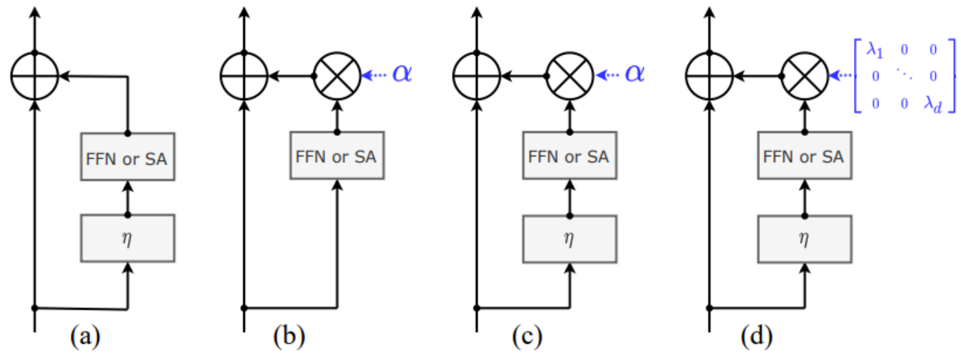
LayerScale facilitates the convergence and improves the accuracy of image transformers at larger depths

Our architecture with specific class-attention offers a more effective processing of the class embedding

Deeper image transformers with LayerScale
goal = increase the stability of the optimization
when training transformers for image classification
when we increase their depth.

vision transformer
,+ data-efficient image transformer (DeiT) optimization procedure

the deeper ViT architectures have a low performance



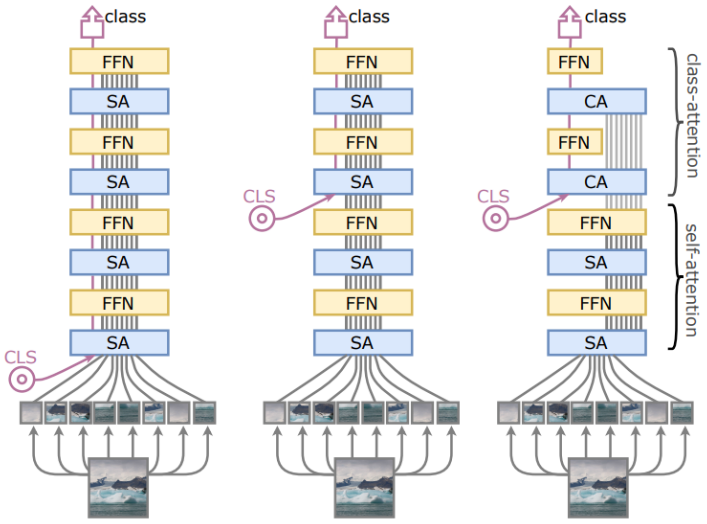
removing the warmup and the layernormalization is what makes training unstable
(c)

Our proposal LayerScale

(d)

$$x'_l = x_l + \mathrm{diag}(\lambda_{l,1},\ldots,\lambda_{l,d}) \times \mathrm{SA}(\eta(x_l))$$
$$x_{l+1} = x'_l + \mathrm{diag}(\lambda'_{l,1},\ldots,\lambda'_{l,d}) \times \mathrm{FFN}(\eta(x'_l)),$$

Specializing layers for class attention



learned weights are asked to optimize two contradictory objectives

(1) guiding the self-attention between patches while (2) summarizing the information useful to the linear classifier.

Later class token.

we insert the so-called class token, denoted by CLS, later in the transformer

eliminates the discrepancy on the first layers of the transformer,

average pooling of all the patches on output of the transformers, as typically employed in convolutional architectures

Architecture

two distinct processing stages visible in Figure 2

1. The self-attention stage is identical to the ViT transformer, but with no class embedding (CLS).
2. The class-attention stage is a set of layers that compiles the set of patch embeddings into a class embedding CLS that is subsequently fed to a linear classifier.

class-attention alternates in turn a layer that we refer to as a multi-head class-attention (CA), and a FFN layer

it is a learnable vector

we do no copy information from the class embedding to the patch embeddings
Only the class embedding is updated by residual in the CA and FFN processing of the class-attention stage

Multi-heads class attention.
role of the CA layer is to extract the information from the set of processed patches

it relies on the attention between
(i) the class embedding xclass (initialized at CLS in the first CA)
(ii) itself plus the set of frozen patch embeddings

h heads and p patches, and denoting by d the embedding size
parametrize the multi-head class-attention with several projection matrices

$$
\begin{aligned}
Q &= W_q \, x_{\text{class}} + b_q, \\
K &= W_k \, z + b_k, \\
V &= W_v \, z + b_v.
\end{aligned}
$$

Complexity
CA is identical to SA in that respect, and we use the same parametrization for the FFNs
the FFN only processes matrix-vector multiplications.
CA function is also less expensive than SA in term of memory and computation because it computes the attention between the class vector and the set of patch embedding

Experiments

Experimental setting
timm

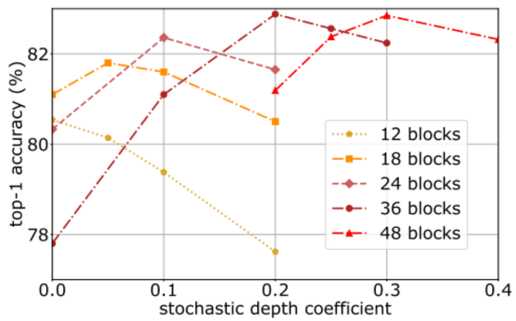
Preliminary analysis with deeper architectures

Vision Transformers become increasingly more difficult to train when we scale architectures.
Depth is one of the main source of instability
we analyse various ways to stabilize the training with different architectures
At this stage we consider a Deit-Small model3 during 300 epochs to allow a direct comparison with the results reports by Touvron et al

Adjusting the drop-rate of stochastic depth.

depth	baseline		scalar α weighting				LayerScale
	$d_r = 0.05$	adjust [d_r]	Rezero	T-Fixup	Fixup	$\alpha = \varepsilon$	
12	79.9	79.9 [*0.05*]	78.3	79.4	80.7	80.4	80.5
18	80.1	80.7 [*0.10*]	80.1	81.7	82.0	81.6	81.7
24	78.9†	81.0 [*0.20*]	80.8	81.5	82.3	81.1	82.4

36	78.9†	81.9 [0.25]	81.6	82.1	82.4	81.6	82.9



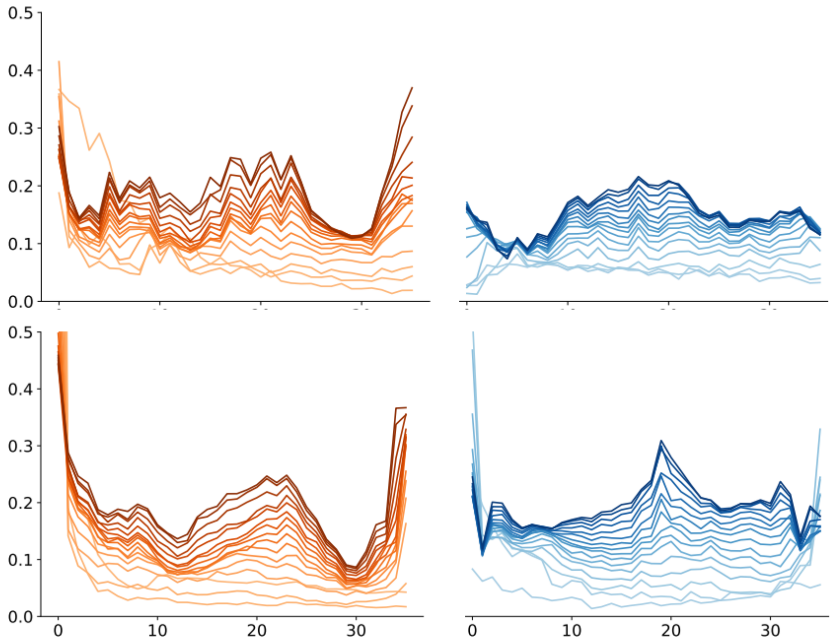
Comparison of normalization strategies

we only use LayerScale in subsequent experiments

It is much simpler and parametrized by a single hyper-parameter ε, and it offers a better performance for the deepest models that we consider, which are also the more accurate

Analysis of Layerscale

Statistics of branch weighting



Class-attention layers

depth: SA+CA	insertion layer	top-1 acc.	#params	FLOPs
Baselines: DeiT-S and average pooling				
12: 12 + 0	0	79.9	22M	4.6B
12: 12 + 0	n/a	80.3	22M	4.6B
Late insertion of class embedding				
12: 12 + 0	2	80.0	22M	4.6B
12: 12 + 0	4	80.0	22M	4.6B
12: 12 + 0	8	80.0	22M	4.6B
12: 12 + 0	10	80.5	22M	4.6B
12: 12 + 0	11	80.3	22M	4.6B
DeiT-S with class-attention stage (SA+FFN)				
12: 9 + 3	9	79.6	22M	3.6B
12: 10 + 2	10	80.3	22M	4.0B
12: 11 + 1	11	80.6	22M	4.3B
13: 12 + 1	12	80.8	24M	4.7B
14: 12 + 2	12	80.8	26M	4.7B
15: 12 + 3	12	80.6	27M	4.8B

Late insertion

The performance increases when we insert the class embedding later in the transformer

it is best to keep 2 layers for compiling the patches embedding into the class embedding via class-attention

Our class-attention layers

there is no benefit in copying information from the class embedding back to the patch embeddings in the forward pass

the performance of CaiT with 10 SA and 2 CA layers is identical to average pooling and better than the DeiT-Small baseline with a lower number of FLOPs

Our CaiT models

Our CaiT models are built upon ViT

LayerScale in each residual block

two-stages architecture with class-attention layers

CAIT model	depth (SA+CA)	d	#params ($\times 10^6$)	FLOPs ($\times 10^9$) @224	FLOPs ($\times 10^9$) @384	Top-1 acc. (%): Imagenet1k-val @224	↑384	@224Υ	↑384Υ
XXS-24	24 + 2	192	12.0	2.5	9.5	77.6	80.4	78.4	80.9
XXS-36	36 + 2	192	17.3	3.8	14.2	79.1	81.8	79.7	82.2

XS-24	24 + 2	288	26.6	5.4	19.3	81.8	83.8	82.0	84.1
XS-36	36 + 2	288	38.6	8.1	28.8	82.6	84.3	82.9	84.8
S-24	24 + 2	384	46.9	9.4	32.2	82.7	84.3	83.5	85.1
S-36	36 + 2	384	68.2	13.9	48.0	83.3	85.0	84.0	85.4
S-48	48 + 2	384	89.5	18.6	63.8	83.5	85.1	83.9	85.3
M-24	24 + 2	768	185.9	36.0	116.1	83.4	84.5	84.7	85.8
M-36	36 + 2	768	270.9	53.7	173.3	83.8	84.9	85.1	86.1

The hyper-parameters

DeiT

CAIT model		XXS-24	XXS-36	XS-24	XS-36	S-24	S-36	S-48	M-24	M-36	M-48
hparams	d_r	0.05	0.1	0.05	0.1	0.1	0.2	0.3	0.2	0.3	0.4
	ε	10^{-5}	10^{-6}	10^{-5}	10^{-6}	10^{-5}	10^{-6}	10^{-6}	10^{-5}	10^{-6}	10^{-6}

we modify depending on the model complexity, namely the drop rate dr associated with uniform stochastic depth, and the initialization value ε of LayerScale

Fine-tuning at higher resolution (↑) and distillation (Υ).

Results

Performance/complexity of CaiT models

width and the depth, both contribute

if one parameter is too small the gain brought by increasing the other is not worth the additional complexity.

o comes with a higher computational cost

leveraging a pre-trained convnet teacher with hard distillation

provides a boost in accuracy without affecting the number of parameters nor the speed

Comparison with the state of the art on Imagenet

Network	nb of param.	nb of FLOPs	image size train	image size test	ImNet top-1	Real top-1	V2 top-1
RegNetY-16GF	84M	16.0B	224	224	82.9	88.1	72.4
EfficientNet-B5	30M	9.9B	456	456	83.6	88.3	73.6
EfficientNet-B7	66M	37.0B	600	600	84.3	-	-
EfficientNet-B5 RA	30M	9.9B	456	456	83.7	-	-
EfficientNet-B7 RA	66M	37.0B	600	600	84.7	-	-
EfficientNet-B7 AdvProp	66M	37.0B	600	600	85.2	89.4	76.0
Fix-EfficientNet-B8	87M	89.5B	672	800	85.7	90.0	75.9
NFNet-F0	72M	12.4B	192	256	83.6	88.1	72.6
NFNet-F1	133M	35.5B	224	320	84.7	88.9	74.4
NFNet-F2	194M	62.6B	256	352	85.1	88.9	74.3

NFNet-F2	194M	62.6B	256	352	85.1	88.9	74.3
NFNet-F3	255M	114.8B	320	416	85.7	89.4	75.2
NFNet-F4	316M	215.3B	384	512	85.9	89.4	75.2
NFNet-F5	377M	289.8B	416	544	86.0	89.2	74.6
NFNet-F6+SAM	438M	377.3B	448	576	**86.5**	89.9	75.8
Transformers							
ViT-B/16	86M	55.4B	24	384	77.9	83.6	-
ViT-L/16	307M	190.7B	224	384	76.5	82.2	-
T2T-ViT t-14	21M	5.2B	224	224	80.7	-	-
TNT-S	24M	5.2B	224	224	81.3	-	-
TNT-S + SE	25M	5.2B	224	224	81.6	-	-
TNT-B	66M	14.1B	224	224	82.8	-	-
DeiT-S	22M	4.6B	224	224	79.8	85.7	68.5
DeiT-B	86M	17.5B	224	224	81.8	86.7	71.5
DeiT-B↑384	86M	55.4B	224	384	83.1	87.7	72.4
DeiT-B↑384Υ 1000 epochs	87M	55.5B	224	384	85.2	89.3	75.2
Our deep transformers							
CaiT-S36	68M	13.9B	224	224	83.3	88.0	72.5
CaiT-S36↑384	68M	48.0B	224	384	85.0	89.2	75.0
CaiT-S48↑384	89M	63.8B	224	384	85.1	89.5	75.5
CaiT-S36Υ	68M	13.9B	224	224	84.0	88.9	74.1
CaiT-S36↑384Υ	68M	48.0B	224	384	85.4	89.8	76.2
CaiT-M36↑384Υ	271M	173.3B	224	384	86.1	<u>90.0</u>	76.3
CaiT-M36↑448Υ	271M	247.8B	224	448	<u>86.3</u>	**90.2**	<u>76.7</u>
CaiT-M48↑448Υ	356M	329.6B	224	448	**86.5**	**90.2**	**76.9**

Transfer learning

Dataset	Train size	Test size	#classes
ImageNet [54]	1,281,167	50,000	1000
iNaturalist 2018 [30]	437,513	24,426	8,142
iNaturalist 2019 [31]	265,240	3,003	1,010
Flowers-102 [46]	2,040	6,149	102
Stanford Cars [38]	8,144	8,041	196
CIFAR-100 [39]	50,000	10,000	100
CIFAR-10 [39]	50,000	10,000	10

Fine-tuning procedure.

Results

Model	ImageNet	CIFAR-10	CIFAR-100	Flowers	Cars	iNat-18	iNat-19	FLOPs
EfficientNet-B7	84.3	98.9	91.7	98.8	**94.7**	-	-	37.0B
ViT-B/16	77.9	98.1	87.1	89.5	-	-	-	55.5B

ViT-L/16	76.5	97.9	86.4	89.7	-	-	-	190.7B
Deit-B 224	81.8	99.1	90.8	98.4	92.1	73.2	77.7	17.5B
CaiT-S-36 224	83.4	99.2	92.2	98.8	93.5	77.1	80.6	13.9B
CaiT-M-36 224	83.7	99.3	93.3	99.0	93.5	76.9	81.7	53.7B
CaiT-S-36 Υ 224	83.7	99.2	92.2	99.0	94.1	77.0	81.4	13.9B
CaiT-M-36 Υ 224	**84.8**	**99.4**	**93.1**	**99.1**	94.2	**78.0**	**81.8**	53.7B

Ablation

Step by step from DeiT-Small to CaiT-S36

Improvement	top-1 acc.	#params	FLOPs
DeiT-S [*d=384,300 epochs*]	79.9	22M	4.6B
+ More heads [*8*]	80.0	22M	4.6B
+ Talking-heads	80.5	22M	4.6B
+ Depth [*36 blocks*]	69.9†	64M	13.8B
+ Layer-scale [*init* $\varepsilon = 10^{-6}$]	80.5	64M	13.8B
+ Stch depth. adaptation [d_r*=0.2*]	83.0	64M	13.8B
+ CaiT architecture [*specialized class-attention layers*]	83.2	68M	13.9B
+ Longer training [*400 epochs*]	83.4	68M	13.9B
+ Inference at higher resolution [*256*]	83.8	68M	18.6B
+ Fine-tuning at higher resolution [*384*]	84.8	68M	48.0B
+ Hard distillation [*teacher: RegNetY-16GF*]	85.2	68M	48.0B
+ Adjust crop ratio [$0.875 \rightarrow 1.0$]	85.4	68M	48.0B

the resolution is another important step for improving the performance and fine-tuning instead of training the model from scratch saves a lot of computation at training time Last but not least, our models benefit from longer training schedules.

Optimization of the number of heads

# heads	dim/head	throughput (im/s)	GFLOPs	top-1 acc.
1	384	1079	4.6	76.80
2	192	1056	4.6	78.06
3	128	1043	4.6	79.35
6	64	989	4.6	79.90
8	48	971	4.6	80.02
12	32	927	4.6	80.08
16	24	860	4.6	80.04
24	16	763	4.6	79.60

This architectural parameter has an impact on both the accuracy, and the efficiency

Adaptation of the crop-ratio

Table 10: We compare performance with the defaut crop-ratio of 0.875 usually used with convnets, and the crop-ratio of 1.0 [68] that we adopt for CaiT.

Network	Crop Ratio		ImNet	Real	V2
	0.875	1.0	top-1	top-1	top-1
S36	✓	-	83.4	88.1	73.0
	-	✓	83.3	88.0	72.5
S36↑384	✓	-	84.8	88.9	74.7
	-	✓	85.0	89.2	75.0
S36Υ	✓	-	83.7	88.9	74.1
	-	✓	84.0	88.9	74.1
M36Υ	✓	-	84.8	89.2	74.9
	-	✓	84.9	89.2	75.0
S36↑384Υ	✓	-	85.2	89.7	75.7
	-	✓	85.4	89.8	76.2
M36↑384Υ	✓	-	85.9	89.9	76.1
	-	✓	86.1	90.0	76.3

Longer training schedules

increasing the number of training epochs from 300 to 400 improves the performance of CaiT-S-36

increasing the number of training epochs from 400 to 500 does not change performance significantly

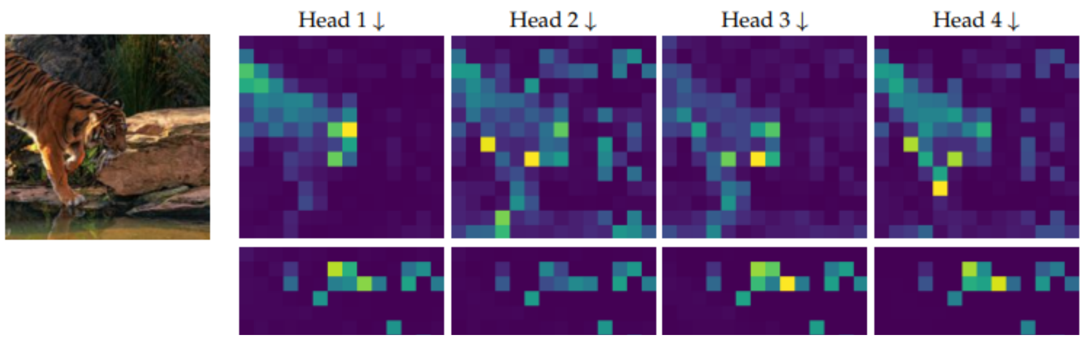
This is consistent with the observation of the DeiT [63] paper, which notes a saturation of performance from 400 epochs for the models trained without distillation.

Visualizations

Attention map

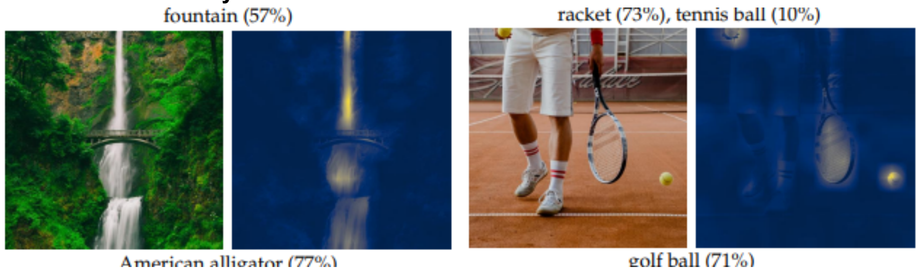
The first class-attention layer clearly focuses on the object of interest

the different heads focus either on the same or on complementary parts of the objects.



The second class-attention layer seems to focus more on the context, or at least the image more globally

Illustration of saliency in class-attention



Related work

Encoder/decoder architectures.

Deeper architectures

usually lead to better performance

complicates their training proces

Conclusion

how train deeper transformer-based image classification neural networks

simple yet effective CaiT architecture

transformer models offer a competitive alternative to the best convolutional neural networks when considering trade-offs between accuracy and complexity