MLP-Mixer: An all-MLP Architecture for Vision

arXiv 2021

21.05.31 Leeminsoo

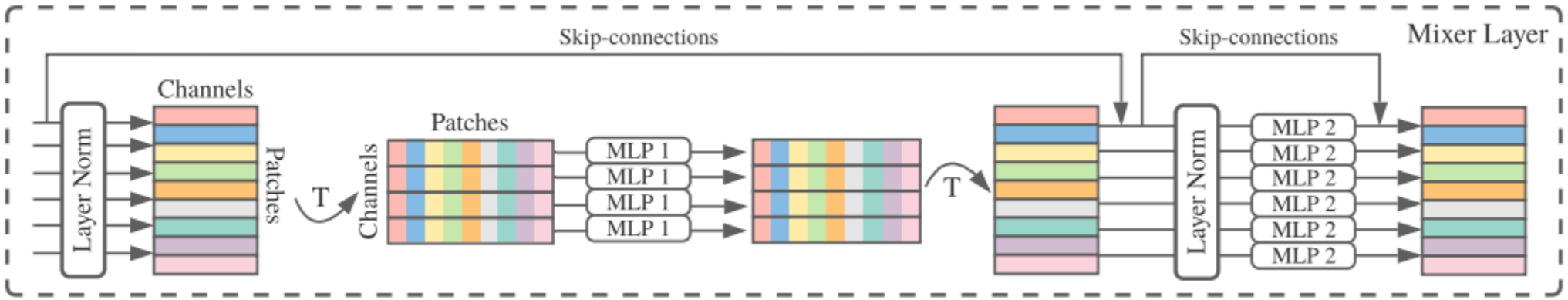
https://arxiv.org/abs/2105.01601
https://github.com/lucidrains/mlp-mixer-pytorch
https://www.youtube.com/watch?v=KQmZIxdnnuY

DAVIAN
Data and Visual Analytics Lab

Introduction

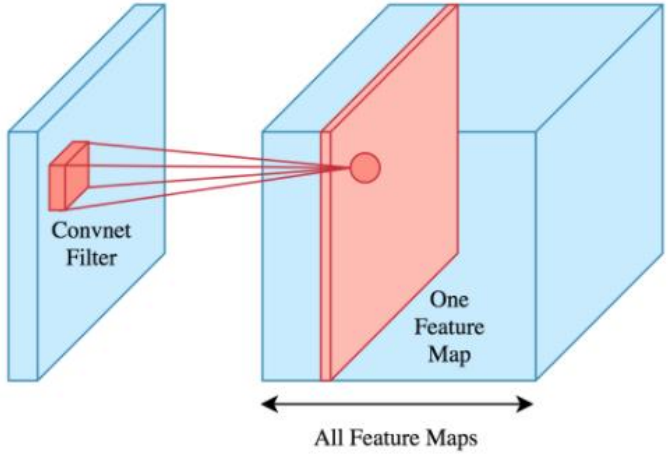
- As the history of computer vision demonstrates, the availability of larger datasets coupled with increased computational capacity often leads to a paradigm shift.
- Recently Vision Transformers (ViT), an alternative based on self-attention layers, attained state-of-the-art performance.
- ViT continues the long-lasting trend of removing hand-crafted visual features and inductive biases from models and relies further on learning from raw data.
- In this paper, the MLP-Mixer which is a competitive but conceptually and technically simple alternative, that does not use convolutions or self-attention is proposed.

Introduction

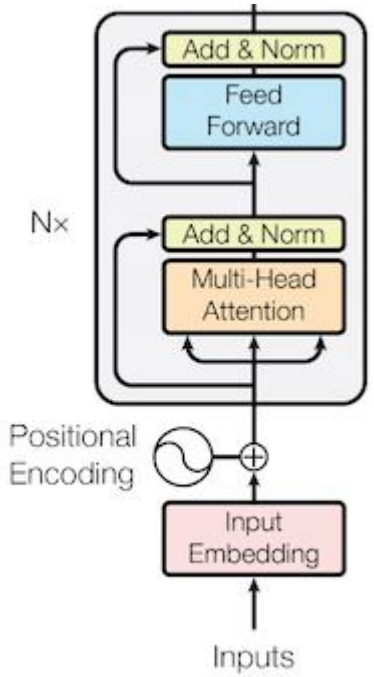


- Modern deep vision architectures consist of layers that mix features (1) at a given spatial location, (2) between different spatial locations, or both at once.
- The idea behind the Mixer architecture is to clearly separate the per-location (channel-mixing) operations and cross-location (token-mixing) operations.
- Both operations are implemented with MLPs.

Introduction



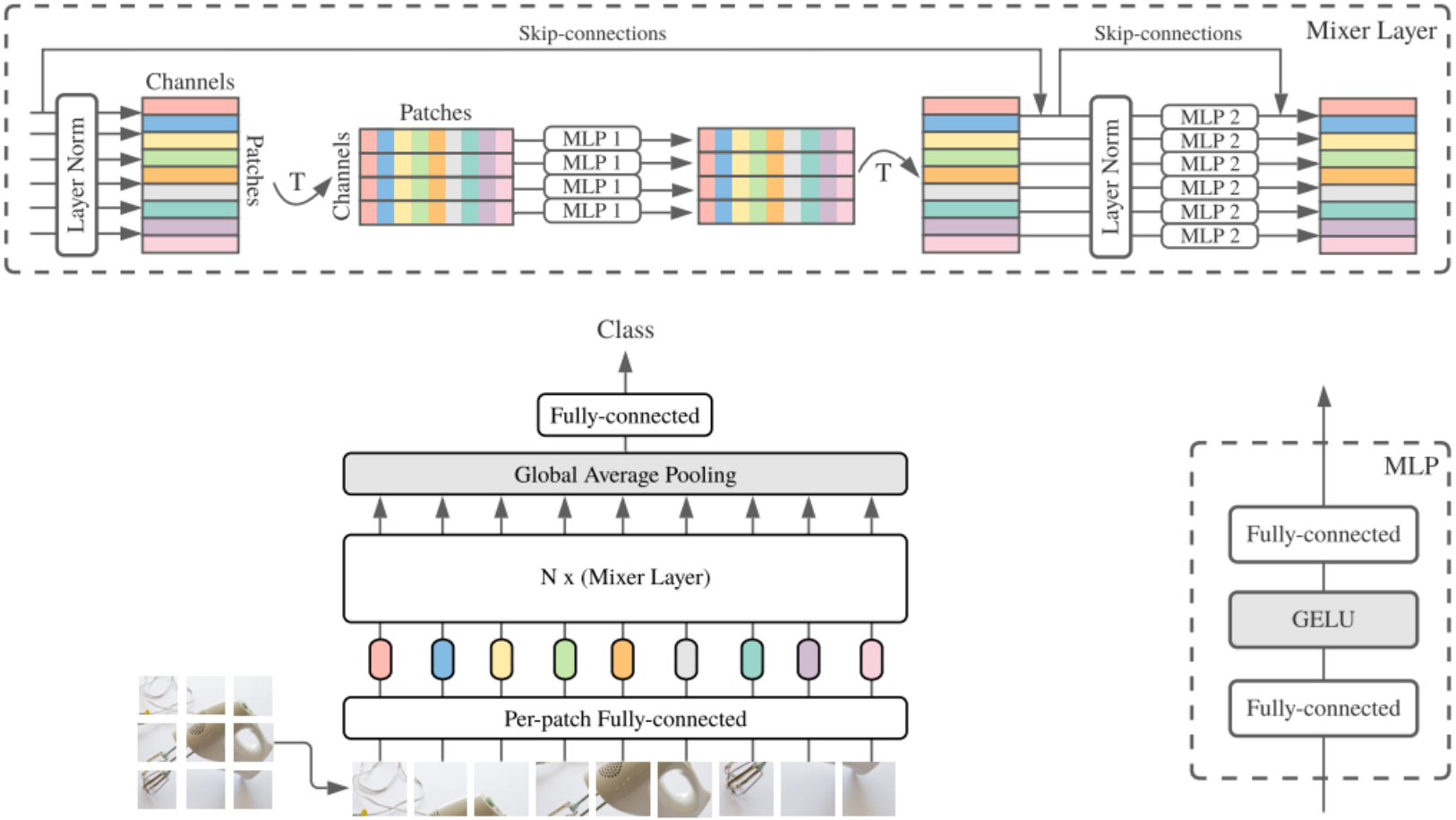
[Convolution Layer]



[Self-Attention Layer]

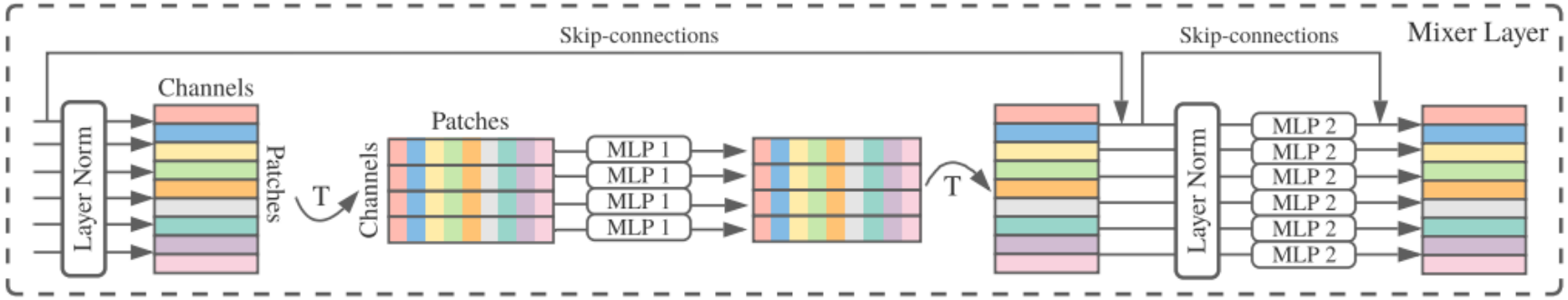
- Modern deep vision architectures consist of layers that mix features (1) at a given spatial location, (2) between different spatial locations, or both at once.
- In CNNS, 1x1 convolutions perform (1), and larger kernels perform both (1) and (2).
- In attention-based architectures, self-attention layers allow both (1) and (2) and the MLP-blocks perform (1).

Method



- Mixer takes as input a sequence of S non-overlapping image patches, each one projected to a desired hidden dimension C. This results in a two-dimensional input table, $\mathbf{X} \in \mathbb{R}^{S \times C}$.

Method



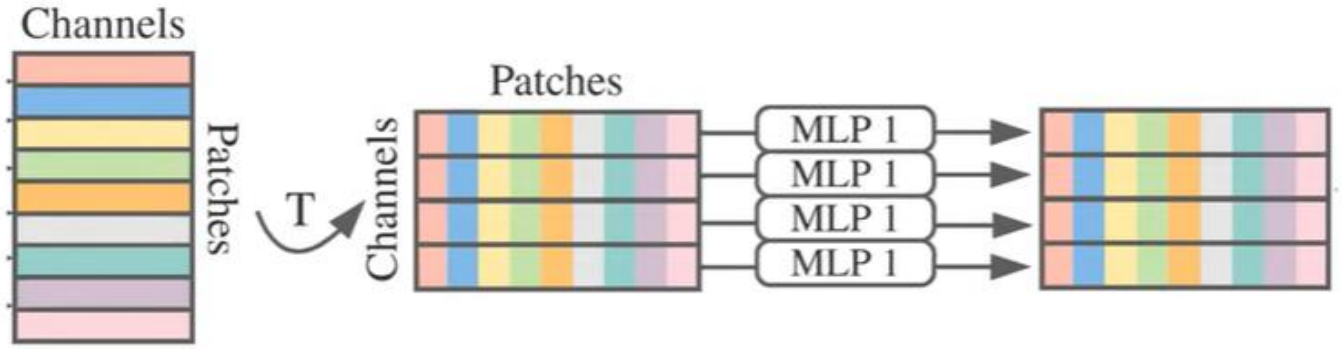
- The first one is the token-mixing MLP block: it acts on columns of $\mathbf{X}$, maps $\mathbb{R}^{S} \rightarrow \mathbb{R}^{S}$, and is shared across all columns.
- The second one is the channel-mixing MLP block: it acts on rows of $\mathbf{X}$, maps $\mathbb{R}^{C} \rightarrow \mathbb{R}^{C}$, and is shared across all rows.
- Mixer does not use positional embeddings because the token-mixing MLPs are sensitive to the order of the input tokens.
- Tunable hidden widths in the MLPs are independent from the number of input patches. Therefore, the computational complexity of the network is linear in the number of input patches, unlike ViT whose complexity is quadratic.

Method

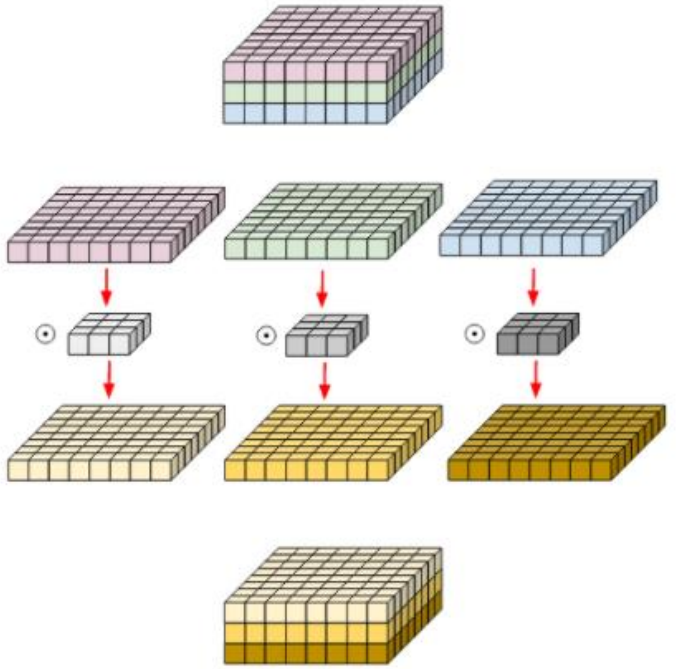
[Per-patch Fully-connected layer]

- Per-patch Fully-connected layer is the same as a $\mathrm{P} \times P$ convolution with stride P.

Method



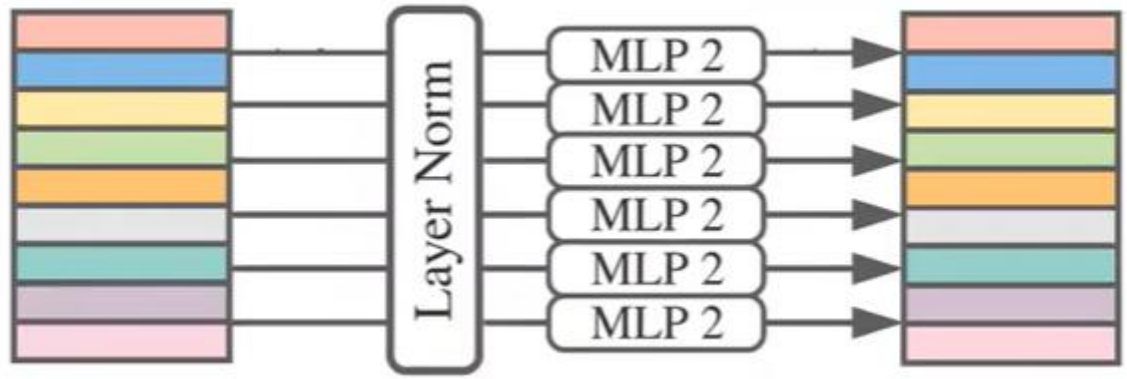
[Token-mixing MLP]



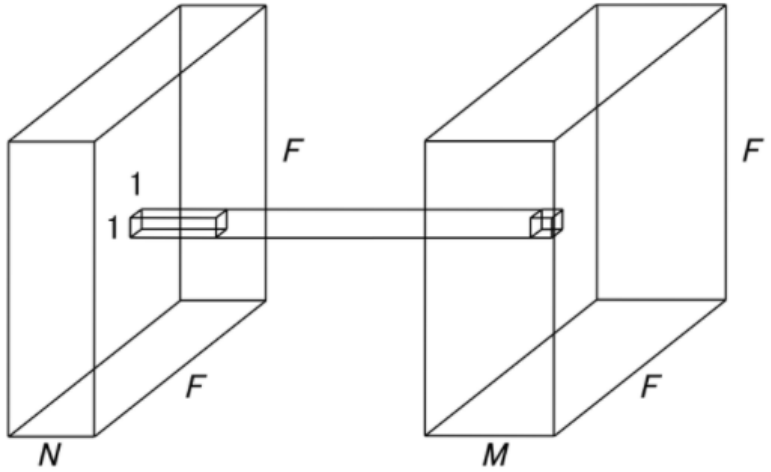
[Depth-wise Convolution]

- Token-mixing MLP is a $\sqrt{S} \times \sqrt{S}$ weight-shared depth-wise convolution.

Method



[Channel-mixing MLP]



[1 × 1 Convolution or
Point-wise Convolution]

- Channel-mixing MLP is a 1×1 convolution.

Experiments

Table 1: Specifications of the Mixer architectures used in this paper. The "B", "L", and "H" (base, large, and huge) model scales follow Dosovitskiy et al. [14]. We use a brief notation: "B/16" means the model of base scale with patches of resolution 16×16. "S" refers to a small scale with 8 Mixer layers. The number of parameters is reported for an input resolution of 224 and does not include the weights of the classifier head.

Specification	S/32	S/16	B/32	B/16	L/32	L/16	H/14
Number of layers	8	8	12	12	24	24	32
Patch resolution $P\times P$	32×32	16×16	32×32	16×16	32×32	16×16	14×14
Hidden size C	512	512	768	768	1024	1024	1280
Sequence length S	49	196	49	196	49	196	256
MLP dimension D_C	2048	2048	3072	3072	4096	4096	5120
MLP dimension D_S	256	256	384	384	512	512	640
Parameters (M)	19	18	60	59	206	207	431

Experiments

	Image size	Pre-Train Epochs	ImNet top-1	ReaL top-1	Avg. 5 top-1	Throughput (img/sec/core)	TPUv3 core-days
Pre-trained on ImageNet (with extra regularization)							
Mixer-B/16	224	300	76.44	82.36	88.33	1384	0.01k(‡)
ViT-B/16 (☎)	224	300	79.67	84.97	90.79	861	0.02k(‡)
Mixer-L/16	224	300	71.76	77.08	87.25	419	0.04k(‡)
ViT-L/16 (☎)	224	300	76.11	80.93	89.66	280	0.05k(‡)
Pre-trained on ImageNet-21k (with extra regularization)							
Mixer-B/16	224	300	80.64	85.80	92.50	1384	0.15k(‡)
ViT-B/16 (☎)	224	300	84.59	88.93	94.16	861	0.18k(‡)
Mixer-L/16	224	300	82.89	87.54	93.63	419	0.41k(‡)
ViT-L/16 (☎)	224	300	84.46	88.35	94.49	280	0.55k(‡)
Mixer-L/16	448	300	83.91	87.75	93.86	105	0.41k(‡)
Pre-trained on JFT-300M							
Mixer-S/32	224	5	68.70	75.83	87.13	11489	0.01k
Mixer-B/32	224	7	75.53	81.94	90.99	4208	0.05k
Mixer-S/16	224	5	73.83	80.60	89.50	3994	0.03k
BiT-R50x1	224	7	73.69	81.92	—	2159	0.08k
Mixer-B/16	224	7	80.00	85.56	92.60	1384	0.08k
Mixer-L/32	224	7	80.67	85.62	93.24	1314	0.12k
BiT-R152x1	224	7	79.12	86.12	—	932	0.14k
BiT-R50x2	224	7	78.92	86.06	—	890	0.14k
BiT-R152x2	224	14	83.34	88.90	—	356	0.58k
Mixer-L/16	224	7	84.05	88.14	94.51	419	0.23k
Mixer-L/16	224	14	84.82	88.48	94.77	419	0.45k
ViT-L/16	224	14	85.63	89.16	95.21	280	0.65k
Mixer-H/14	224	14	86.32	89.14	95.49	194	1.01k
BiT-R200x3	224	14	84.73	89.58	—	141	1.78k
Mixer-L/16	448	14	86.78	89.72	95.13	105	0.45k
ViT-H/14	224	14	86.65	89.56	95.57	87	2.30k
Mixer-H/14	448	14	87.78	90.08	95.62	40	1.01k
ViT-L/16(†) [14]	512	14	87.76	90.54	95.63	32	0.65k
BiT-R152x4 [22]	480	40	87.54	90.54	95.33	26	9.90k
ViT-H/14(†) [14]	518	14	88.55	90.72	95.97	15	2.30k

Table 3: Performance of Mixer and other models from the literature across various model and pre-training dataset scales. "Avg. 5" denotes the average performance across five downstream tasks and is presented where available. Mixer and ViT models are averaged over three fine-tuning runs and standard deviations are smaller than 0.15. (†) ViT models reported were fine-tuned with Polyak averaging [36]. (‡) Extrapolated from the numbers reported for the same models pre-trained on JFT-300M without extra regularization. (☎) Numbers provided by authors of Dosovitskiy et al. [14] through personal communication. Rows are sorted by throughput.

- MLP-based Mixer models
- Convolution-based models
- Attention-based models

- Mixer-B/16 attains a reasonable score of 76.4% at resolution 224, but it tends to overfit when using random initialization.
- This score is similar to a vanilla ResNet50, but behind state-of-the-art CNNs/hybrids for the ImageNet "from scratch" setting (BotNet : 84.7%, NFNet : 86.5%).
- When the size of the upstream dataset increases, Mixer's performance improves significantly. Mixer-H/14 pre-trained on JFT-300M and fine-tuned at 224 resolution is only 0.3% behind ViT-H/14 while running 2.2 times faster.

Experiments

Table 2: Transfer performance, inference throughput, and training cost. The rows are sorted by inference throughput (fifth column). Mixer has comparable transfer accuracy to state-of-the-art models with similar cost. The Mixer models are fine-tuned at resolution 448. Mixer performance numbers are averaged over three fine-tuning runs and standard deviations are smaller than 0.1.

	ImNet top-1	ReaL top-1	Avg 5 top-1	VTAB-1k 19 tasks	Throughput img/sec/core	TPUv3 core-days
Pre-trained on ImageNet-21k (public)						
● HaloNet [50]	85.8	—	—	—	120	0.10k
● Mixer-L/16	84.15	87.86	93.91	74.95	105	0.41k
● ViT-L/16 [14]	85.30	88.62	94.39	72.72	32	0.18k
● BiT-R152x4 [22]	85.39	—	94.04	70.64	26	0.94k
Pre-trained on JFT-300M (proprietary)						
● NFNet-F4+ [7]	89.2	—	—	—	46	1.86k
● Mixer-H/14	87.94	90.18	95.71	75.33	40	1.01k
● BiT-R152x4 [22]	87.54	90.54	95.33	76.29	26	9.90k
● ViT-H/14 [14]	88.55	90.72	95.97	77.63	15	2.30k
Pre-trained on unlabelled or weakly labelled data (proprietary)						
● MPL [34]	90.0	91.12	—	—	—	20.48k
● ALIGN [21]	88.64	—	—	79.99	15	14.82k

● MLP-based Mixer models
● Convolution-based models
● Attention-based models

- When pre-trained on ImageNet-21k, Mixer achieves an overall strong performance, although slightly inferior to other models.

Experiments

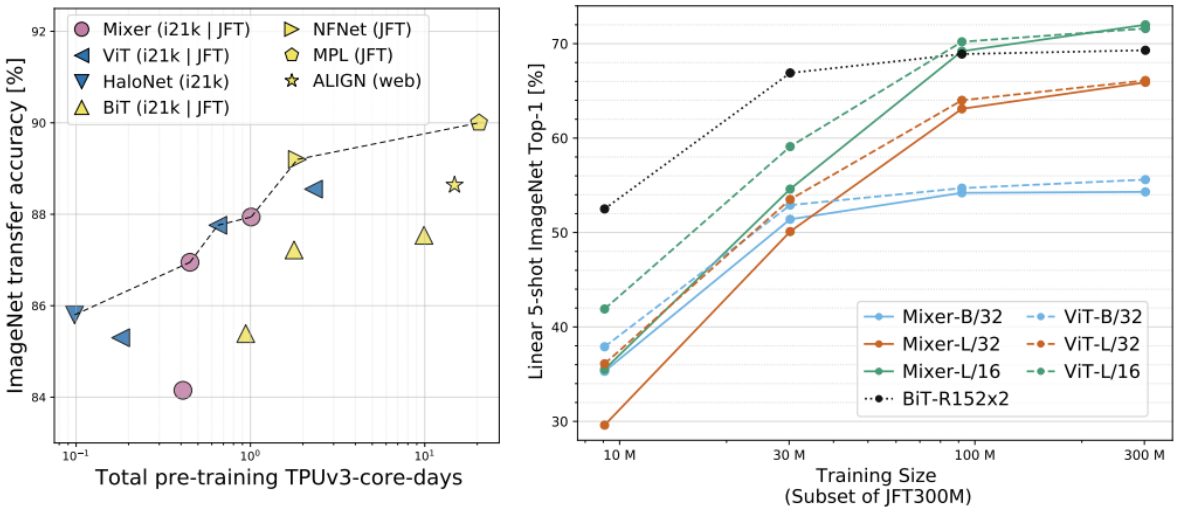


Figure 2: **Left:** ImageNet accuracy/training cost Pareto frontier (dashed line) for the SOTA models presented in Table 2. These model are pre-trained on ImageNet-21k, or JFT (labelled, or pseudo-labelled for MPL), or noisy web image text pairs. In addition, we include ViT-L/16, Mixer-L/16, and BiT-R200x3 (Adam) for context. Mixer is as good as these extremely performant ResNets, ViTs, and hybrid models, and sits on frontier with HaloNet, ViT, NFNet, and MPL. **Right:** Mixer (solid) catches or exceeds BiT (dotted) and ViT (dashed) as the data size grows. Every point on a curve uses the same pre-training compute; they correspond to pre-training on 3%, 10%, 30%, and 100% of JFT-300M for 233, 70, 23, and 7 epochs, respectively. Mixer improves more rapidly with data than ResNets, or even ViT, and the gap between large scale Mixer and ViT models shrinks until the performance is matched on the entire dataset.

- When pre-trained on the smallest subset of JFT-300M, all Mixer models strongly overfit. Bit models also overfit, but to a lesser extent, possibly due to the strong inductive biases associated with the convolutions.
- As the dataset increases, the performance of both Mixer and ViT grows faster than BiT.
- The performance gap between Mixer-L/16 and Vit-L/16 shrinks with data scale. It appears that Mixer models benefit from the growing pre-training dataset size even more than ViT.
- Perhaps, self-attention layers in ViT lead to certain properties of the learned functions that are less compatible with the true underlying distribution than those discovered with Mixer architecture.
(the difference in inductive biases)

Experiments

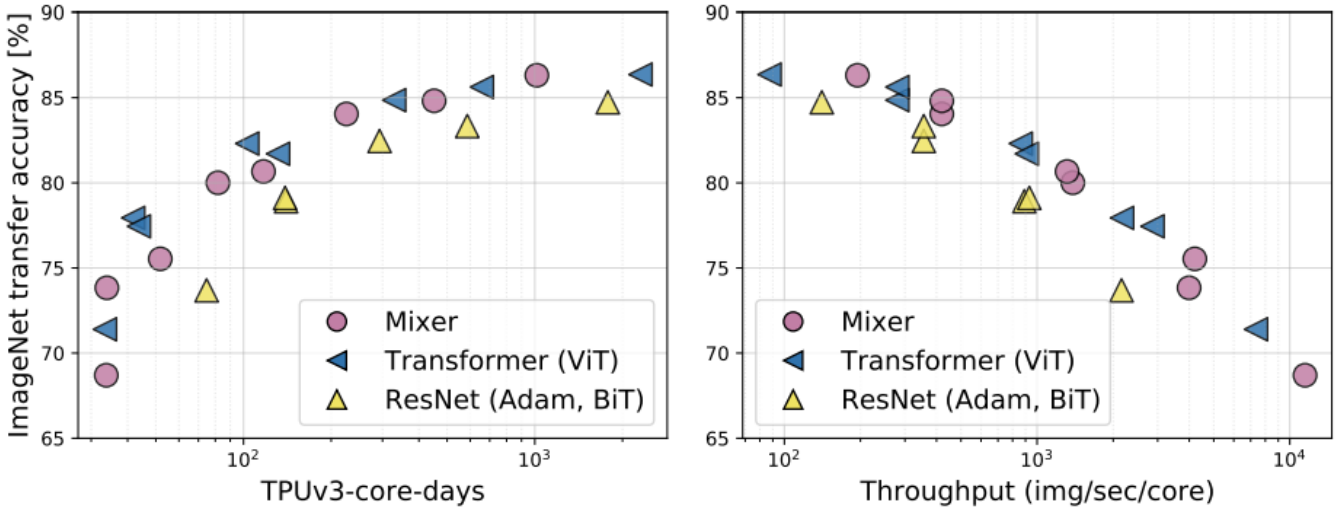


Figure 3: The role of the model scale. ImageNet validation top-1 accuracy vs. total pre-training compute (**left**) and throughput (**right**) of ViT, BiT, and Mixer models at various scales. All models are pre-trained on JFT-300M and fine-tuned at resolution 224, which is lower than in Figure 2 (left).

Experiments

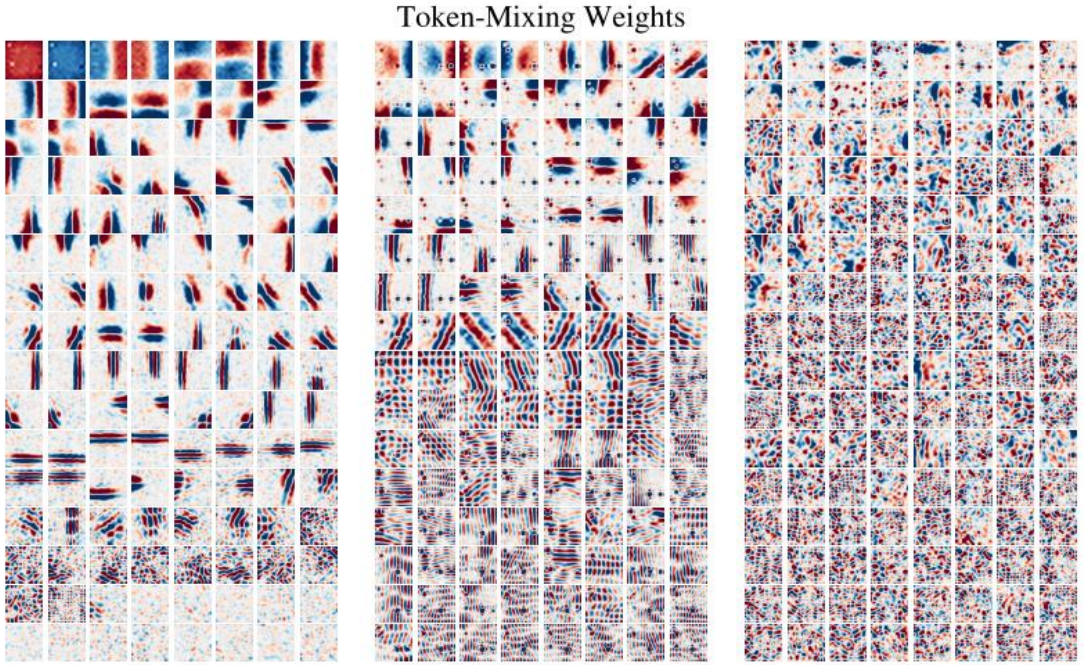


Figure 4: A selection of input weights to the hidden units in the first (**left**), second (**center**), and third (**right**) token-mixing MLPs of a Mixer-B/16 model trained on JFT-300M. Each unit has $14 \times 14 = 196$ weights, one for each of the 14×14 incoming patches. We pair units whose inverse is closest, to easily visualize the emergence of kernels of opposing phase. Pairs are sorted approximately by filter frequency. We highlight that in contrast to the kernels of convolutional filters, where each weight corresponds to one pixel in the input image, one weight in any plot from the left column corresponds to a particular 16×16 patch of the input image. Complete plots in Supplementary D.

EOD