Faculty of Computer Science
Applied Mathematics and
Information Science

2022

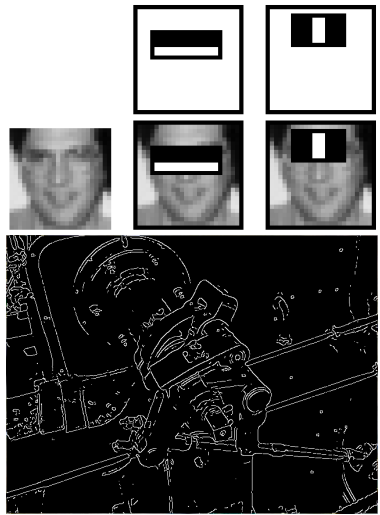
MLP-Mixer: An all-MLP Architecture for Vision

Mixing guide from top DJs

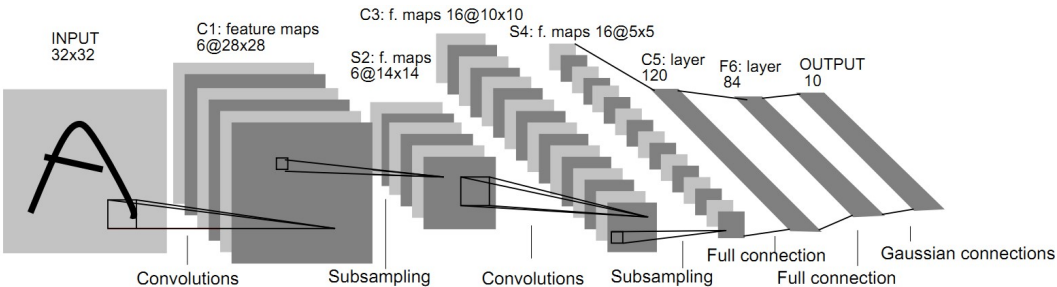
Daniil Volgin
devolgin@edu.hse.ru

Computer Vision
Feature engineering

- Haar features
- Sobel, Canny edge detection
- Laplacian, Gaussian filters
- SIFT, SURF, FAST
- Hough, Fourier, Wavelet transforms
- Geometric hashing

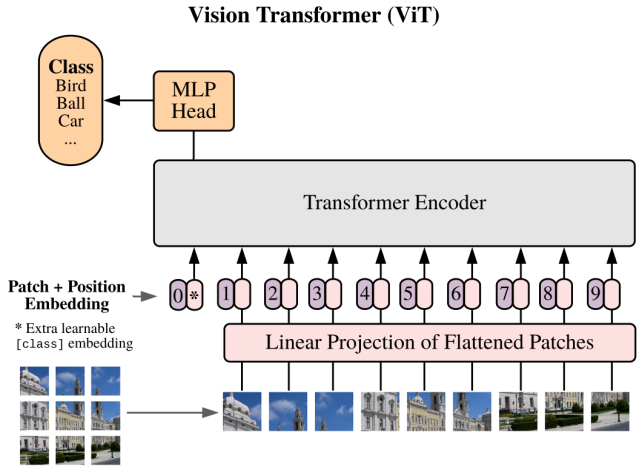


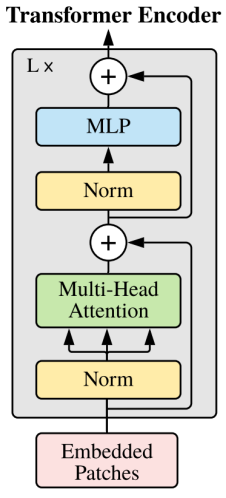
- (1989) LeNet
- (2012) AlexNet
- (2022) ConvNext



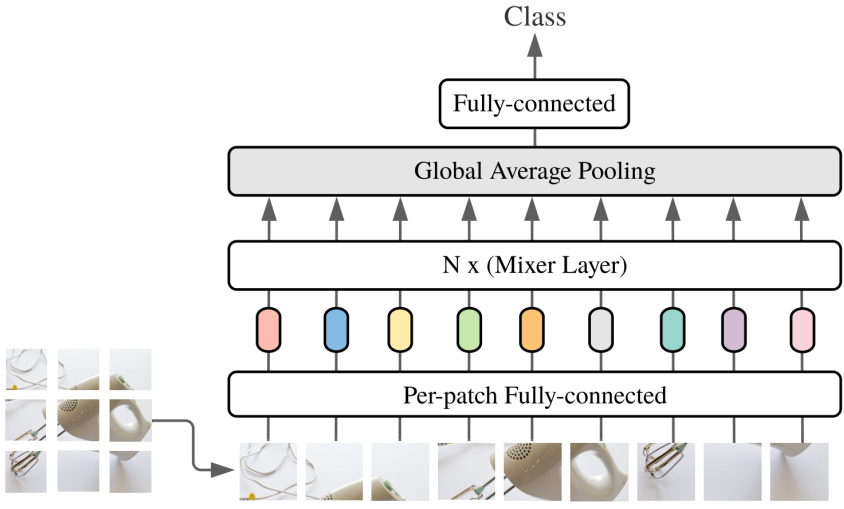
Computer Vision

An Image is Worth 16x16 Words



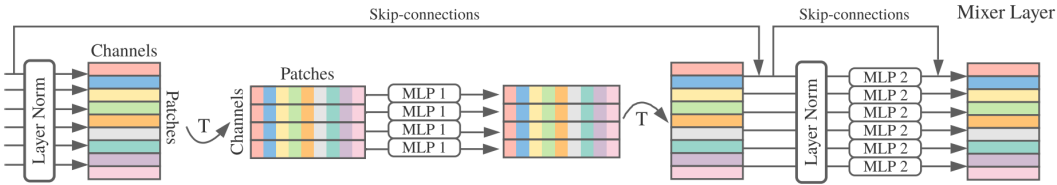


MLP-Mixer
Architecture



MLP-Mixer

Mixer Layer



$$\mathbf{U}_{*,i} = \mathbf{X}_{*,i} + \mathbf{W}_2\sigma\left(\mathbf{W}_1\text{LayerNorm}\left(\mathbf{X}\right)_{*,i}\right), \quad \text{for } i = 1 \dots C,$$

$$\mathbf{Y}_{j,*} = \mathbf{U}_{j,*} + \mathbf{W}_4\sigma\left(\mathbf{W}_3\text{LayerNorm}\left(\mathbf{U}\right)_{j,*}\right), \quad \text{for } j = 1 \dots S$$

Experiments
Setup

Standard transfer learning setup: pre-training followed by fine-tuning on the downstream tasks.

Pre-training

- ILSVRC2021 ImageNet
- ImageNet-21k (21k classes and 14M images)
- JFT-300M (18k classes and 300M images)

Downstream tasks

- ILSVRC2012 "ImageNet" (1.3M training examples, 1k classes): original validation labels and cleaned-up ReaL labels
- CIFAR-10/100 (50k examples, 10/100 classes)
- Oxford-IIIT Pets (3.7k examples, 36 classes)
- Oxford Flowers-102 (2k examples, 102 classes)
- Visual Task Adaptation Benchmark (VTAB-1k): 19 diverse datasets, each with 1k training examples

Experiments
Setup

Metrics

- Total pre-training time on TPU-v3 accelerators
- Throughput in images/sec/core on TPU-v3
- Top-1 downstream accuracy after fine-tuning
- Few-shot accuracies

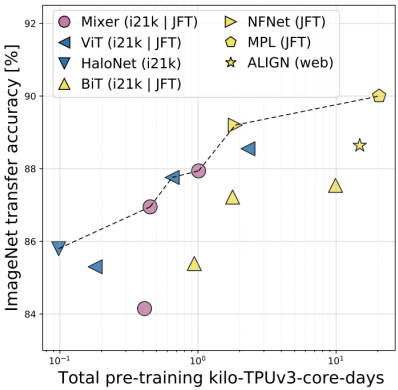
Models

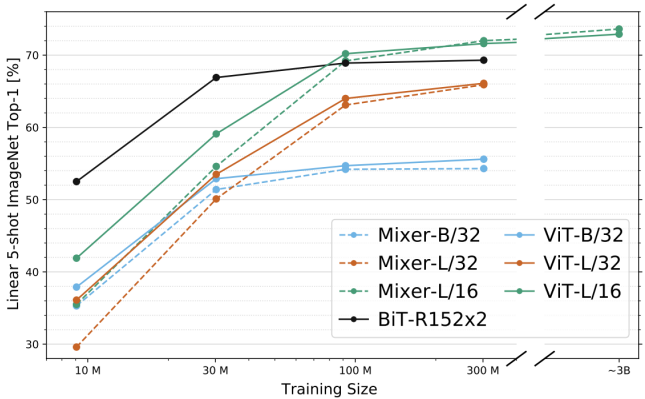
- MLP-based Mixer models
- Convolution-based models
- Attention-based models

Experiments
Transfer performance

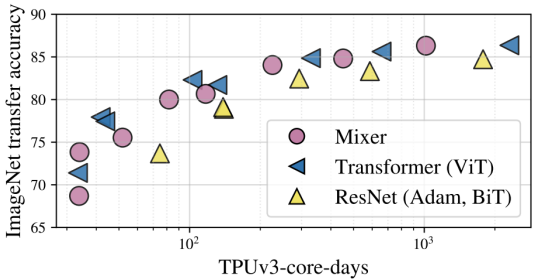
	ImNet top-1	ReaL top-1	Avg 5 top-1	VTAB-1k 19 tasks	Throughput img/sec/core	TPUv3 core-days
Pre-trained on ImageNet-21k (public)						
HaloNet [51]	85.8	—	—	—	120	0.10k
Mixer-L/16	84.15	87.86	93.91	74.95	105	0.41k
ViT-L/16 [14]	85.30	88.62	94.39	72.72	32	0.18k
BiT-R152x4 [22]	85.39	—	94.04	70.64	26	0.94k
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

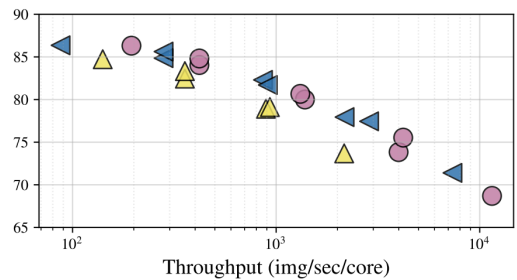
Experiments
Accuracy/training



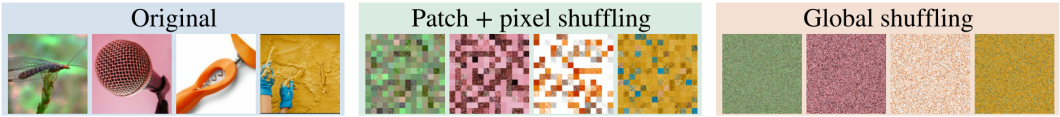


Experiments
Model scale

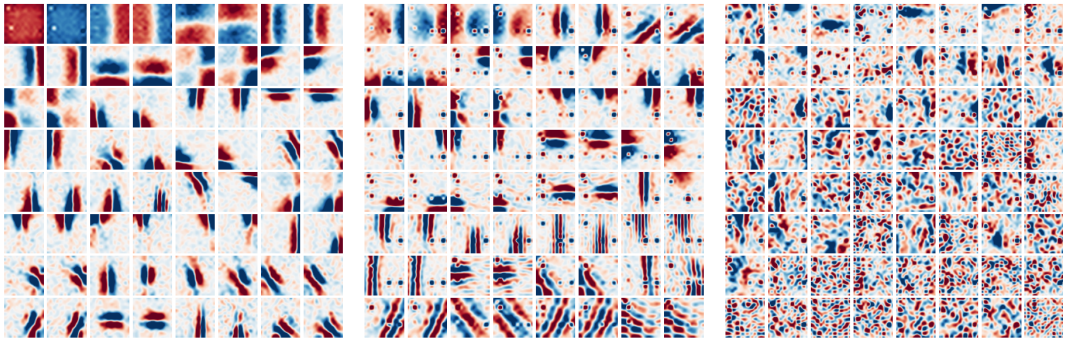




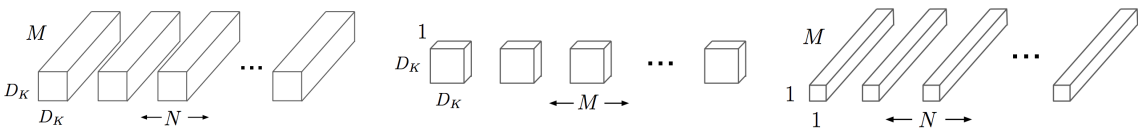
Experiments
Content shuffling



Experiments
Visualization



Hidden units in the first (left), second (center), and third (right) token-mixing MLPs of a Mixer-B/16 model trained on JFT-300M



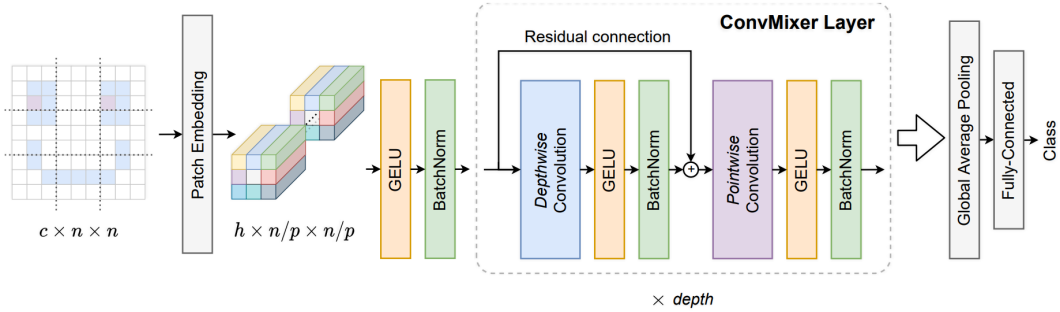
- Depthwise convolution – a spatial convolution performed independently over each channel of an input.
- Pointwise convolution – a 1x1 convolution, projecting the channels output by the depthwise convolution onto a new channel space.

ConvMixer

Patches Are All You Need?

- Whether, fundamentally, the strong performance of vision transformers may result more from this patch-based representation than from the Transformer architecture itself?
- A simple convolutional architecture dubbed “ConvMixer” due to its similarity to the MLP-Mixer
- Similar to the Vision Transformer (and MLP-Mixer):
  - Directly operates on patches
  - Maintains an equal-resolution-and-size representation throughout all layers
  - Does no downsampling of the representation at successive layers
  - Separates “channel-wise mixing”

ConvMixer
Architecture



$$z_0 = \text{BatchNorm}\left(\sigma\left\{\text{Conv}_{c_{in}\to h}\left(X, \text{stride} = p, \text{kernel\_size} = p\right)\right\}\right)$$

$$z'_l = \text{BatchNorm}\left(\sigma\left\{\text{ConvDepthwise}\left(\text{z}_{\text{l}-1}\right)\right\}\right) + z_{l-1}$$

$$z_{l+1} = \text{BatchNorm}\left(\sigma\left\{\text{ConvPointwise}\left(\text{z}'_{\text{l}}\right)\right\}\right)$$

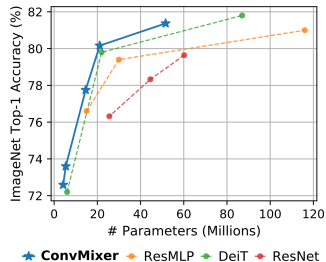
Experiments
Setup

- ImageNet-1k classification without any pretraining or additional data
- Timm framework (Wightman, 2019) with RandAugment (Cubuk et al., 2020), mixup (Zhang et al., 2017), CutMix (Yun et al., 2019), random erasing (Zhong et al., 2020), and gradient norm clipping in addition to default timm augmentation
- AdamW (Loshchilov & Hutter, 2018) optimizer and a simple triangular learning rate schedule
- *No hyperparameter tuning* on ImageNet and training for *fewer epochs than competitors*. Models could be over- or under-regularized, and the accuracies likely underestimate the capabilities of model
- Also CIFAR-10

Experiments
Results

Network	Patch Size	Kernel Size	# Params ($\times 10^6$)	Throughput (img/sec)	Act. Fn.	# Epochs	ImNet top-1 (%)
ConvMixer-1536/20	7	9	51.6	134	G	150	81.37
ConvMixer-768/32	7	7	21.1	206	R	300	80.16
ResNet-152	–	3	60.2	828	R	150	79.64
DeiT-B	16	–	86	792	G	300	81.8
ResMLP-B24/8	8	–	129	181	G	400	81.0

Models trained and evaluated on 224×224 ImageNet-1k only



Experiments

More results

⋆ Using new, better regularization hyperparameters based on Wightman et al. (2021)'s A1 procedure

† Throughput tested, but not trained

Activations: **R**eLU, **G**ELU

Models with matching colored dots (•) are informative to compare with each other

Network	Patch Size	Kernel Size	# Params ($\times 10^6$)	Throughput (img/sec)	Act. Fn.	# Epochs	ImNet top-1 (%)
ConvMixer-1536/20⋆	7	9	51.6	134	G	150	82.20
ConvMixer-1536/20 •	7	9	51.6	134	G	150	81.37
ConvMixer-1536/20⋆	7	3	49.4	246	G	150	81.60
ConvMixer-1536/20	7	3	49.4	246	G	150	80.43
ConvMixer-1536/20	14	9	52.3	538	G	150	78.92
ConvMixer-1536/24⋆	14	9	62.3	447	G	150	80.21
ConvMixer-768/32 •	7	7	21.1	206	R	300	80.16
ConvMixer-1024/16	7	9	19.4	244	G	100	79.45
ConvMixer-1024/12	7	8	14.6	358	G	90	77.75
ConvMixer-512/16	7	8	5.4	599	G	90	73.76
ConvMixer-512/12 •	7	8	4.2	798	G	90	72.59
ConvMixer-768/32	14	3	20.2	1235	R	300	74.93
ConvMixer-1024/20 •	14	9	24.4	750	G	150	76.94
ResNet-152 •	–	3	60.2	828	R	150	79.64
ResNet-101 •	–	3	44.6	1187	R	150	78.33
ResNet-50	–	3	25.6	1739	R	150	76.32
DeiT-B†	7	–	86.7	83	G	–	–
DeiT-S†	7	–	22.1	174	G	–	–
DeiT-Ti†	7	–	5.7	336	G	–	–
DeiT-B •	16	–	86	792	G	300	81.8
DeiT-S •	16	–	22	1610	G	300	79.8
DeiT-Ti •	16	–	5.7	2603	G	300	72.2
ResMLP-S12/8 •	8	–	22.1	872	G	400	79.1
ResMLP-B24/8 •	8	–	129	181	G	400	81.0
ResMLP-B24	16	–	116	1597	G	400	81.0
Swin-S •	4	–	50	576	G	300	83.0
Swin-T •	4	–	29	878	G	300	81.3
ViT-B/16 •	16	–	86	789	G	300	77.9
Mixer-B/16 •	16	–	59	1025	G	300	76.44
Isotropic MobileNetv3 •	8	3	20	–	R	–	80.6
Isotropic MobileNetv3 •	16	3	20	–	R	–	77.6

```
def ConvMixer(h,d,k,p,n):
 S,C,A=Sequential,Conv2d,lambda x:S(x,GELU(),BatchNorm2d(h))
 R=type('',(S,),{'forward':lambda s,x:s[0](x)+x})
 return S(A(C(3,h,p,p)),*[S(R(A(C(h,h,k,groups=h,padding=k//2))),A(C(h,h,1))) for i
 ↪  in range(d)],AdaptiveAvgPool2d(1),Flatten(),Linear(h,n))
```

An implementation of ConvMixer in less than 280 characters, in case you happen to know of any means of disseminating information that could benefit from such a length.
All you need to do to run this is `from torch.nn import *`.