

رسالة محمد



مبانی بینایی کامپیوتر

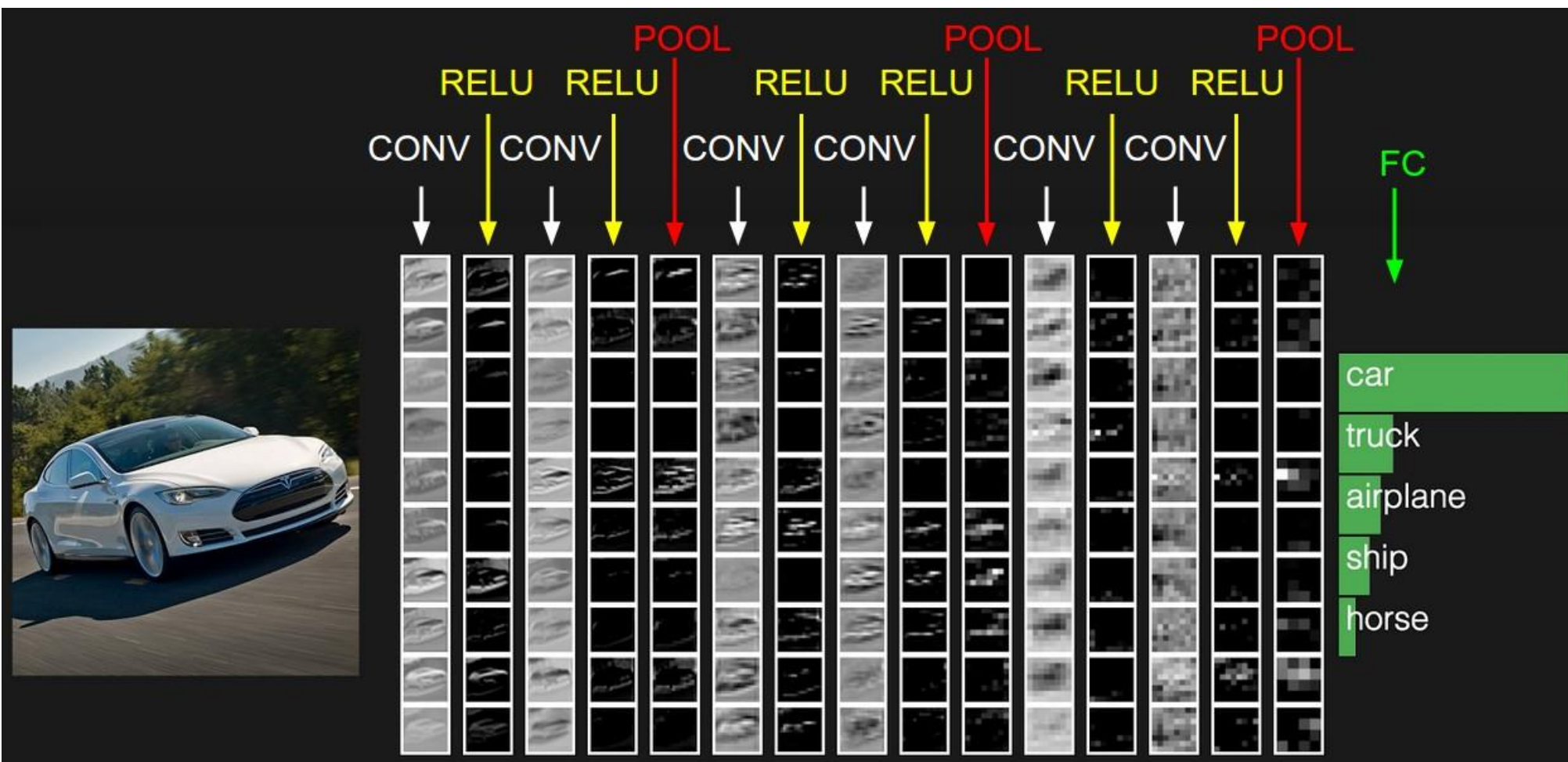
مدرس: محمدرضا محمدی

بهار ۱۴۰۲

شبکه‌های عصبی کانولوشنی

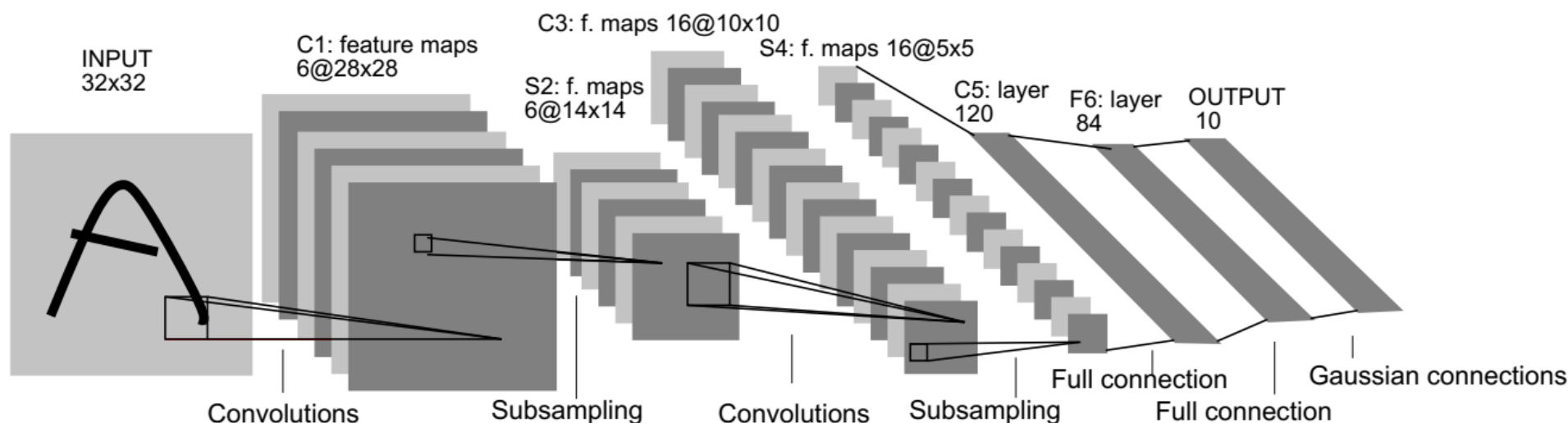
Convolutional Neural Networks

شبکه‌های کانولوشنی برای دسته‌بندی



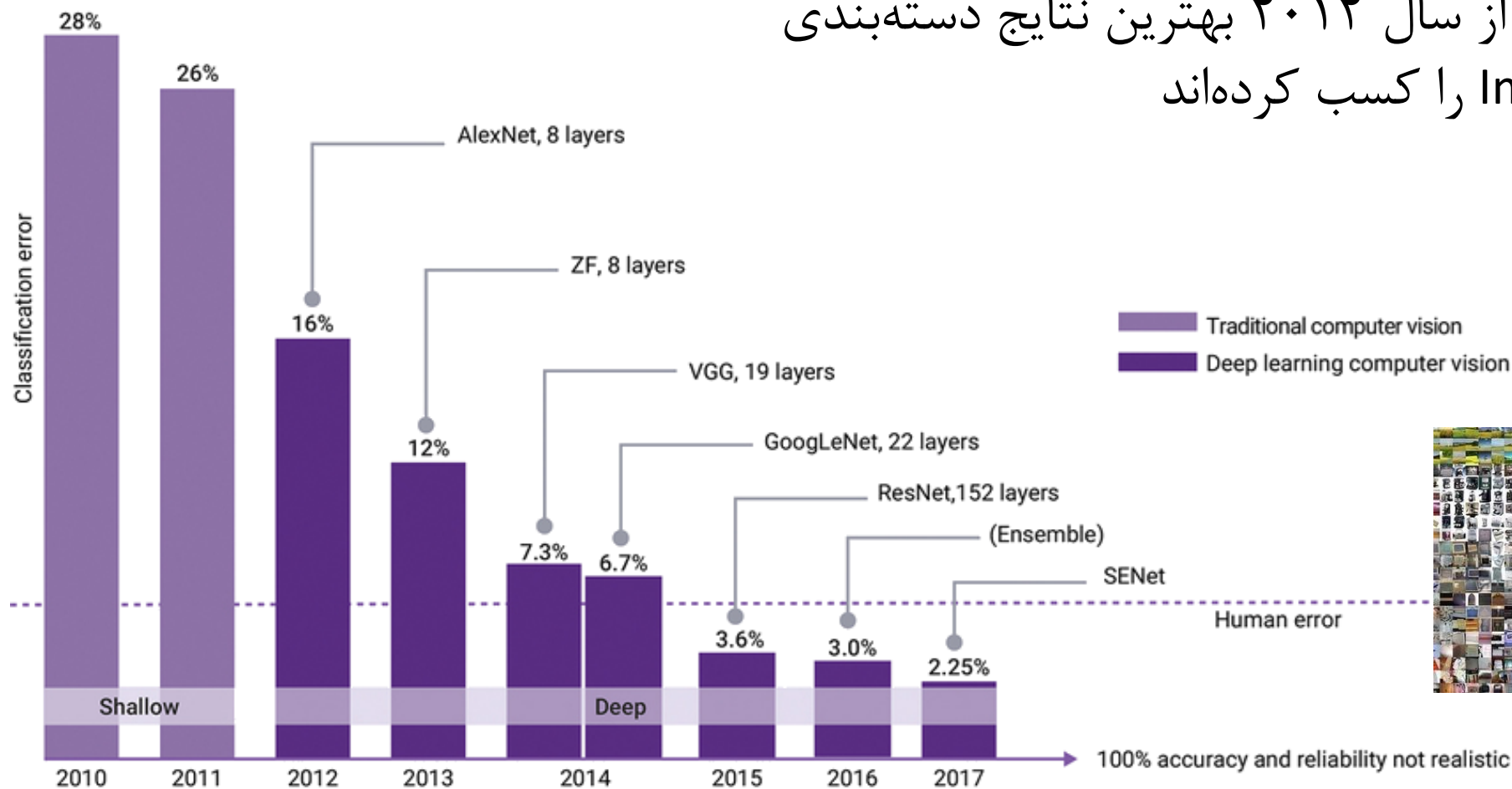
شبکه LeNet-5

- شبکه LeNet-5 در سال ۱۹۹۸ برای شناسایی اعداد و حروف دستنویس پیشنهاد شد
- این شبکه تنها دارای ۵ لایه آموزشی است: ۲ لایه کانولوشنی و ۳ لایه کاملاً متصل



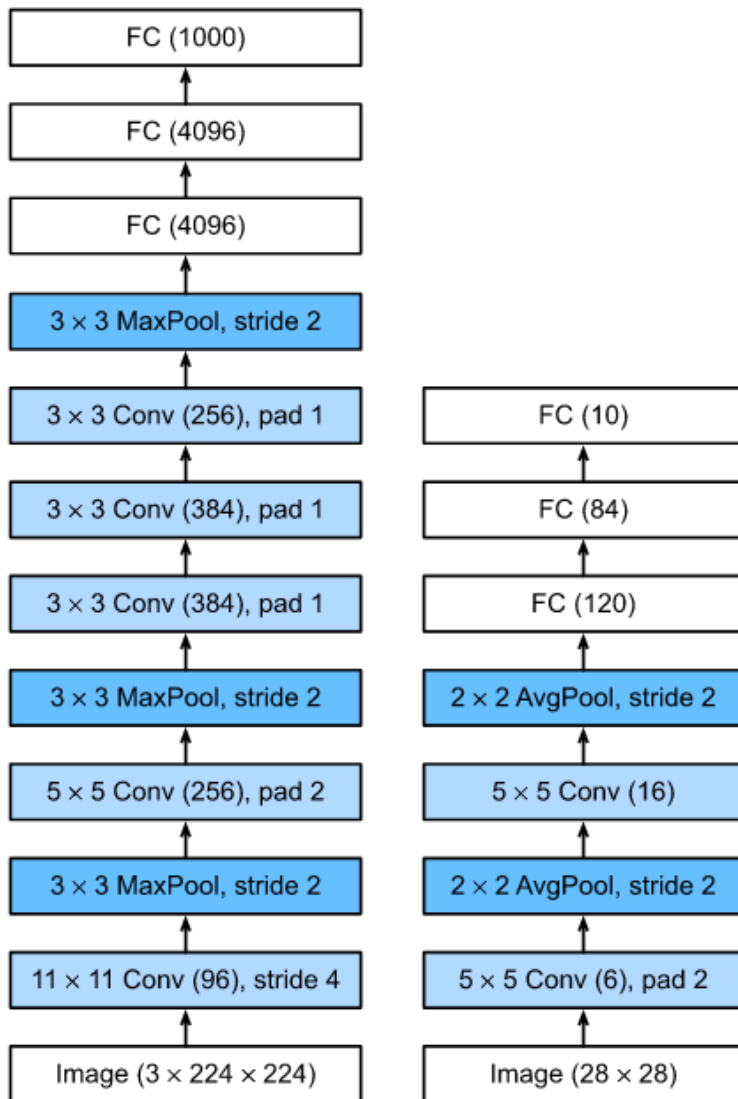
نتایج ILSVRC

- معماری‌های مختلف CNN از سال ۲۰۱۲ بهترین نتایج دسته‌بندی تصویر در چالش ImageNet را کسب کرده‌اند



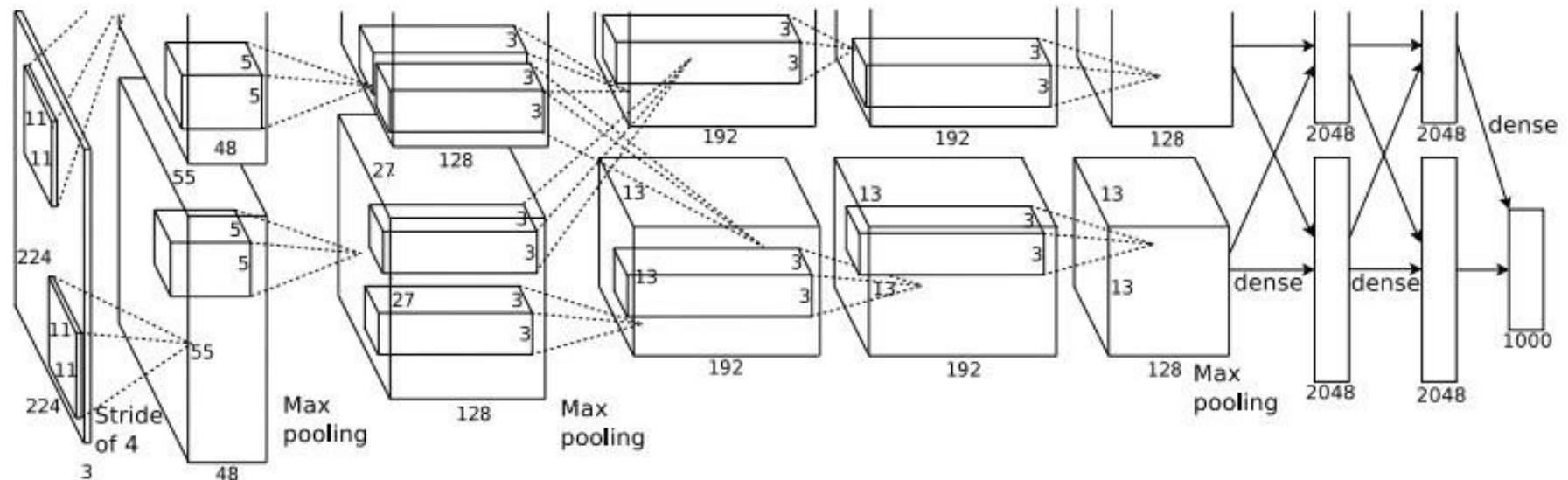
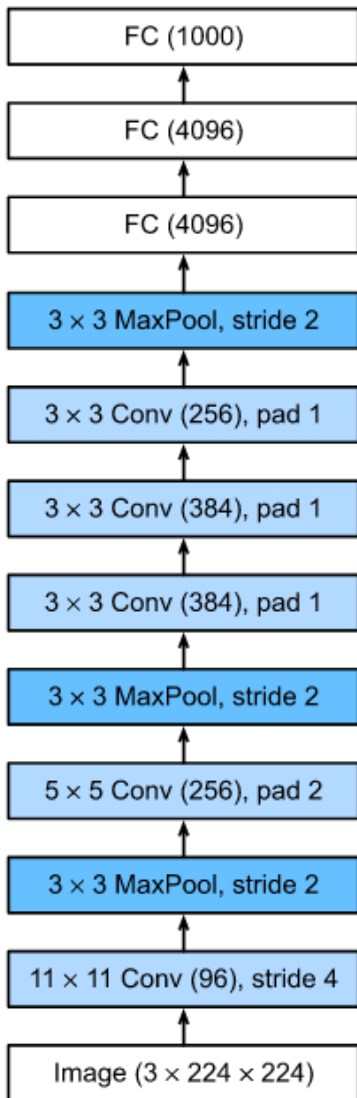
AlexNet

- اگرچه LeNet در مجموعه داده های کوچک اولیه به نتایج خوبی دست یافت، عملکرد CNN بر روی مجموعه داده های بزرگتر و واقعی تر هنوز مشخص نبود
- در الگوریتم های بینایی کامپیوتر رقیب، معمولاً ابتدا از تصویر ویژگی های دست ساز استخراج می شوند
- با توسعه سخت افزارها و مجموعه داده های بزرگ، یادگیری ویژگی توسط شبکه های کانولوشنی عمیق نتایج بسیار خوبی بدست آوردند



AlexNet

- شبکه AlexNet یک شبکه دارای ۸ لایه آموزشی است که در سال ۲۰۱۲ پیشنهاد شد و توانست خطای top-5 در چالش ILSVRC'12 را به ۱۵.۳٪ کاهش دهد
- استفاده از Dropout، ReLU و داده‌افزایی نیز در عملکرد AlexNet موثر بوده‌اند
- به دلیل محدودیت سخت‌افزار، به صورت موازی روی دو GPU پیاده‌سازی شده بود



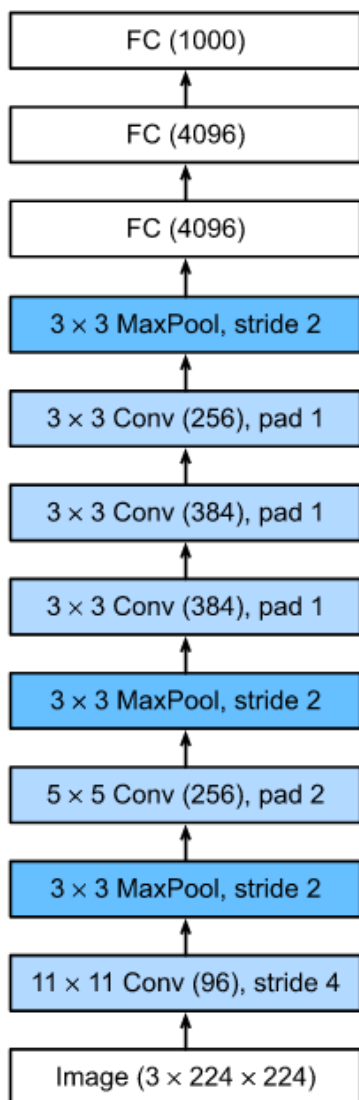
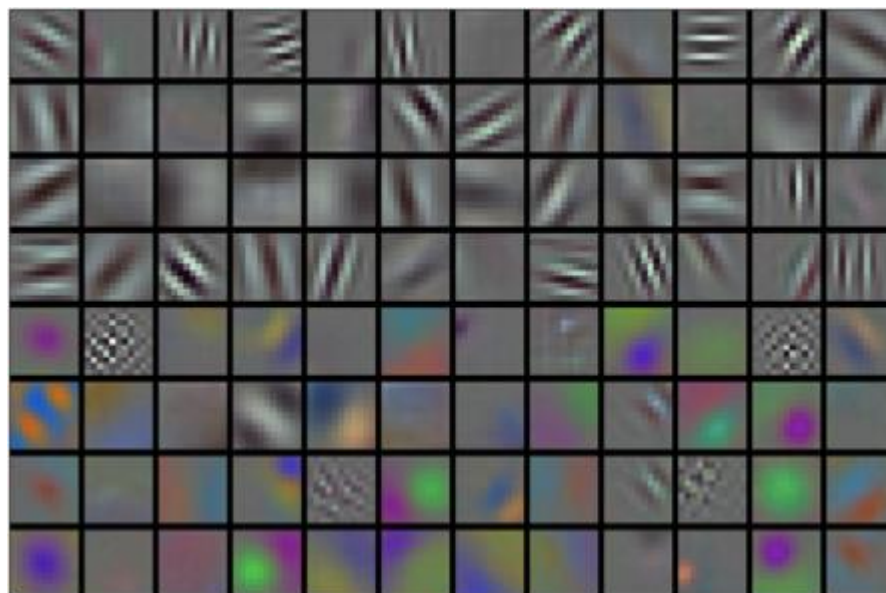
یادگیری بازنمایی

- تا سال ۲۰۱۲، بازنمایی‌های جدید از تصاویر معمولاً به صورت دست‌ساز طراحی می‌شدند
- رویکرد دیگر طراحی مدل‌هایی است که پارامترهای آنها قابل آموزش است و می‌تواند بازنمایی مناسب برای حل مسئله را یاد بگیرد

- ۹۶ فیلتر $11 \times 11 \times 3$ لایه نخست:

- توصیفگرهای سطح پائین تصویر

- در لایه‌های بالاتر، ساختارهای پیچیده‌تر و بزرگتری مانند چشم تشخیص داده می‌شوند



Keras Applications

keras.io/api/applications/

Losses

Data loading

Built-in small datasets

Keras Applications

Xception

EfficientNet B0 to B7

EfficientNetV2 B0 to B3 and S, M, L

ConvNeXt Tiny, Small, Base, Large, XLarge

VGG16 and VGG19

ResNet and ResNetV2

MobileNet, MobileNetV2, and MobileNetV3

DenseNet

NasNetLarge and NasNetMobile

InceptionV3

InceptionResNetV2

Mixed precision

Utilities

KerasTuner

KerasCV

Model	Size (MB)	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth	Time (ms) per inference step (CPU)	Time (ms) per inference step (GPU)
Xception	88	79.0%	94.5%	22.9M	81	109.4	8.1
VGG16	528	71.3%	90.1%	138.4M	16	69.5	4.2
VGG19	549	71.3%	90.0%	143.7M	19	84.8	4.4
ResNet50	98	74.9%	92.1%	25.6M	107	58.2	4.6
ResNet50V2	98	76.0%	93.0%	25.6M	103	45.6	4.4
ResNet101	171	76.4%	92.8%	44.7M	209	89.6	5.2
ResNet101V2	171	77.2%	93.8%	44.7M	205	72.7	5.4
ResNet152	232	76.6%	93.1%	60.4M	311	127.4	6.5
ResNet152V2	232	78.0%	94.2%	60.4M	307	107.5	6.6
InceptionV3	92	77.9%	93.7%	23.9M	189	42.2	6.9
InceptionResNetV2	215	80.3%	95.3%	55.9M	449	130.2	10.0
MobileNet	16	70.4%	89.5%	4.3M	55	22.6	3.4
MobileNetV2	14	71.3%	90.1%	3.5M	105	25.9	3.8
DenseNet121	33	75.0%	92.3%	8.1M	242	77.1	5.4
DenseNet169	57	76.2%	93.2%	14.3M	338	96.4	6.3
DenseNet201	80	77.3%	93.6%	20.2M	402	127.2	6.7
NASNetMobile	23	74.4%	91.9%	5.3M	389	27.0	6.7
NASNetLarge	343	82.5%	96.0%	88.9M	533	344.5	20.0
EfficientNetB0	29	77.1%	93.3%	5.3M	132	46.0	4.9

Keras Applications

Available models

Usage examples for image classification models

Classify ImageNet classes with ResNet50

Extract features with VGG16

Extract features from an arbitrary intermediate layer with VGG19

Fine-tune InceptionV3 on a new set of classes

Build InceptionV3 over a custom input tensor

Cars Dataset



Overview

The *Cars* dataset contains 16,185 images of 196 classes of cars. The data is split into 8,144 training images and 8,041 testing images, where each class has been split roughly in a 50-50 split. Classes are typically at the level of *Make*, *Model*, *Year*, e.g. 2012 Tesla Model S or 2012 BMW M3 coupe.

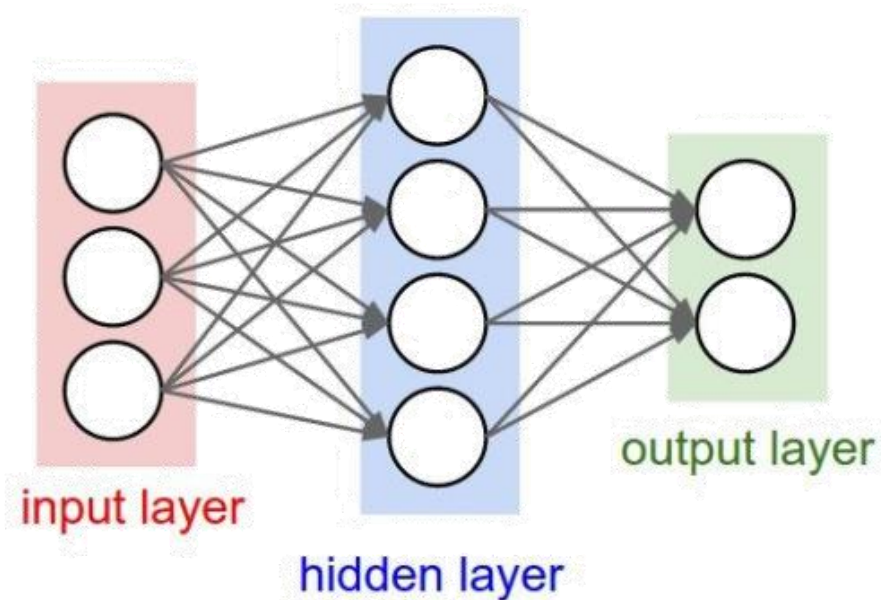
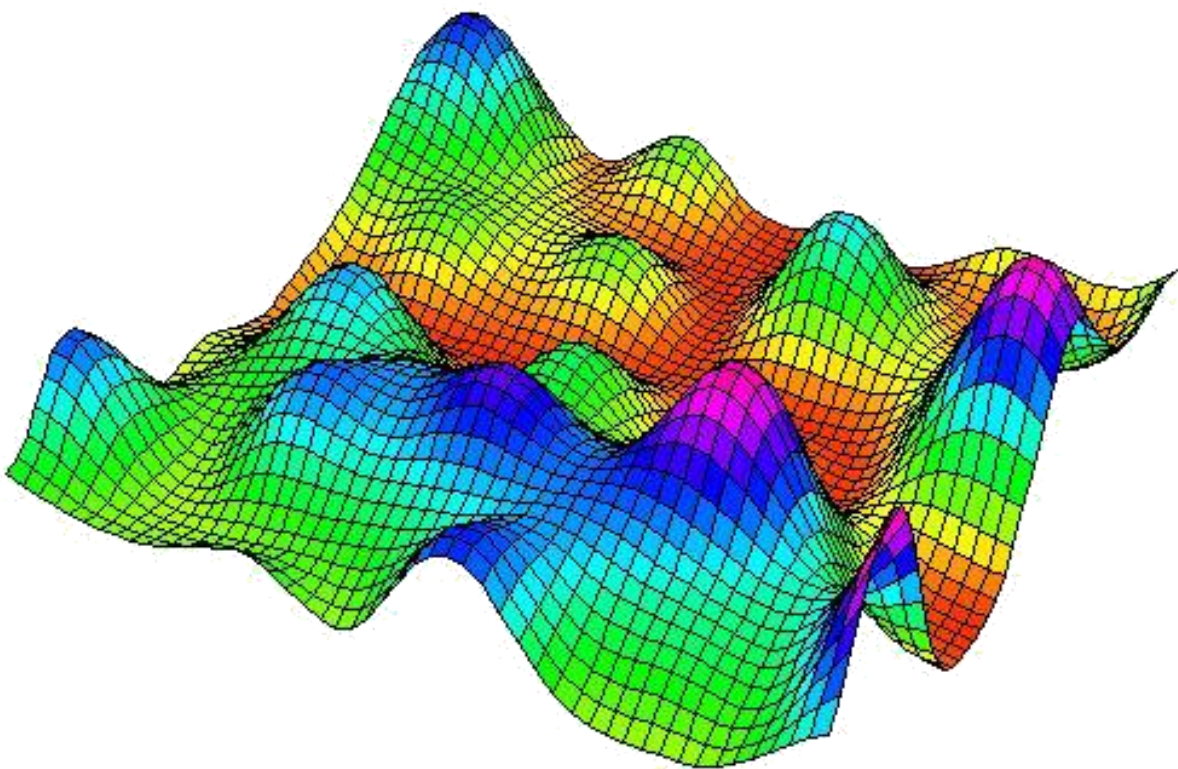


مقداردهی اولیه وزن‌ها

Weight Initialization

مقداردهی اولیه

- در روش‌های بهینه‌سازی مبتنی بر تکرار، نقطه شروع بهینه‌سازی بسیار مهم است
- اگر در ابتدای کار تمام وزن‌های شبکه مقدار صفر داشته باشند چه اتفاقی می‌افتد؟



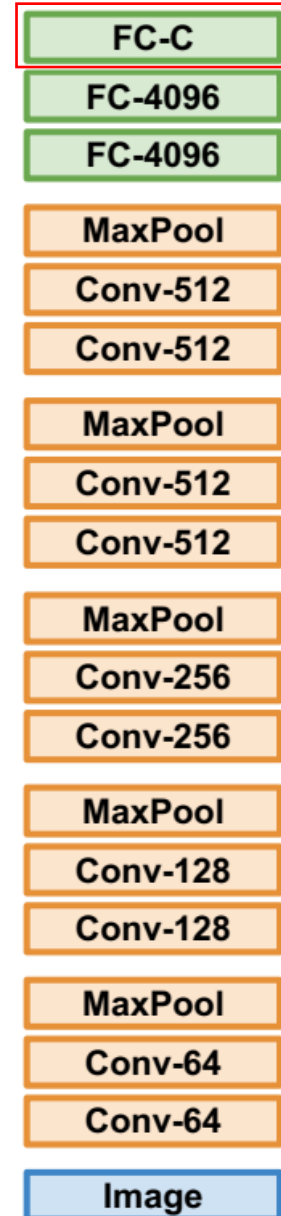
مقداردهی اولیه

- روش مقداردهی اولیه Xavier یکی از معروفترین روش‌های وزن‌دهی اولیه است
- مقداردهی اولیه مناسب هنوز یک زمینه تحقیقاتی فعال است
- به خصوص برای مجموعه داده‌های کوچک، مقداردهی اولیه بسیار حائز اهمیت است
- برای آموزش یک شبکه CNN با میلیون‌ها پارامتر، حجم زیادی از داده‌های آموزشی لازم است
- با استفاده از تقویت داده و دیگر روش‌های تنظیم پارامترهای شبکه می‌توان تا حدی کمبود داده را جبران کرد
- یکی از بهترین روش‌ها برای مقداردهی اولیه پارامترهای یک شبکه استفاده از شبکه‌های pretrained است
- انتقال یادگیری روش بسیار موثری است تا دانش بدست آمده توسط یک شبکه به شبکه جدید منتقل شود

Train on ImageNet



Small Dataset (C classes)



Reinitialize
this and train

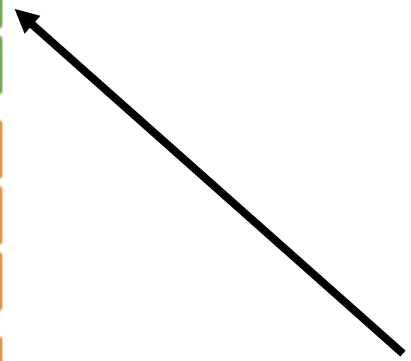
Freeze
these

Bigger Dataset (C classes)

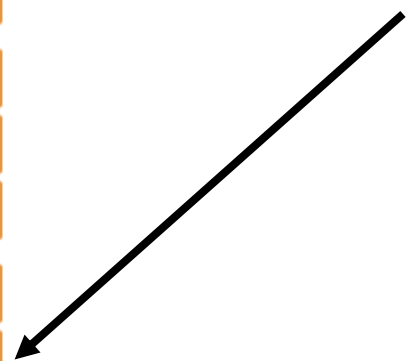


Train these

Freeze
these



More specific



More generic



مجموعه داده بسیار متفاوت	مجموعه داده بسیار مشابه	
?	?	مجموعه داده خیلی کم
?	?	مجموعه داده زیاد



مجموعه داده بسیار متفاوت	مجموعه داده بسیار مشابه	
?	از یک دسته‌بند خطی در آخرین لایه استفاده شود	مجموعه داده خیلی کم
?	?	مجموعه داده زیاد



مجموعه داده بسیار متفاوت	مجموعه داده بسیار مشابه	
?	از یک دسته‌بند خطی در آخرین لایه استفاده شود	مجموعه داده خیلی کم
?	تعدادی از لایه‌های انتهایی تنظیم دقیق شوند	مجموعه داده زیاد



مجموعه داده بسیار متفاوت	مجموعه داده بسیار مشابه	
مشکل است! می توان دسته بند خطی را در گام های مختلف امتحان کرد	از یک دسته بند خطی در آخرین لایه استفاده شود	مجموعه داده خیلی کم
?	تعدادی از لایه های انتهایی تنظیم دقیق شوند	مجموعه داده زیاد



مجموعه داده بسیار متفاوت	مجموعه داده بسیار مشابه	
مشکل است! می توان دسته بند خطی را در گام های مختلف امتحان کرد	از یک دسته بند خطی در آخرین لایه استفاده شود	مجموعه داده خیلی کم
تعداد زیادی از لایه های انتهایی تنظیم دقیق شوند	تعدادی از لایه های انتهایی تنظیم دقیق شوند	مجموعه داده زیاد

انتقال یادگیری

- در صورتیکه مجموعه داده‌های شما به اندازه کافی بزرگ نیست و مسئله پیچیده است (شبکه دارای پارامترهای زیادی است):
 - یک مجموعه داده بسیار بزرگ که به مجموعه داده مورد نظر مشابه است انتخاب و شبکه کانولوشنی با آن آموزش ببیند
 - انتقال یادگیری به مجموعه داده مورد نظر انجام شود
- خوشبختانه مدل‌های پیش‌آمोخته زیادی در دسترس هستند

PyTorch: <https://github.com/pytorch/vision>

TensorFlow: <https://github.com/tensorflow/models>

Caffe: <https://github.com/BVLC/caffe/wiki/Model-Zoo>

MatConvNet: <http://www.vlfeat.org/matconvnet/pretrained/>

Keras: <https://github.com/fchollet/deep-learning-models/releases/>