Convolutional Neural Networks III

Semester 2, 2025

Kris Ehinger

Outline

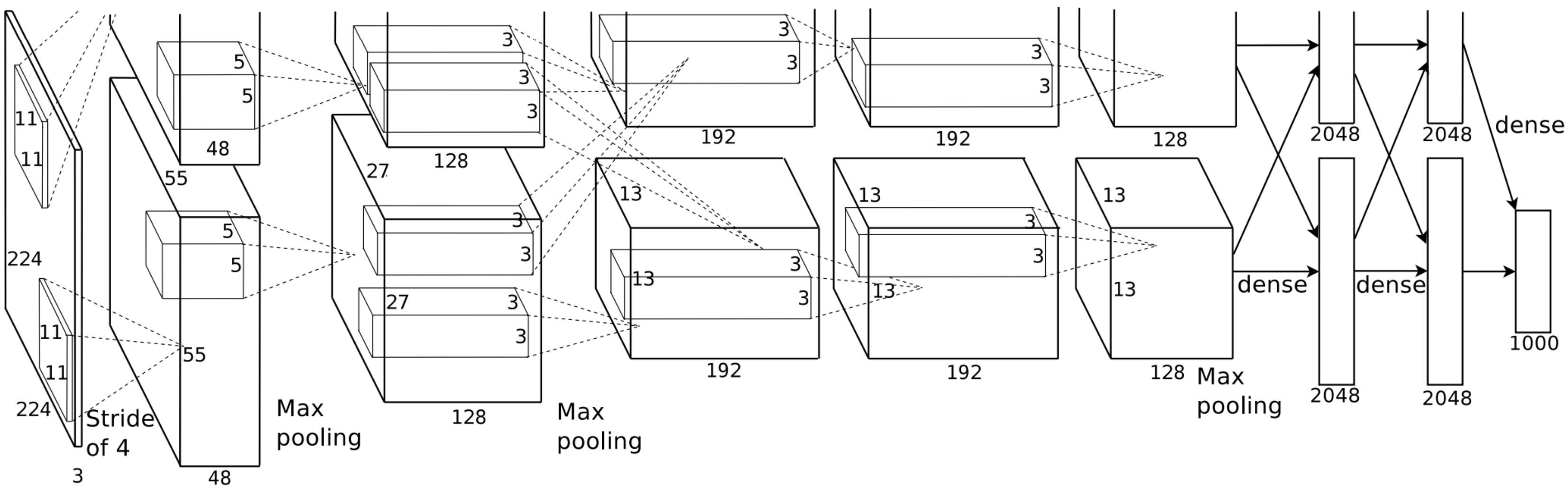
- CNN architectures for ImageNet classification
- ImageNet classification results
- Transfer learning

Learning outcomes

- Explain the key differences between and main ideas behind different architectures for ImageNet classification
- Evaluate image classification results
- Explain and implement transfer learning with networks pretrained on ImageNet

CNN architectures

AlexNet

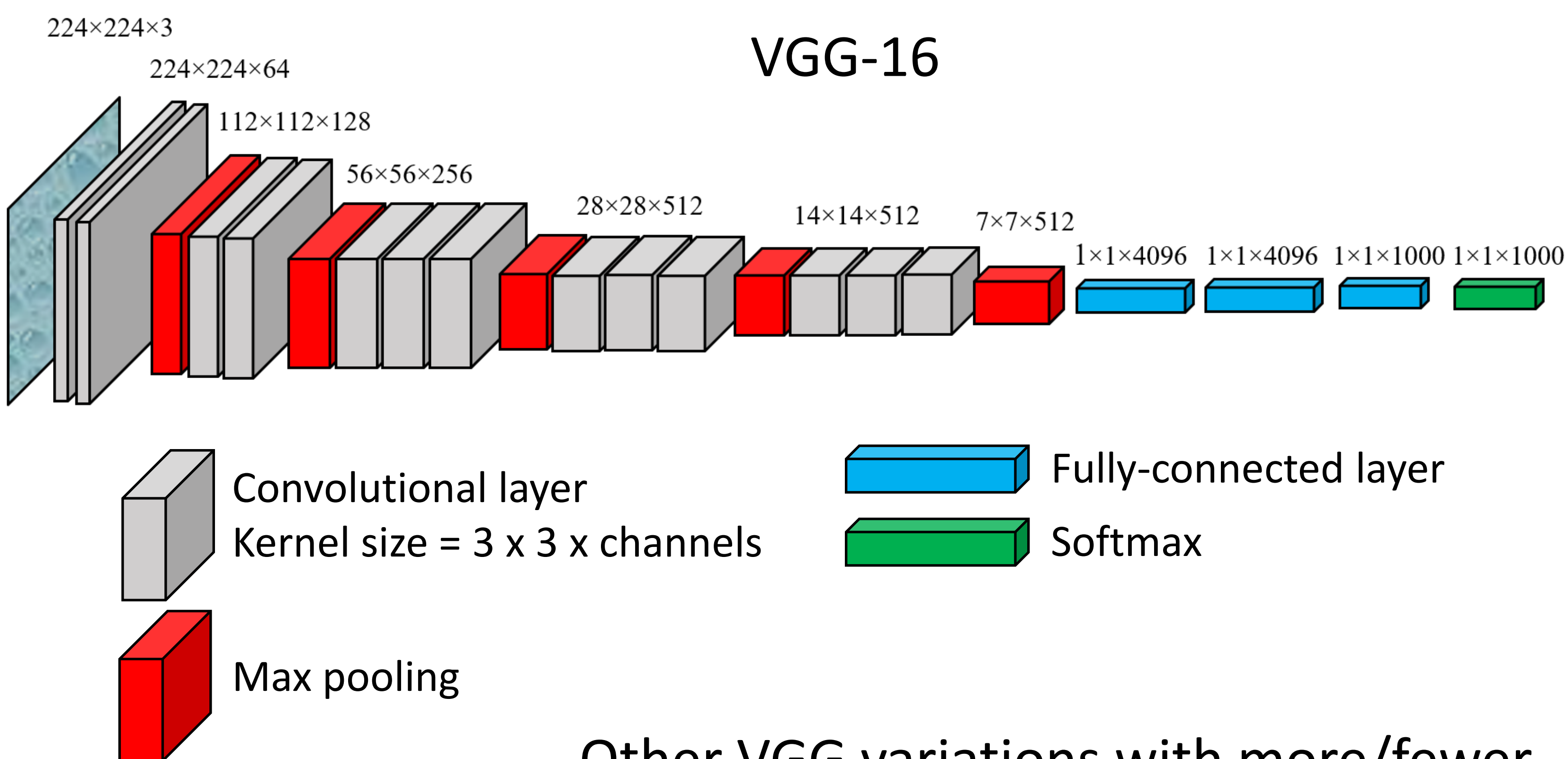


Krizhevsky, Sutskever, & Hinton (2012)

AlexNet innovations

- ReLU (Rectified Linear Unit) activation function – faster training
- Training on GPU – parallelisation allows faster training (actually required 2 GPUs at the time!)
- Overlapping max pooling regions, response normalisation after ReLU – small accuracy increase
- Data augmentation – reduces overfitting
- Dropout – reduces overfitting

VGG

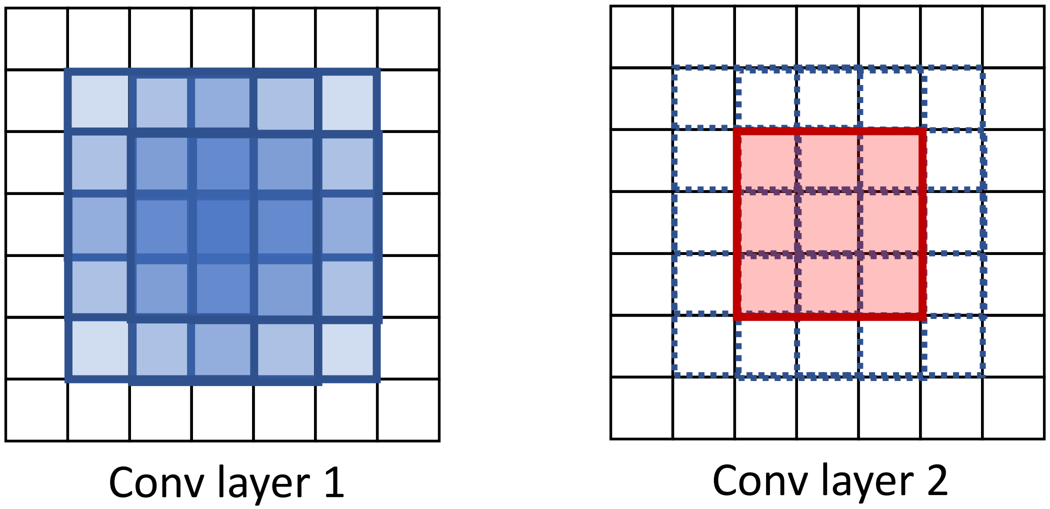


Other VGG variations with more/fewer layers (e.g., VGG-19)

Simonyan & Zisserman (2014)

Stacked convolutional layers

- VGG stacks multiple 3 x 3 convolutional kernels to effectively make larger kernels:
  - Two 3 x 3 conv. layers = effective receptive field of 5 x 5
  - Three 3 x 3 conv. layers = effective receptive field of 7 x 7

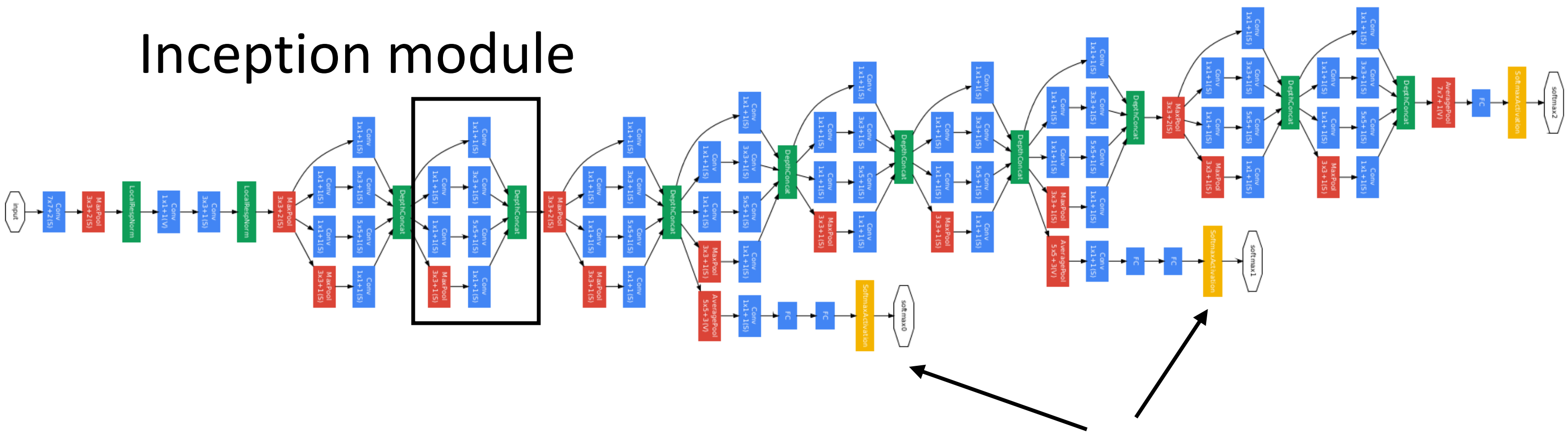


VGG innovations

- Stacked 3x3 convolutional layers
  - Learn more complex features thanks to additional non-linearities
  - Fewer parameters than 1 layer with the equivalent receptive field
- Doesn’t use AlexNet’s response normalisation – allows faster training with only very small accuracy drop

GoogLeNet (Inception)

Inception module



Auxillary classifers

Convolutional layer

Concatenate & normalise

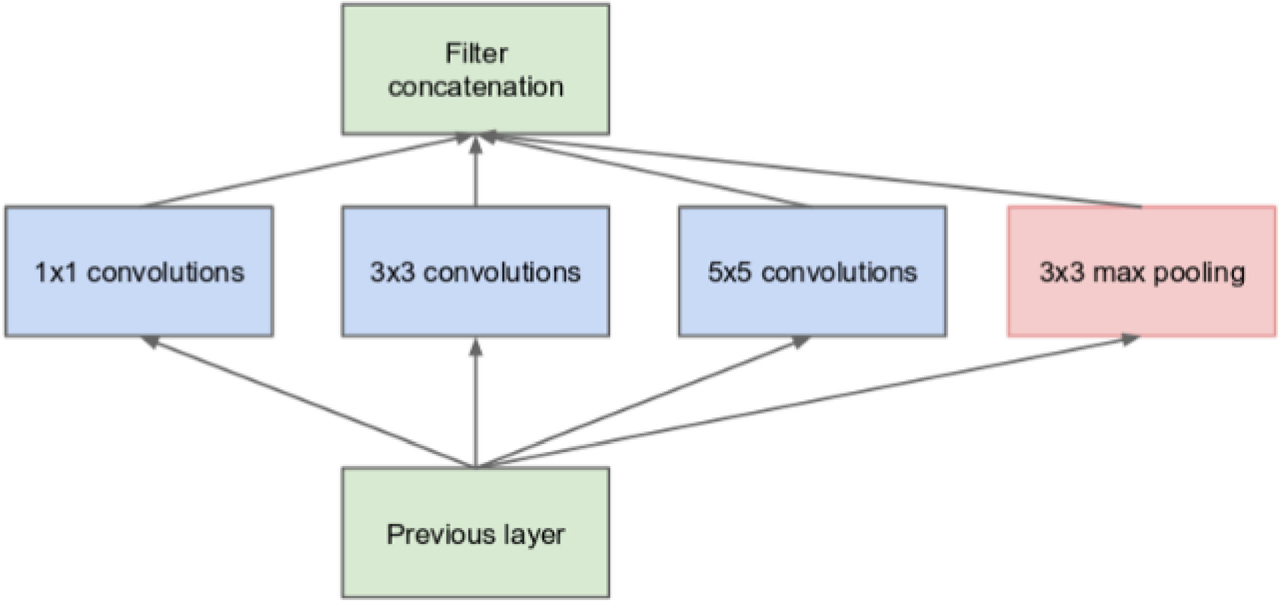
Max pooling

Softmax

Szegedy, et al. (2014)

Inception module

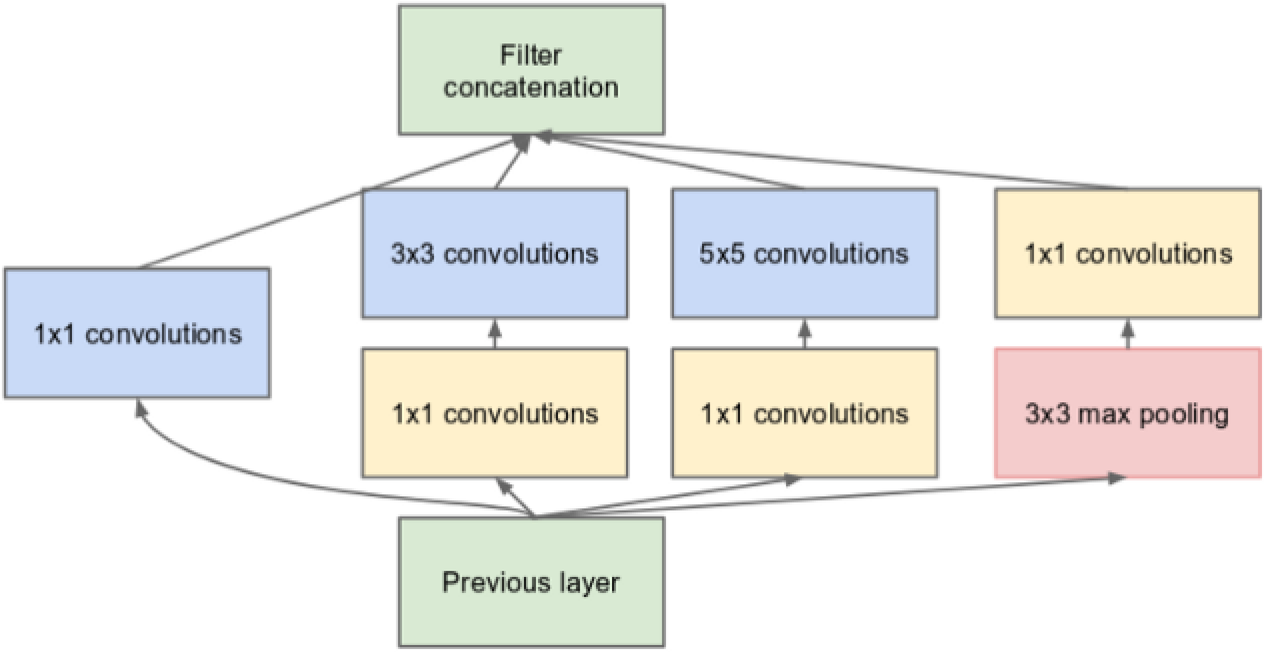
- Choosing the right kernel size in CNNs is difficult because objects/features can appear at any scale
- Solution: use multiple kernel sizes and concatenate



Szegedy, et al. (2014)

Inception module

- 1x1 convolutional layers reduce the number of channels (dimensionality reduction)



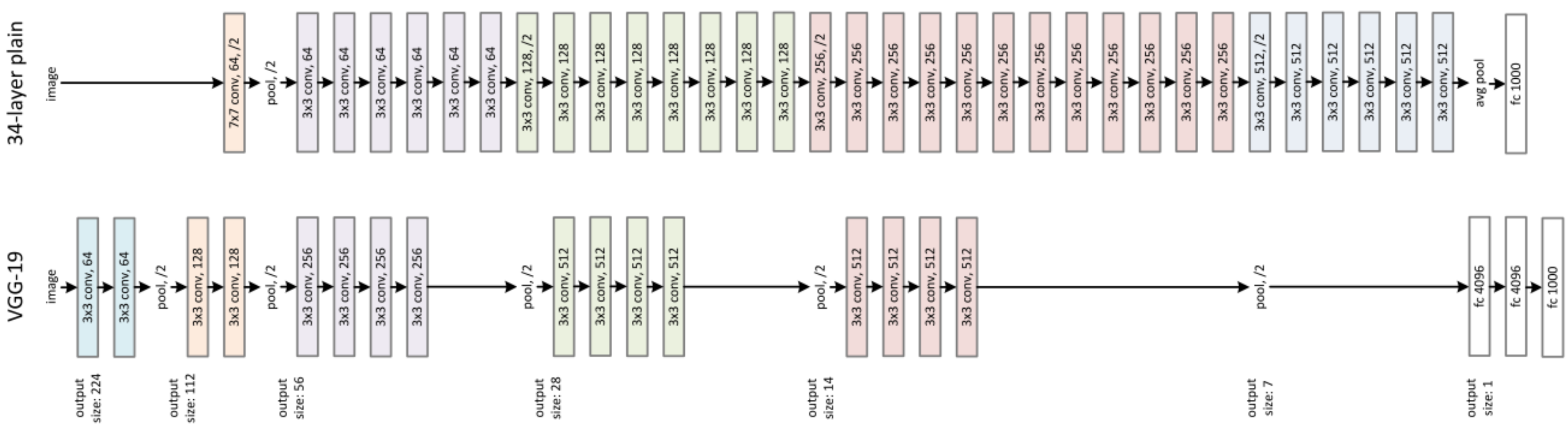
Szegedy, et al. (2014)

GoogLeNet innovations

- Inception module
  - Learns features at a variety of kernel sizes/scales
- Auxillary classifiers
  - Used during training only – classify images based on early layer representations and update parameters
  - Helps with vanishing gradient problem

ResNet: Background

- Will deeper neural networks will always give better performance?

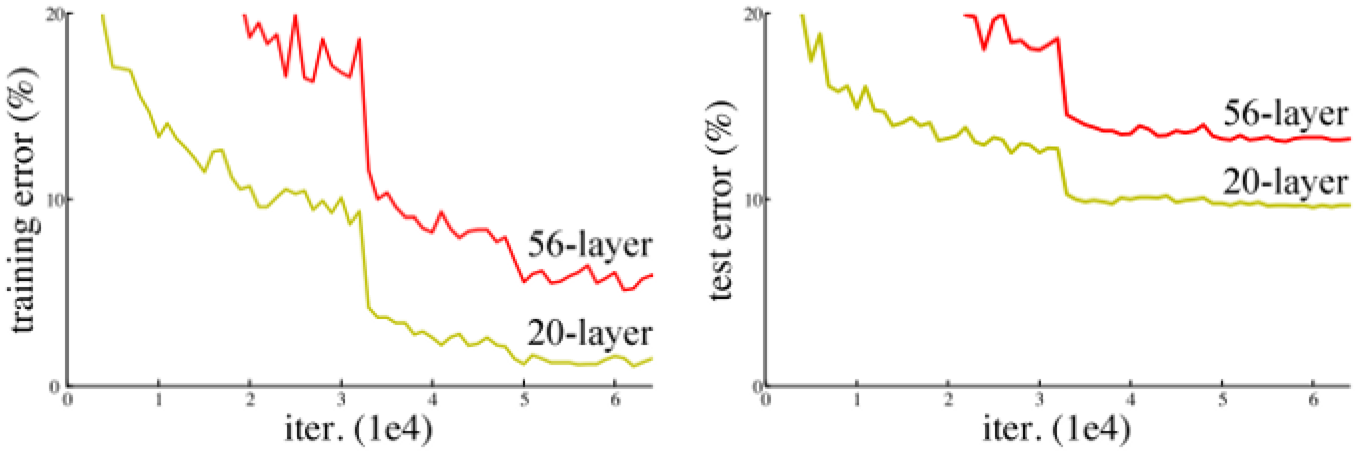


He, Zhang, Ren, & Sun (2016)

ResNet: Background

- No, performance saturates and then decreases
- Not due to overfitting – performance is worse on the training set

CIFAR-10 classification



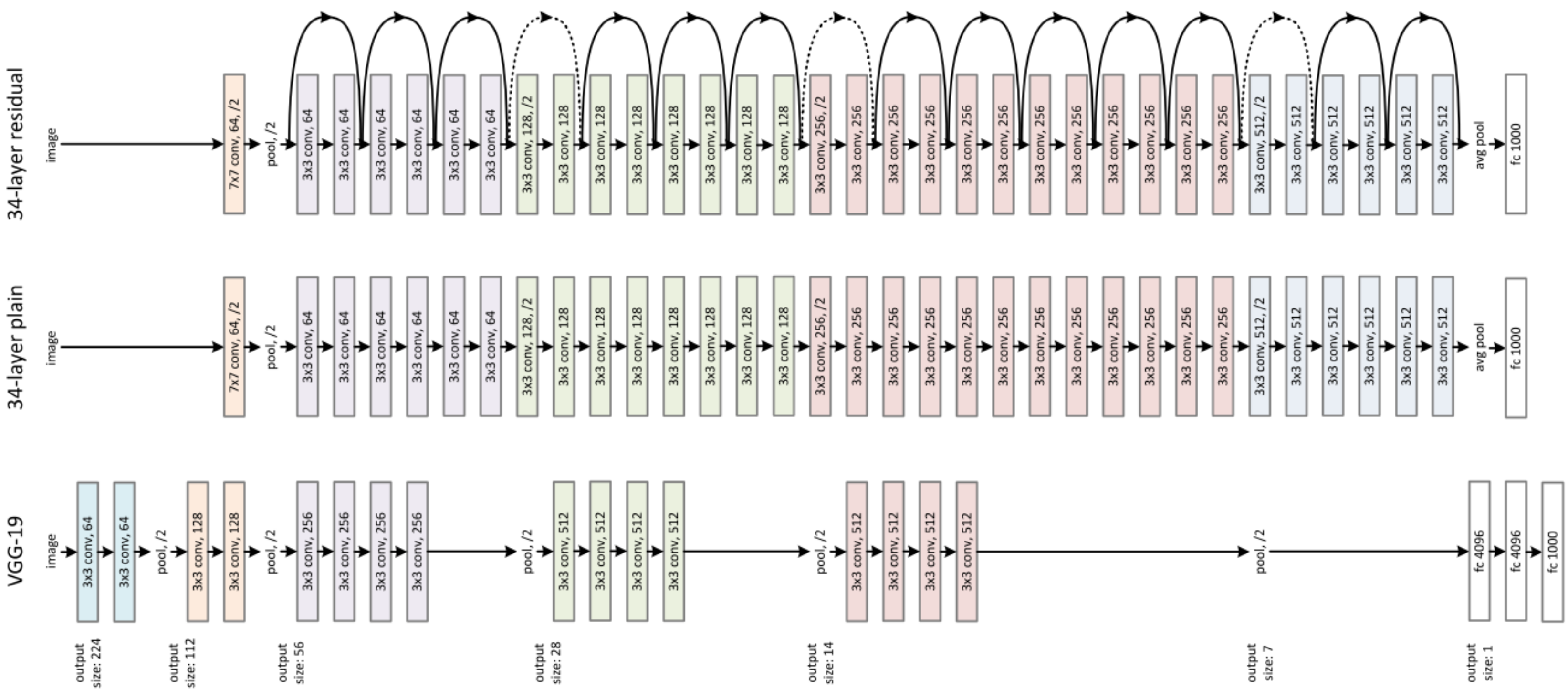
He, Zhang, Ren, & Sun (2016)

ResNet: Background

- It should be possible to learn parameters in the deep network that would allow it to act like the small network
  - For example, some conv. layers learn identity kernels, while others learn the shallow network's kernels
- However, deep CNNs cannot learn this solution (at least, not within a reasonable training time)

ResNet

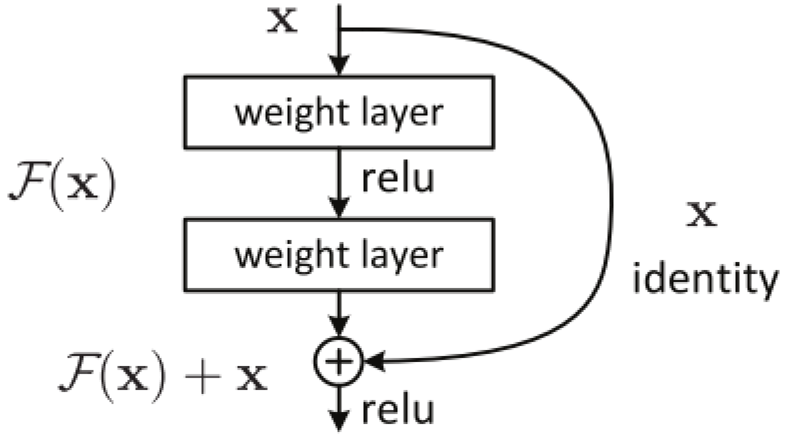
- Solution: Add “shortcut connections” that skip some layers



He, Zhang, Ren, & Sun (2016)

Residual learning

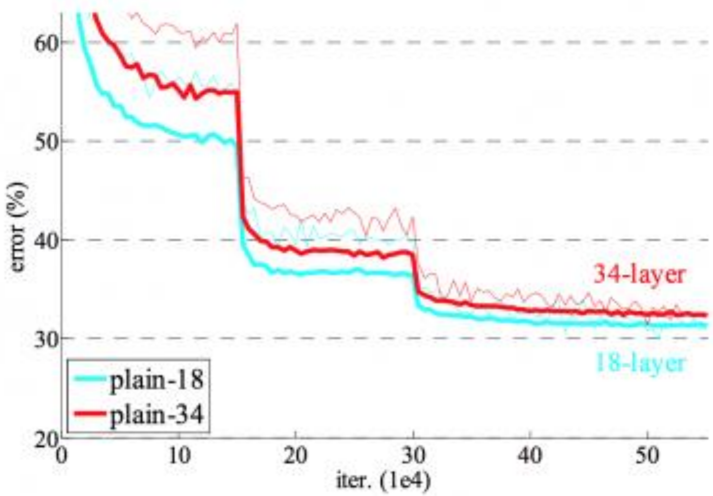
- Reformulate the learning problem:
  - Traditional network: input x, output $\mathcal{H}(x)$, which is the feature representation of x
  - Residual network: input x, learn $\mathcal{H}(x) - x$, which is then added to x to get $\mathcal{H}(x)$
- Makes it easier to learn identity mapping

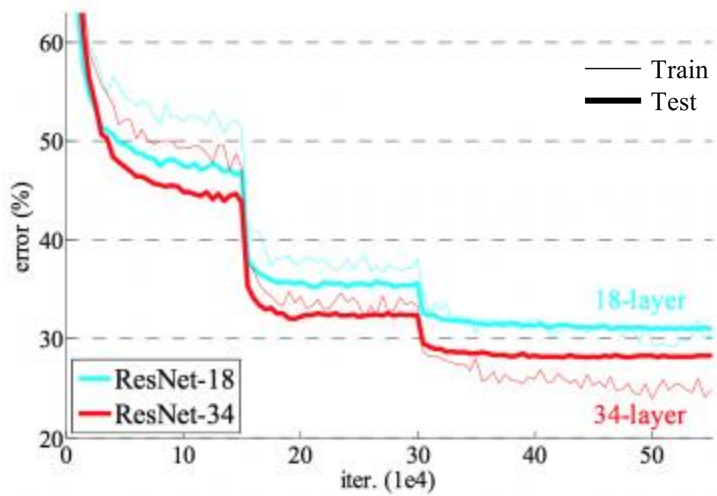


He, Zhang, Ren, & Sun (2016)

ResNet innovations

- Residual block
  - Simplifies the learning problem by making it easier for networks to learn identity mapping
  - Allows deeper networks to improve accuracy

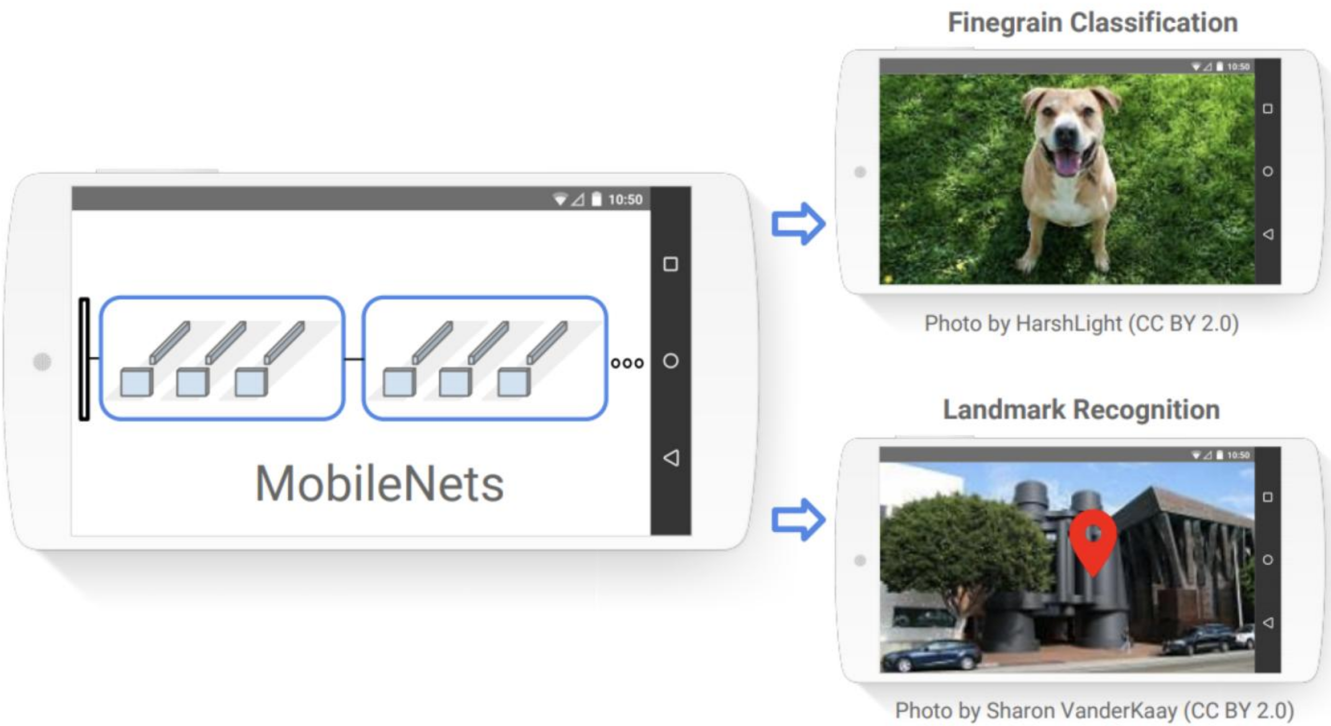




He, Zhang, Ren, & Sun (2016)

MobileNets

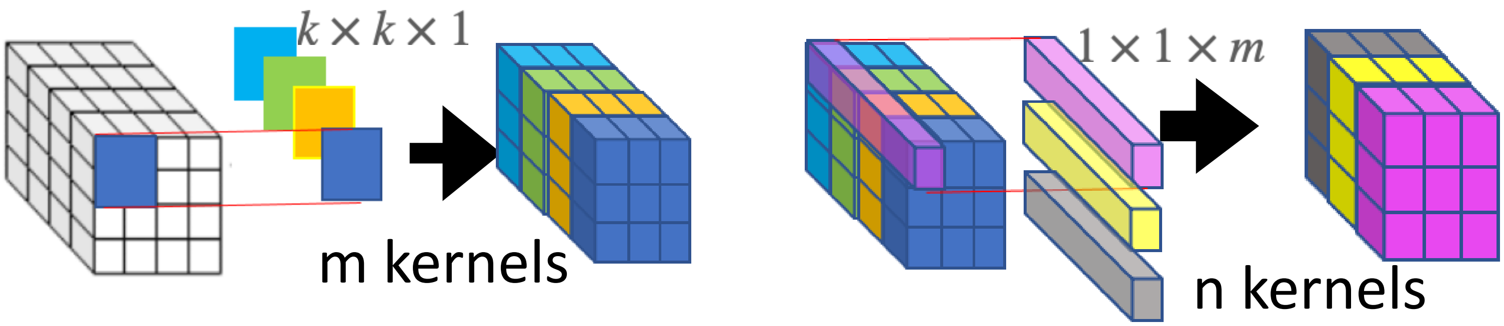
- Lightweight architecture for mobile apps



Howard, et al. (2017)

MobileNets

- Separable filters
  - Recall that filtering with a 2D filter is equivalent to filtering with two orthogonal 1D filters
  - Similarly, filtering with a 3D filter is equivalent to filtering with a 2D filter and an orthogonal 1D filter
- MobileNets uses depthwise-separable filters – 2D filters in x,y and 1D filters over channels



MobileNets innovations

- Depthwise separable convolution
  - Fewer parameters and less computation
  - Limits what kernels the model can learn – not all kernels are separable
- Smaller and faster than other architectures
  - Lower accuracy than VGG, ResNet, etc.
  - But better suited for real-time applications, phones

EfficientNet

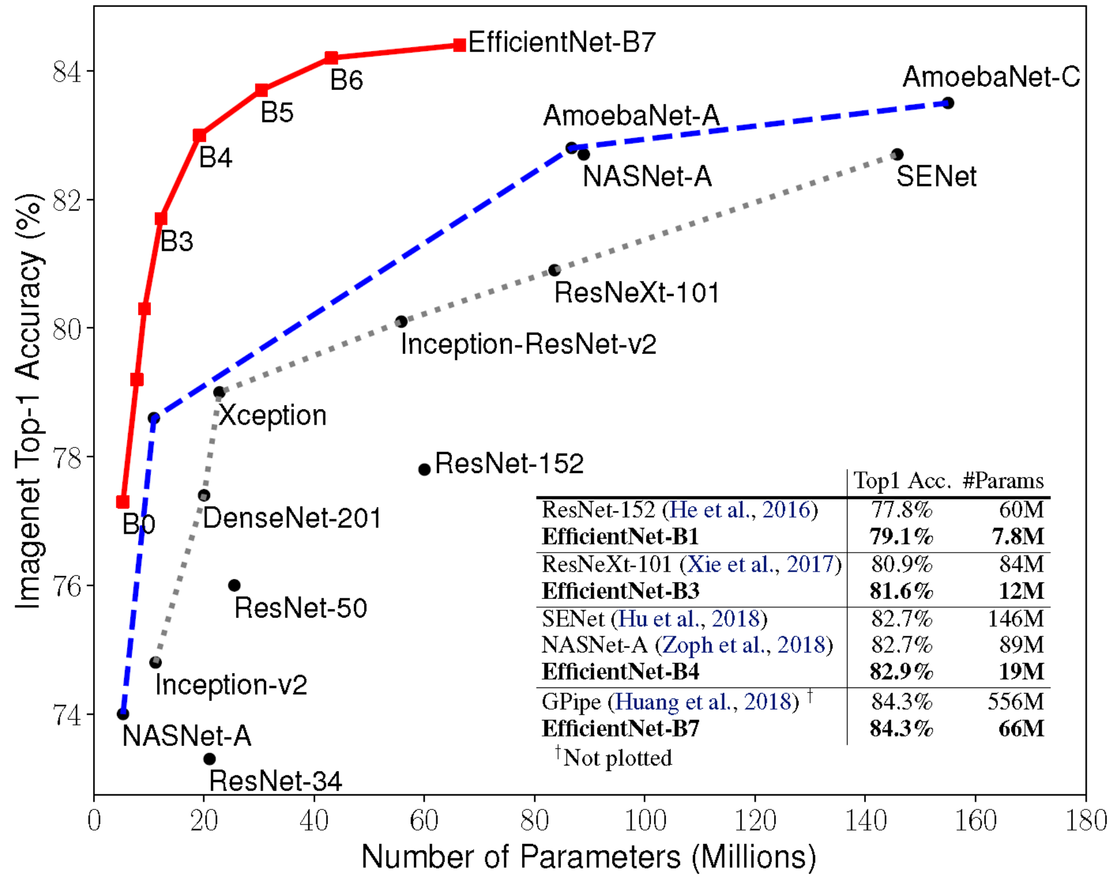
- User-defined parameters in CNNs:
  - **Depth** = number of layers
  - **Width** = number of kernels per layer
  - Kernel size
  - OR, if you use a fixed kernel size (e.g., 3x3), the equivalent parameter is **Resolution** (image size)
- Increasing any one parameter tends to improve accuracy, but with diminishing returns
- How to optimize these parameters {d, w, r}?

EfficientNet innovations

- Solution: exhaustive (grid) search!
- Specifically:
  - Assume you have fixed, limited computational resources and find the best {d, w, r} for a low-resource model
  - Computational complexity of a CNN scales linearly with depth and quadratically with width and resolution, so find values such that:
    $$d \times w^2 \times r^2 \approx 2, d \geq 1, w \geq 1, r \geq 1$$
  - To build a larger models, scale all parameters by an exponent φ (which means computational resources increase by 2^{φ})

EfficientNet innovations

- Effective way to optimise CNN parameters
- Grid searching parameters for large networks might give even higher accuracy (but is too slow)



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

† Not plotted

Tan & Le (2019)

Summary

- Many different CNN architectures image recognition
- Common themes:
  - Optimise user-defined parameters (number of layers, number of kernels, kernel size)
  - Improve efficiency
  - Improve feature learning
- Choice of architecture depends on your application
  - Runtime, memory, processing power

Classification results

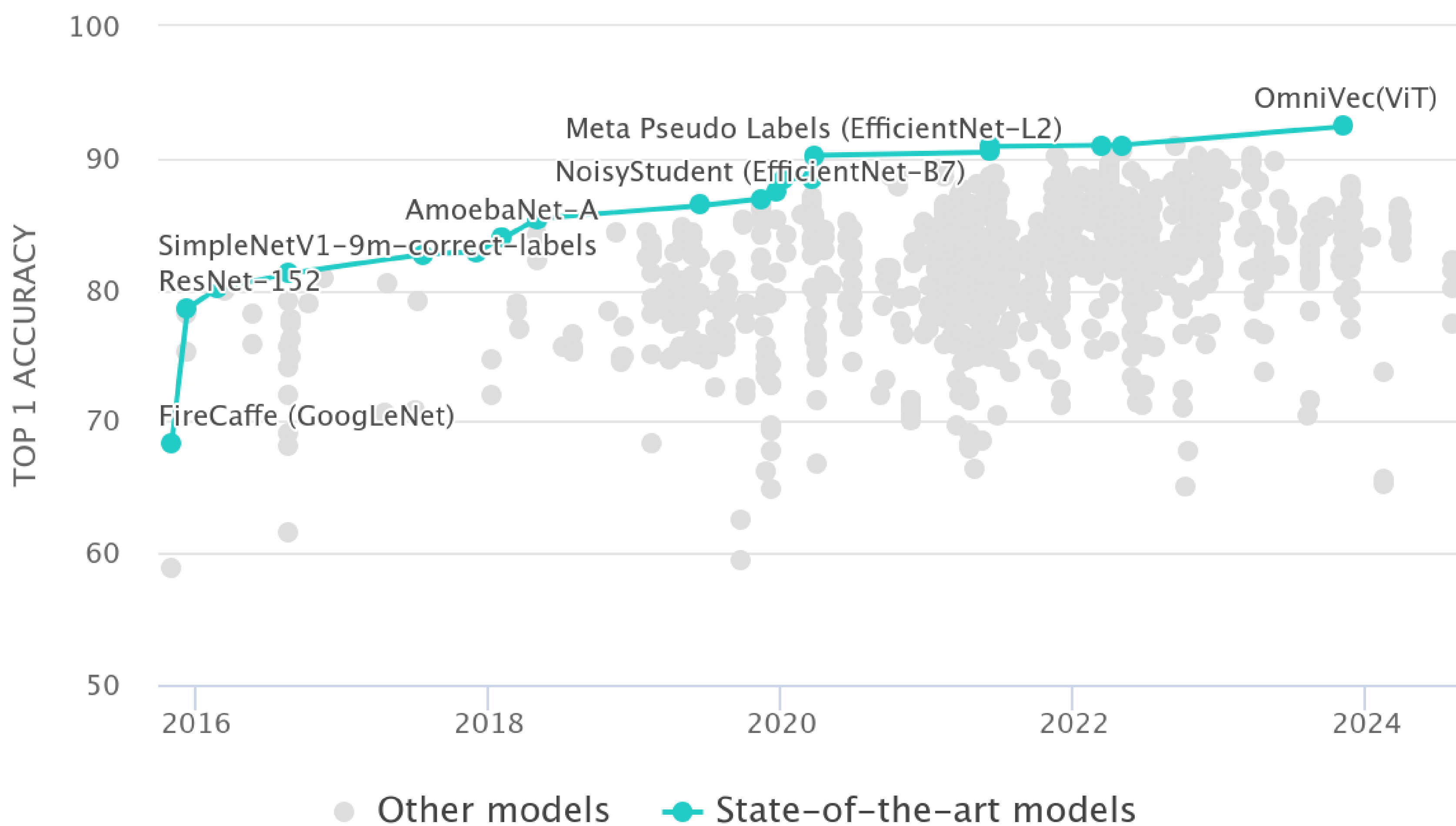
ImageNet classification

- 1000 object classes
- Model output = a probability distribution (from softmax) over 1000 class labels
- Top-1 accuracy
  - For each test image, model is correct if the most likely class == ground truth class
- Top-N accuracy
  - For each test image, model is correct if any of the N most likely classes == ground truth class

Classification performance

CNN Architecture	Layers	Top-5 error
AlexNet	8	16.4%
VGG-19	19	7.3%
GoogleNet	22	6.7%
ResNet	152	3.57%

Classification performance



https://paperswithcode.com/sota/image-classification-on-imagenet

Classification errors

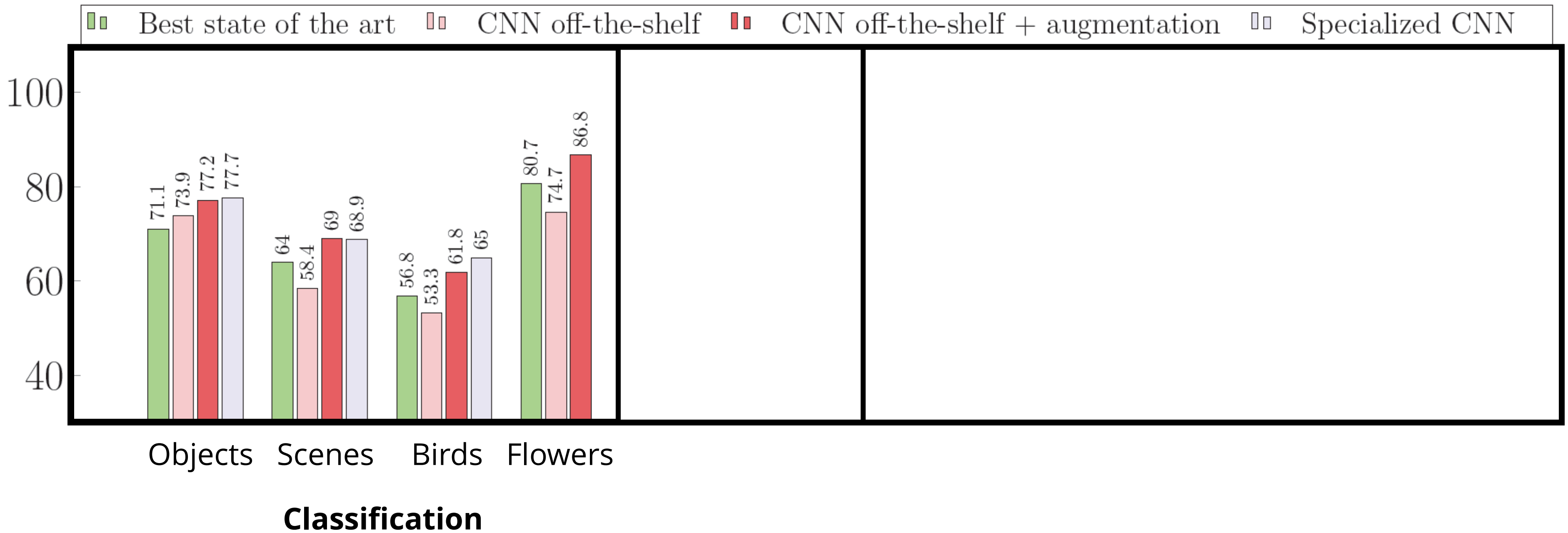
ImageNet Classification Failures (GoogLeNet 2014)

ruler	**king crab**	**sidewinder**	**saltshaker**	**reel**	**hatchet**
pencil box	pizza	maze	pill bottle	stethoscope	vase
rubber eraser	strawberry	gar	water bottle	whistle	pitcher
ballpoint pen	orange	valley	lotion	ice lolly	coffeepot

Russakovsky et al. (2014)

Generalisation

- Features from neural networks are good representations for a range of tasks



Razavian et al. (2014)

Summary

- CNNs are the state-of-the-art for image classification, exceeding human performance on ImageNet
- CNN classification errors are often understandable (odd views, small objects), which suggests they learn reasonable features for this task

Transfer learning

Image recognition: Pixels

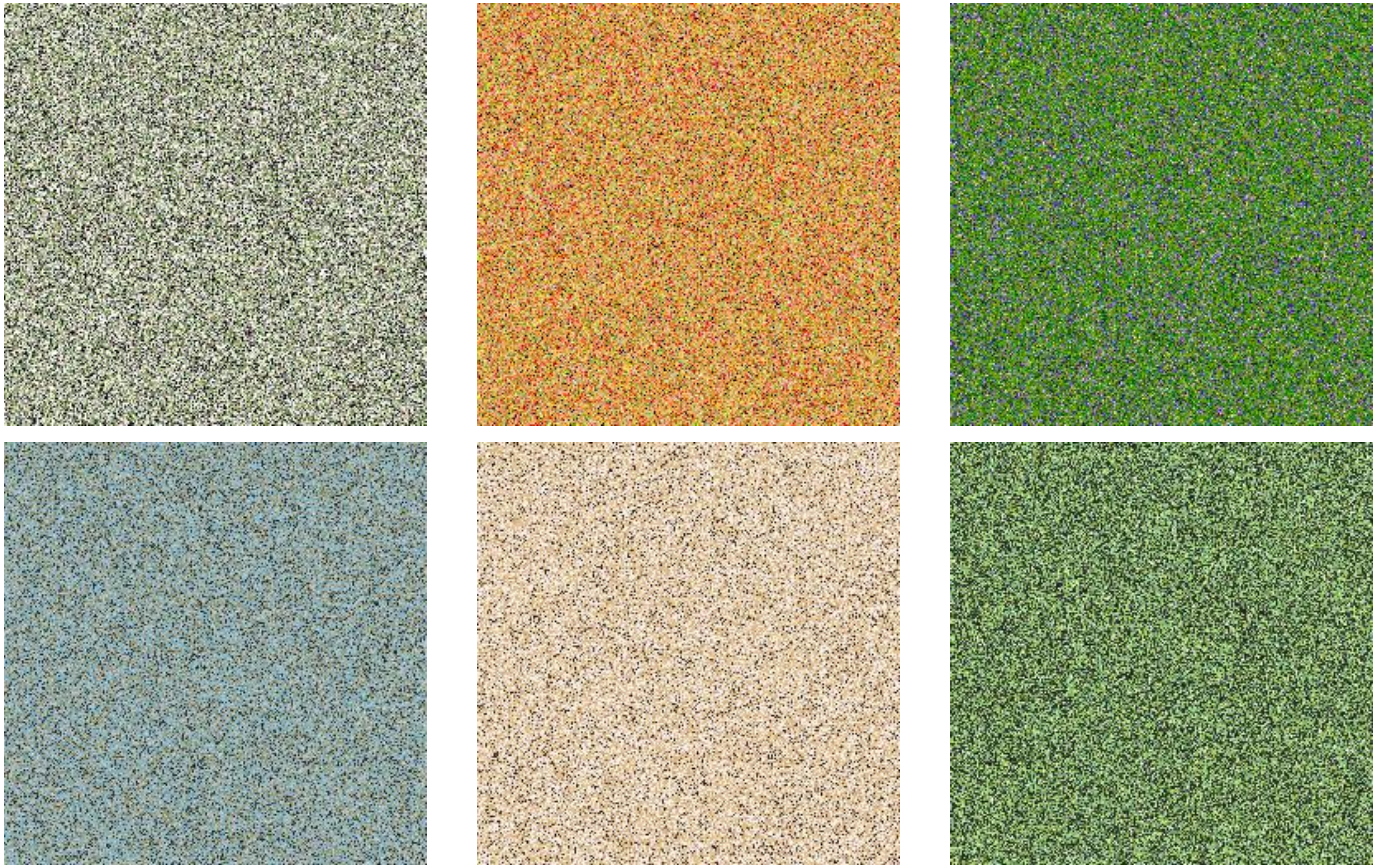


Image recognition: Pixels

- Pixels are a poor space for classification
  - High-dimensional space: 256 x 256 x 3 image = 196,608 attributes
  - Irrelevant transformations (translation, lighting change, scale change, rotation, etc.) cause large changes in pixel values

Image recognition: Features

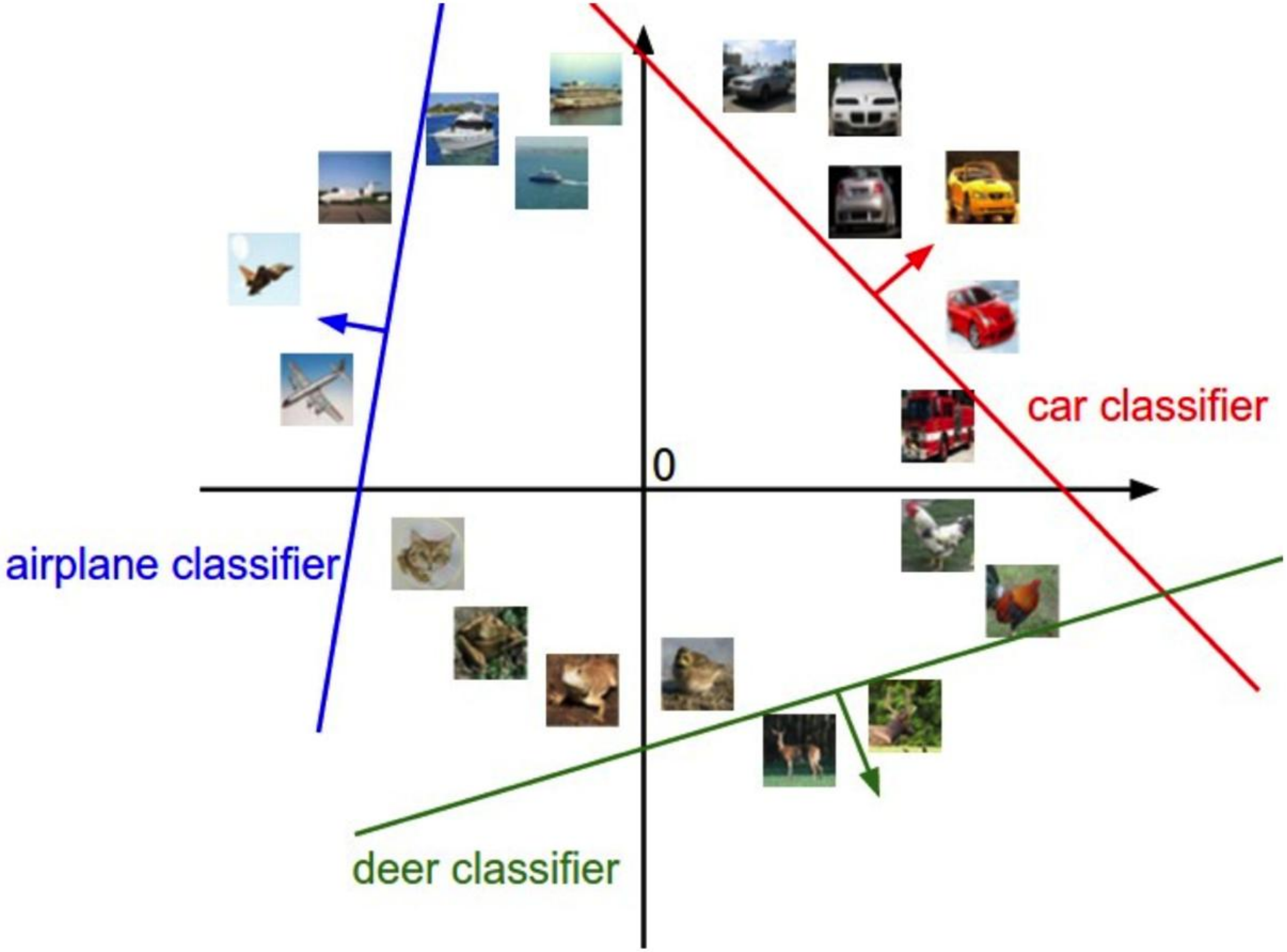


Image recognition: Features

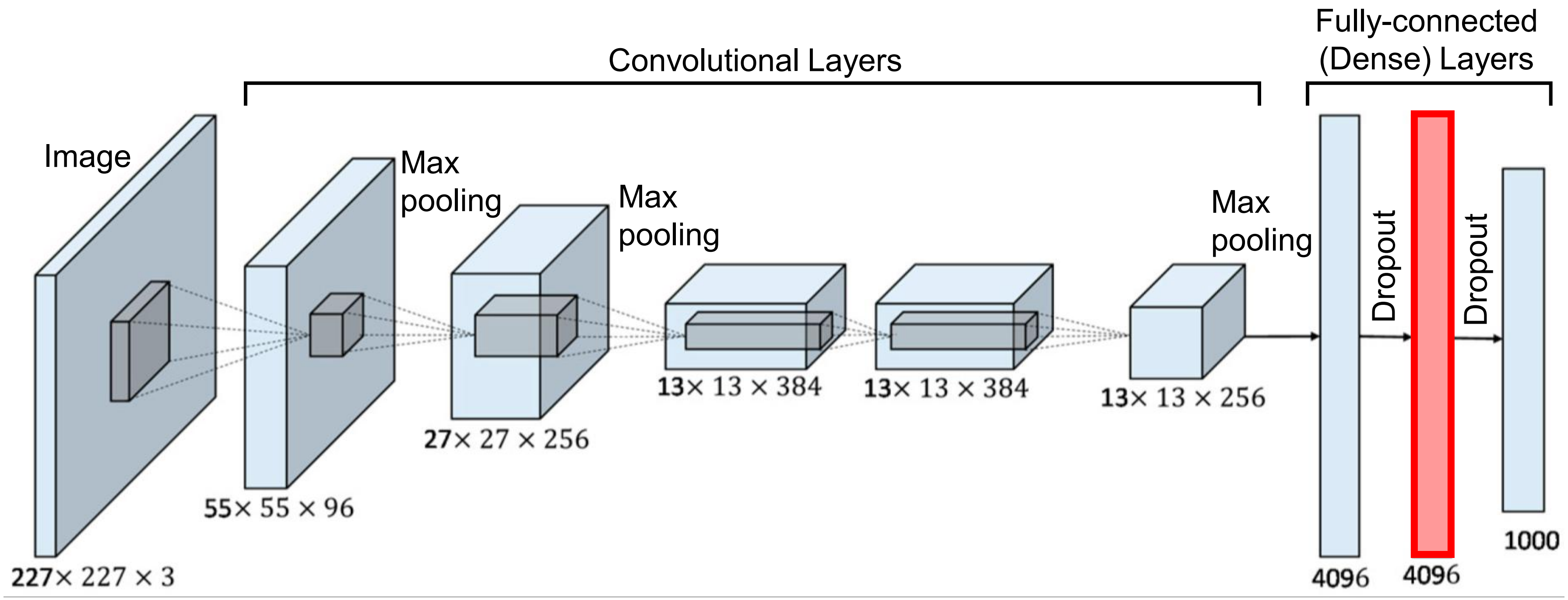
- A good feature space for image recognition:
  - Is lower-dimensional – e.g., 1000s of values per image
  - Projects images from the same class into a similar part of the space (images with the same class label have similar features)

Using pretrained networks

- CNNs convert images from pixels to high-level features that are good for classification (feature embedding)
- These high-level features give good performance on a range of computer vision tasks
- Transfer learning – use features from a CNN trained on a large-scale task (e.g., ImageNet classification) as input for another task, with minimal retraining

Transfer learning

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

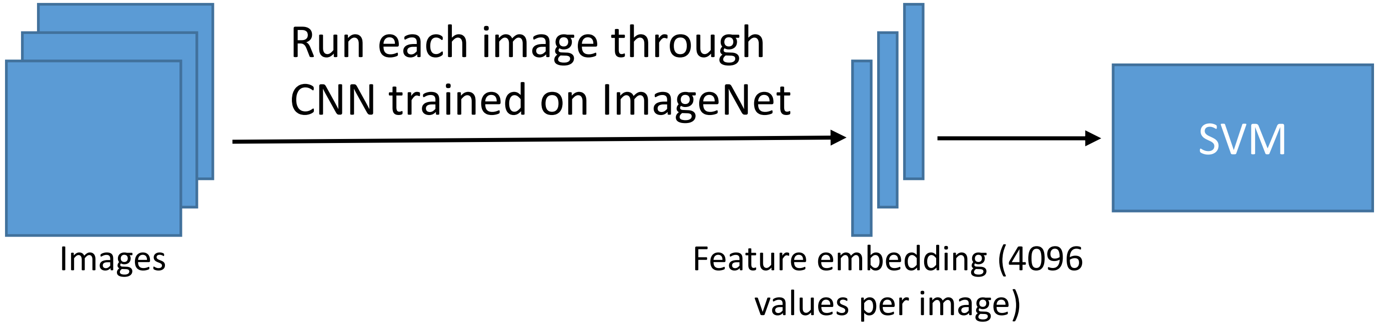


Embedding of an input = the network's response to the input at some layer

Image: Han, Zhong, Cao, & Zhang (2017)

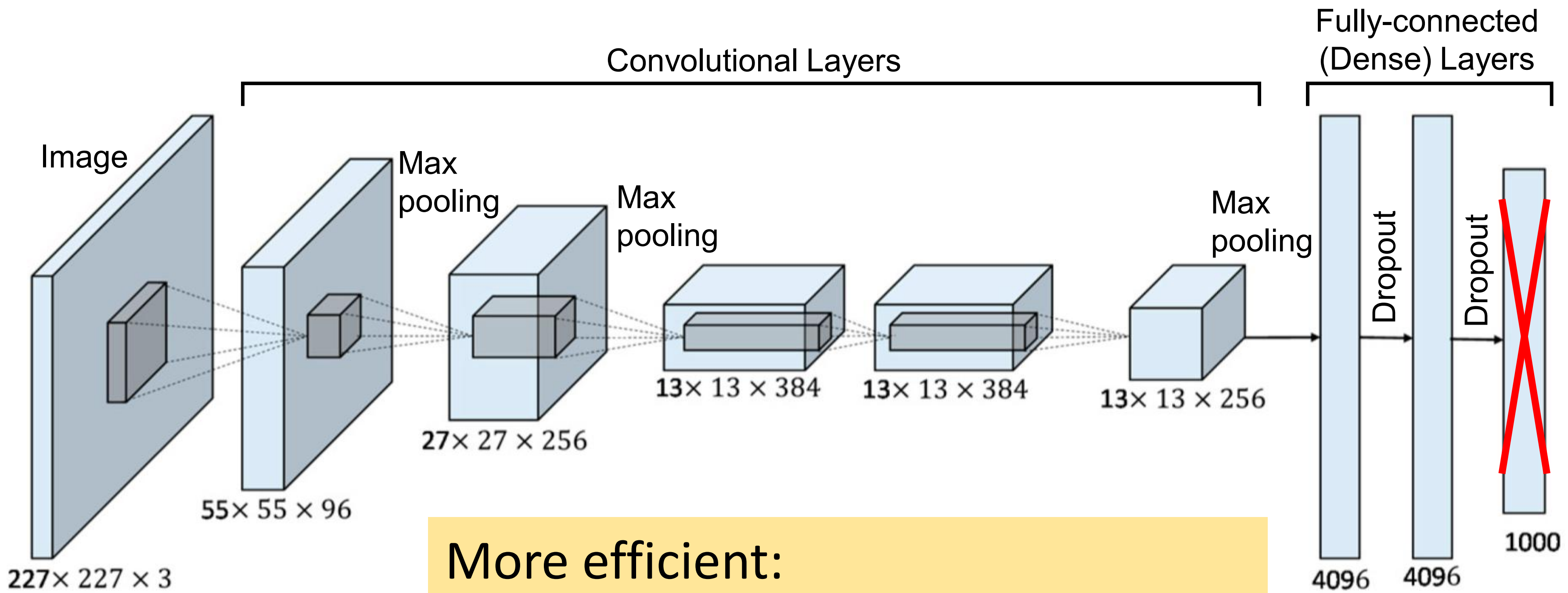
Transfer learning

- Extract the representation from a late layer of a CNN trained on ImageNet
  - E.g., for each image take the activations from the 4096 neurons that feed into the 1000-way ImageNet classification
- Use the neurons' activations as the attributes for a classifier of your choice (e.g., SVM, K-NN etc.)



Transfer learning

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

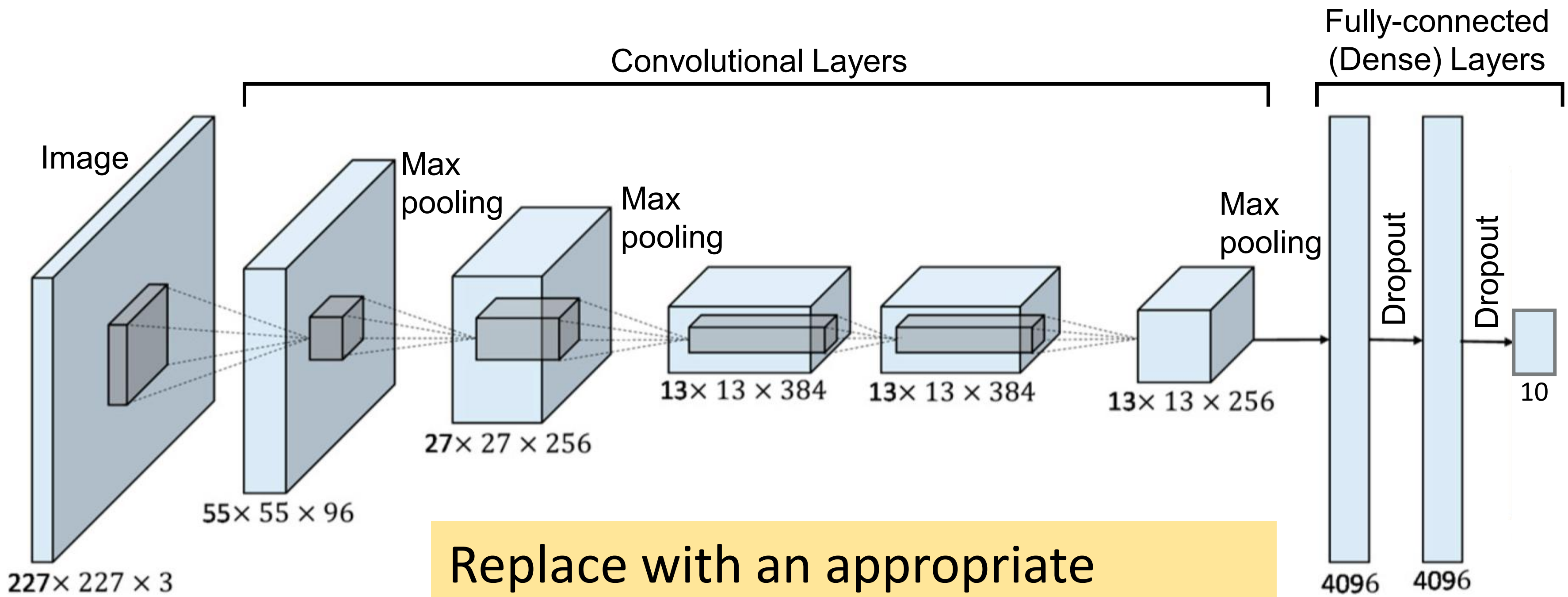


More efficient:
Remove the output layer of a CNN trained on ImageNet

Image: Han, Zhong, Cao, & Zhang (2017)

Transfer learning

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

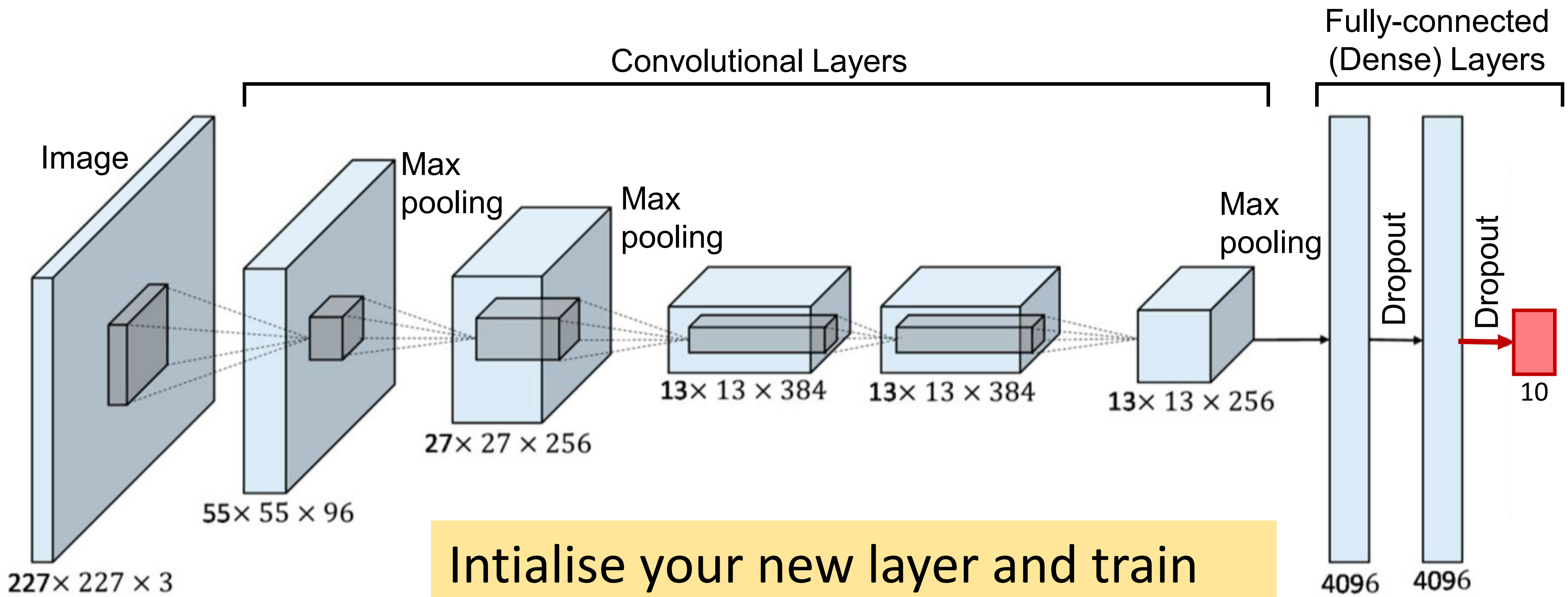


Replace with an appropriate output layer for your task

Image: Han, Zhong, Cao, & Zhang (2017)

Transfer learning

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

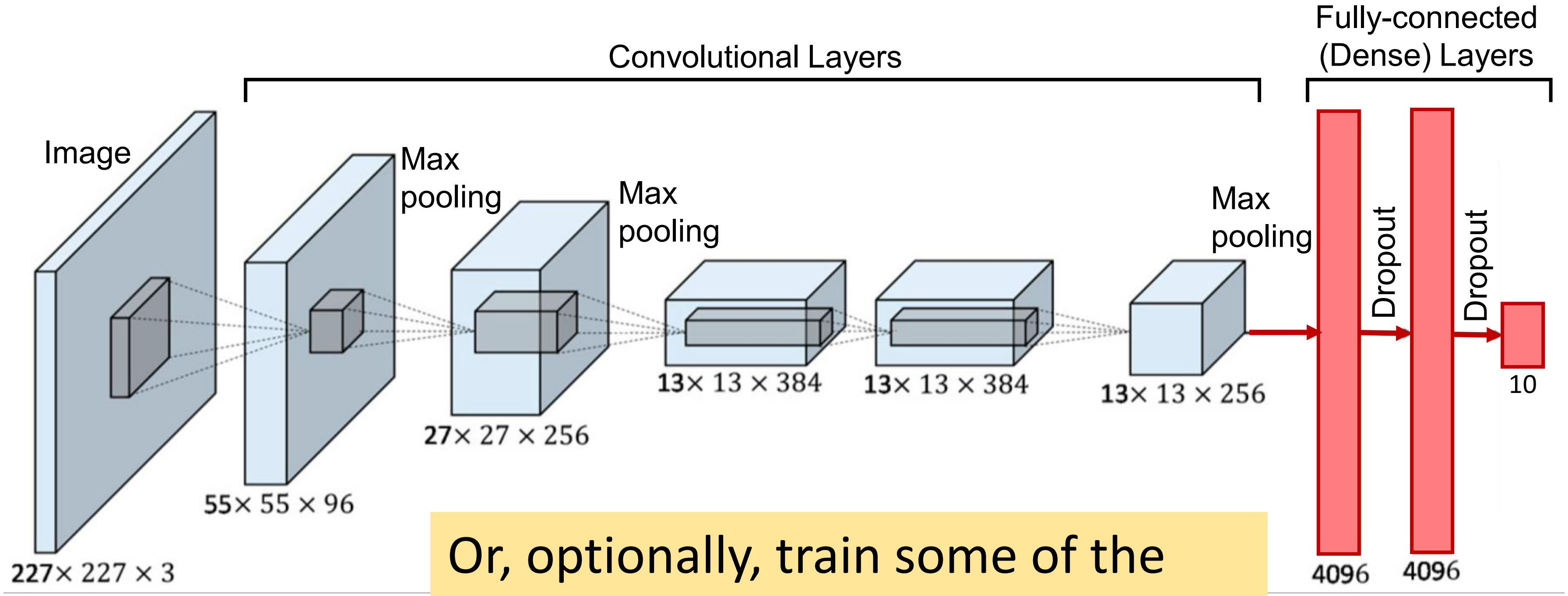


Intialise your new layer and train *only* this layer; freeze all other network parameters

Image: Han, Zhong, Cao, & Zhang (2017)

Transfer learning

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)



Or, optionally, train some of the later layers but freeze earlier layers

Image: Han, Zhong, Cao, & Zhang (2017)

Retraining layers

- **Finetuning** = retraining layers of a pretrained CNN
- How many layers to fine tune depends on dataset size and how similar it is to ImageNet
  - More dissimilar datasets may need more retraining of lower layers
  - If dataset size is limited, training lower layers may just lead to overfitting

Summary

- ImageNet-trained CNNs produce state-of-the-art performance on image recognition tasks
- It’s common to use CNNs pretrained on ImageNet for a variety of computer vision tasks, either as-is (“off the shelf” feature embedding) or with some finetuning