EVML

MORE CNN
HANDSON

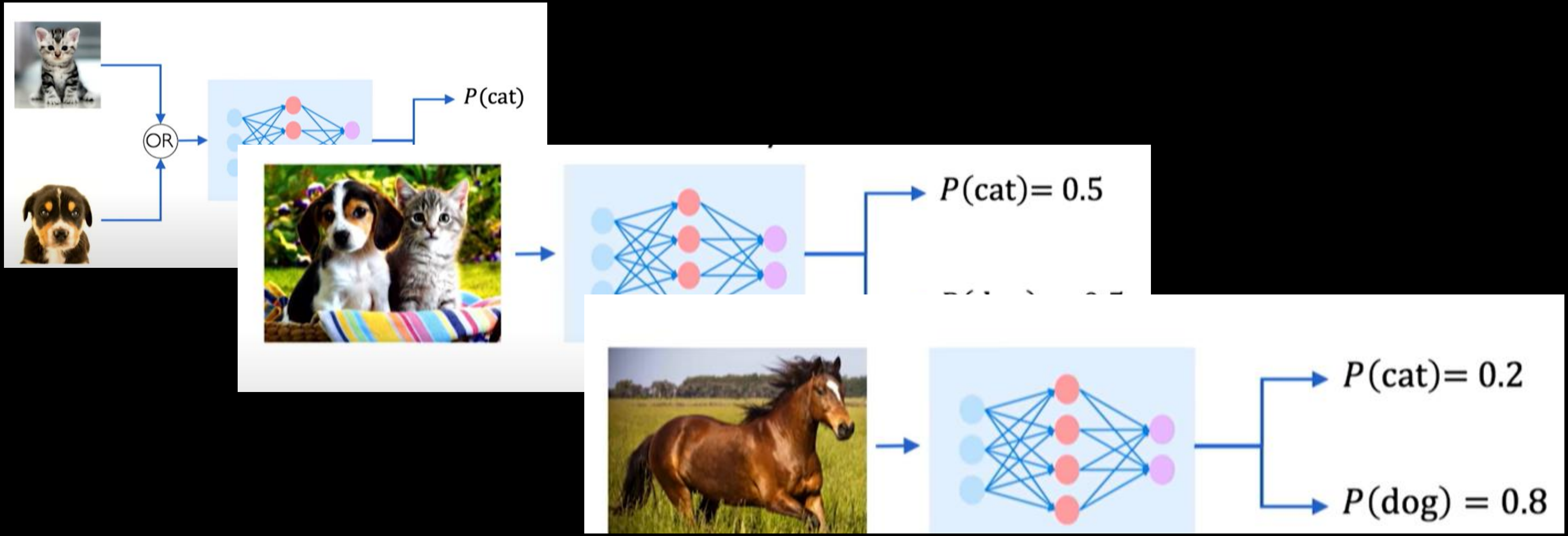
JEROEN VEEN

HAN_UNIVERSITY OF APPLIED SCIENCES

AGENDA

- Limitations of neural nets
- Either restart the Keras tuner exercise
- Or build your own CNN
- Or starting with transfer learning

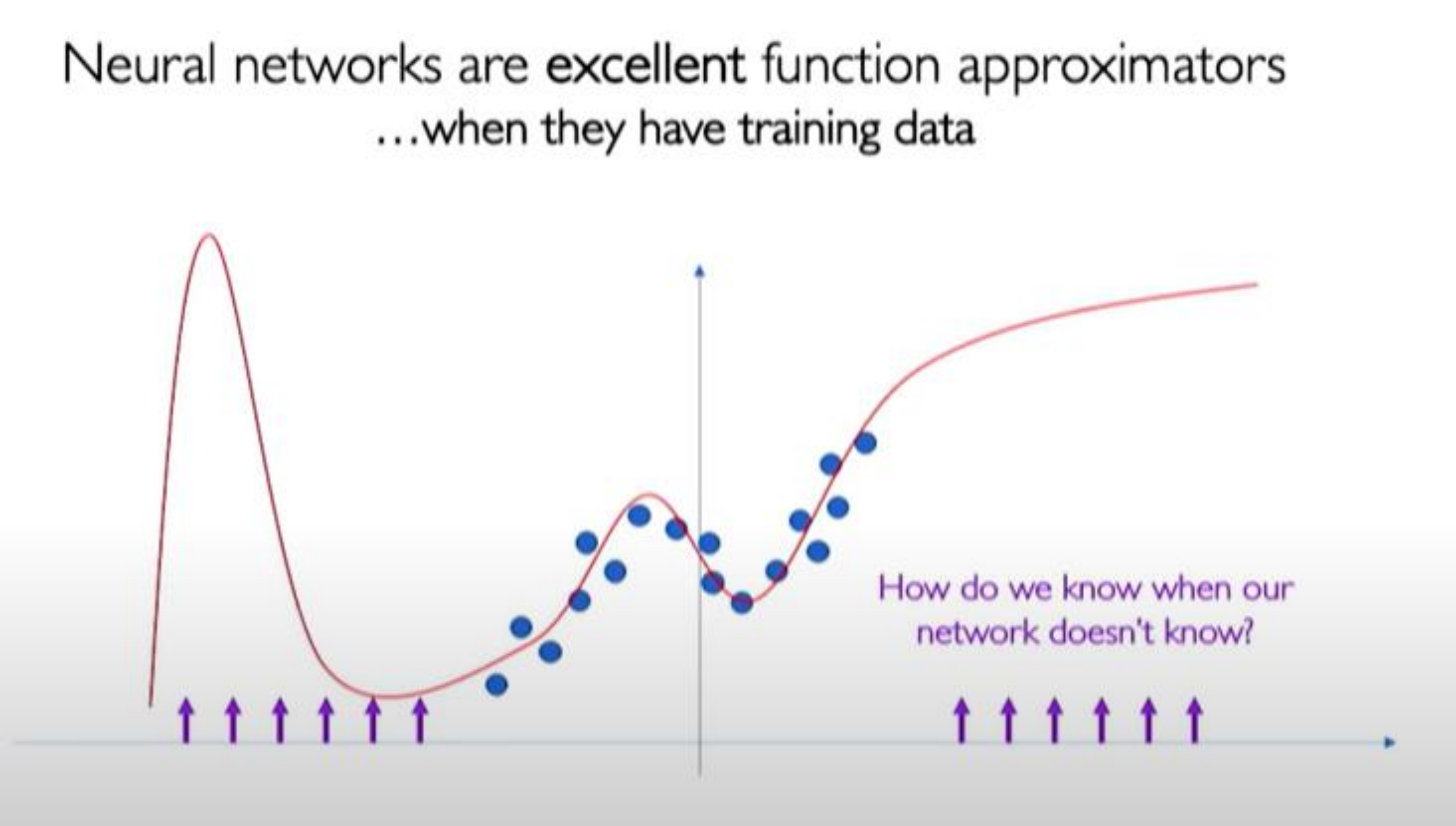
LIMITATIONS OF NEURAL NETS



Poor at representing uncertainty, difficult to trust

Source: MIT Deep Learning 6.S191 (introtodeeplearning.com)

LIMITATIONS



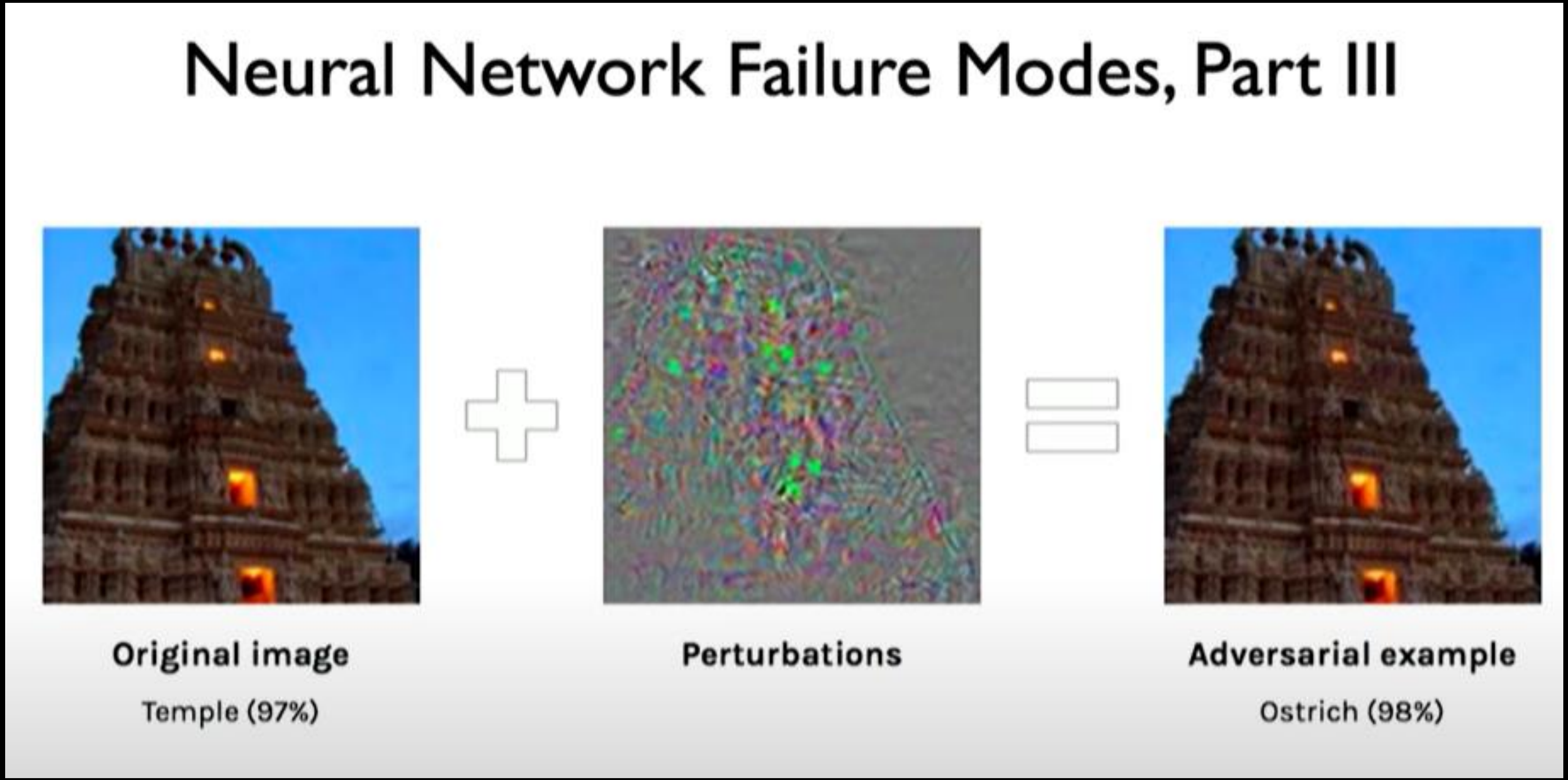
Neural networks are **excellent** function approximators

…when they have training data

How do we know when our network doesn't know?

Source: MIT Deep Learning 6.S191 (introtodeeplearning.com)

LIMITATIONS OF NEURAL NETS



Neural Network Failure Modes, Part III

Original image

Temple (97%)

Perturbations

Adversarial example

Ostrich (98%)

Easily fooled by adversarial examples

Source: MIT Deep Learning 6.S191 (introtodeeplearning.com)

USING THE KERAS TUNER

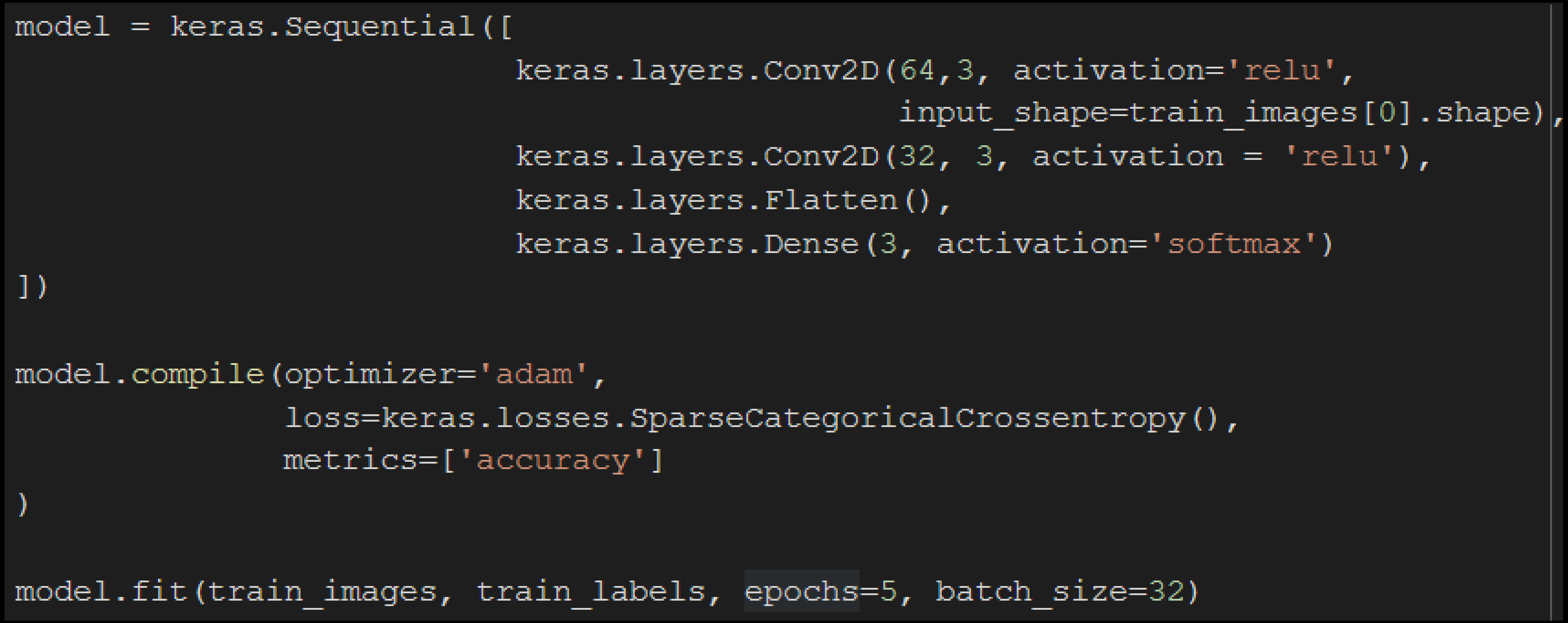
- Please study: https://www.tensorflow.org/tutorials/keras/keras_tuner

Use cross-validation to re-evaluate model with the optimal hyperparameters!

- Additional resources: https://www.youtube.com/watch?v=O85gh3OzluI

BUILD YOUR FIRST CNN

- Train a naive CNN

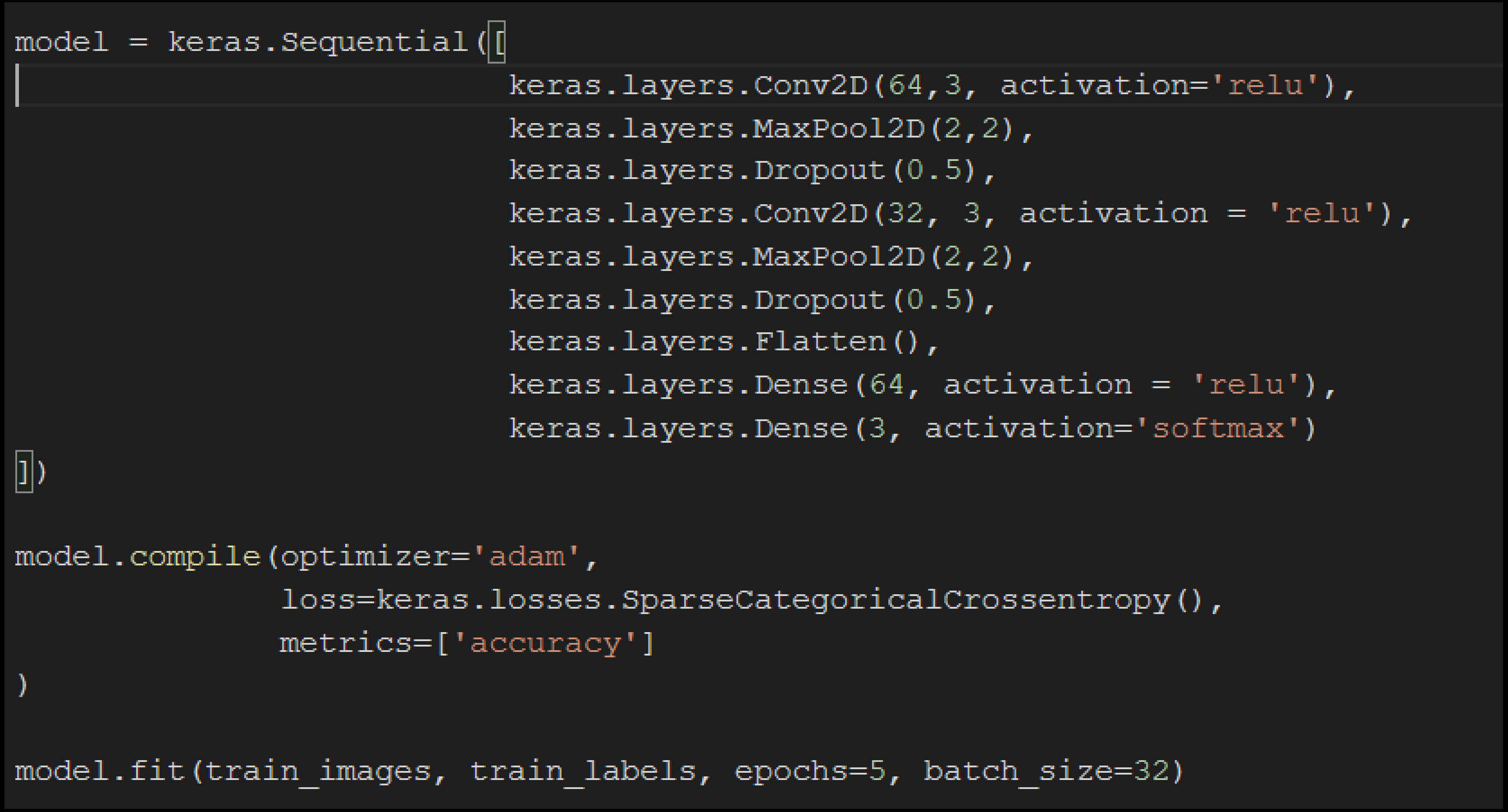


```
model = keras.Sequential([
                          keras.layers.Conv2D(64,3, activation='relu',
                                              input_shape=train_images[0].shape),
                          keras.layers.Conv2D(32, 3, activation = 'relu'),
                          keras.layers.Flatten(),
                          keras.layers.Dense(3, activation='softmax')
])

model.compile(optimizer='adam',
              loss=keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy']
)

model.fit(train_images, train_labels, epochs=5, batch_size=32)
```

MORE SOPHISTICATED CNN



```
model = keras.Sequential([
                          keras.layers.Conv2D(64,3, activation='relu'),
                          keras.layers.MaxPool2D(2,2),
                          keras.layers.Dropout(0.5),
                          keras.layers.Conv2D(32, 3, activation = 'relu'),
                          keras.layers.MaxPool2D(2,2),
                          keras.layers.Dropout(0.5),
                          keras.layers.Flatten(),
                          keras.layers.Dense(64, activation = 'relu'),
                          keras.layers.Dense(3, activation='softmax')
])

model.compile(optimizer='adam',
              loss=keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy']
)

model.fit(train_images, train_labels, epochs=5, batch_size=32)
```

RESNET MODEL

- https://www.kaggle.com/dansbecker/transfer-learning
- https://youtu.be/mPFq5KMxKVw

MOBILENETS

- Efficient Convolutional Neural Networks for Mobile Vision Applications

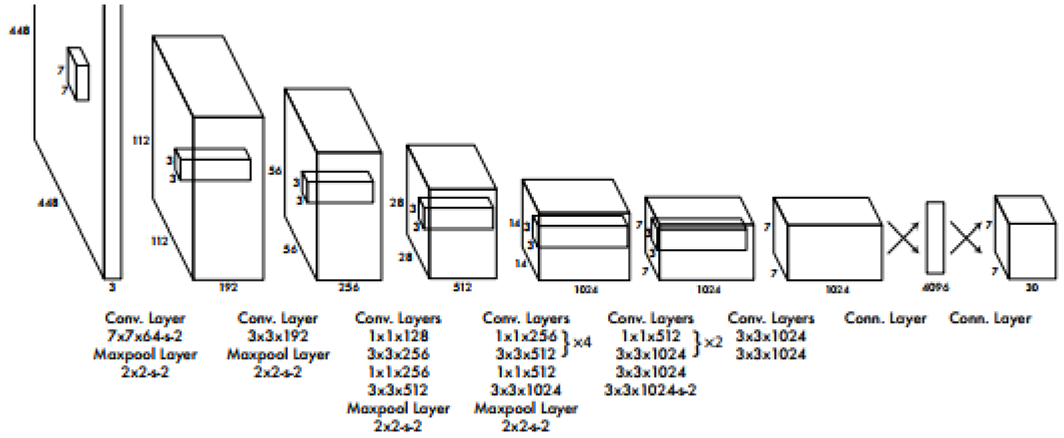


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 × 1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 × 224 input image) and then double the resolution for detection.

Source: A Howard et al. 2017, https://arxiv.org/pdf/1704.04861.pdf

VARIANTS

- V2: M. Sandler et al. 2019, https://arxiv.org/pdf/1801.04381.pdf
- MobileNet-Tiny, https://nitheshsinghsanjay.github.io/
- Single-Shot Multibox Detector (SSD)

Source:

RETRAINING AN IMAGE CLASSIFIER

- https://www.tensorflow.org/hub/tutorials/tf2_image_retraining
- Image size
- Normalization
- Data generator