

Copyright Notice

These slides are distributed under the Creative Commons License.

[DeepLearning.AI](#) makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite [DeepLearning.AI](#) as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>

Image Classification

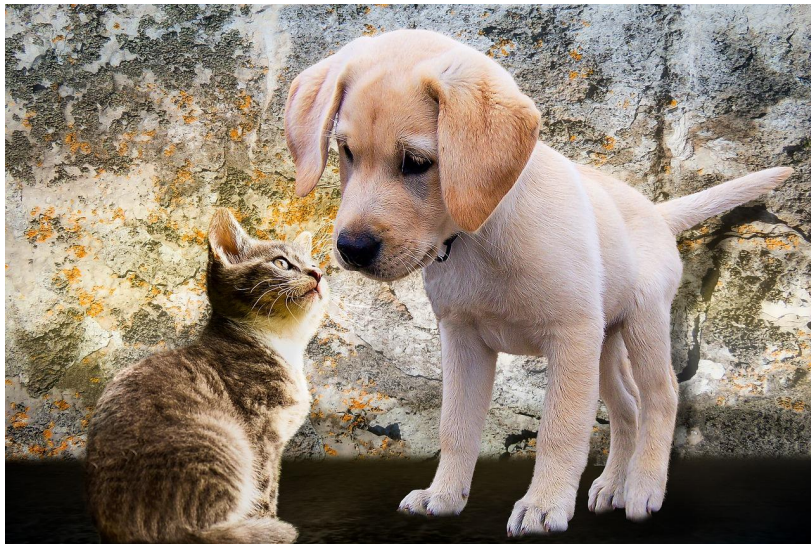
- Multi-class Classification
 - Binary Classification (Subset of the problem)
- Multi-label Classification

Image Classification



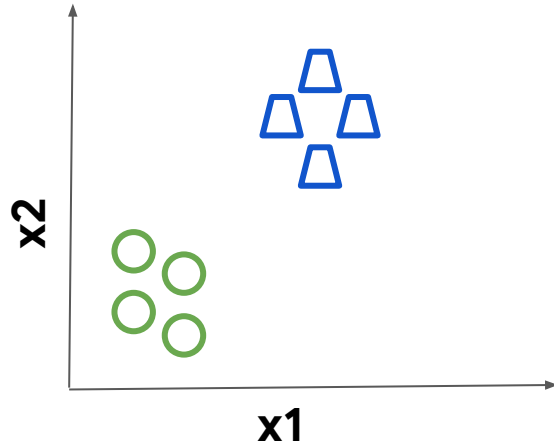
CAT

Multi-label Classification



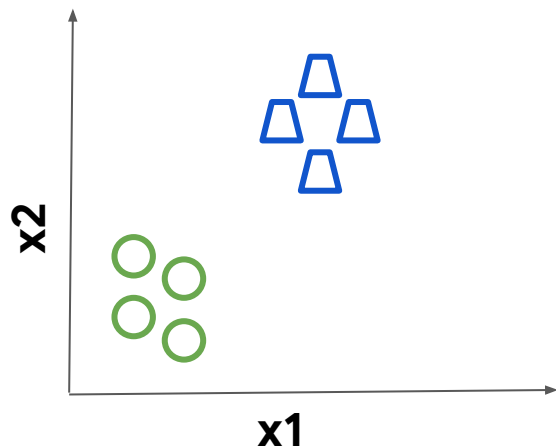
CAT, DOG

Binary vs. Multi Class Classification

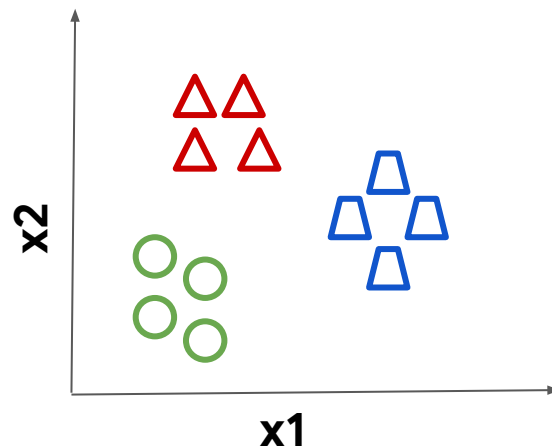


Binary Classification

Binary vs. Multi Class Classification

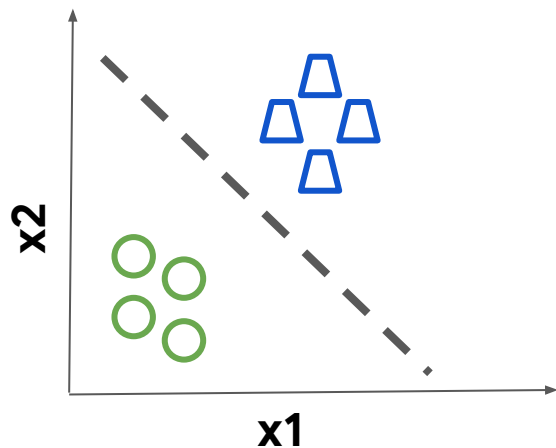


Binary Classification

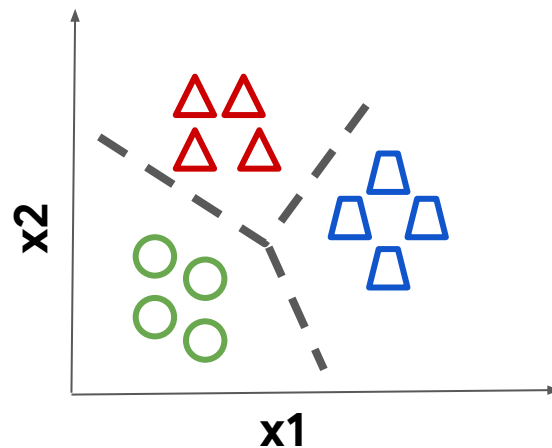


Multi-class Classification

Binary vs. Multi Class Classification

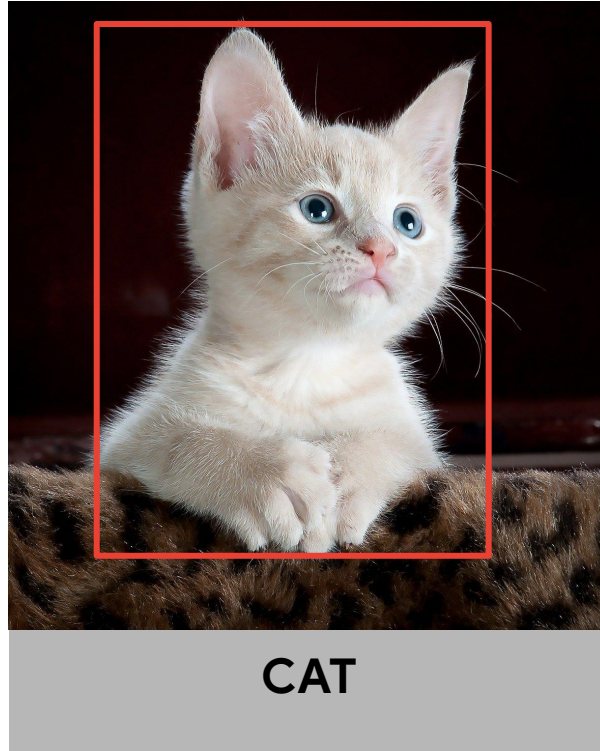


Binary Classification



Multi-class Classification

Object Localization



<https://pixabay.com/photos/kitty-cat-kitten-pet-animal-cute-551554/>

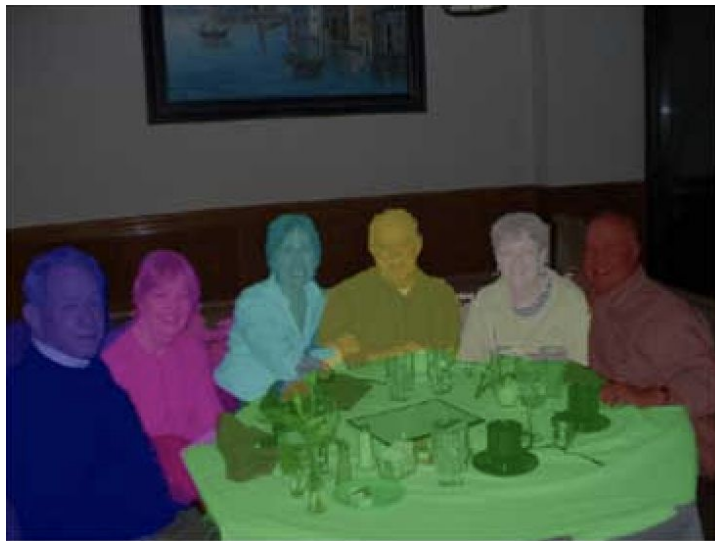
Object Detection



Object Detection

- For each object:
 - confidence scores
 - bounding boxes.
- Popular algorithms
 - R-CNN
 - Faster-RCNN
 - YOLO
 - SSD

Image Segmentation



[source](#)

Semantic vs. Instance Segmentation

Semantic vs. Instance Segmentation

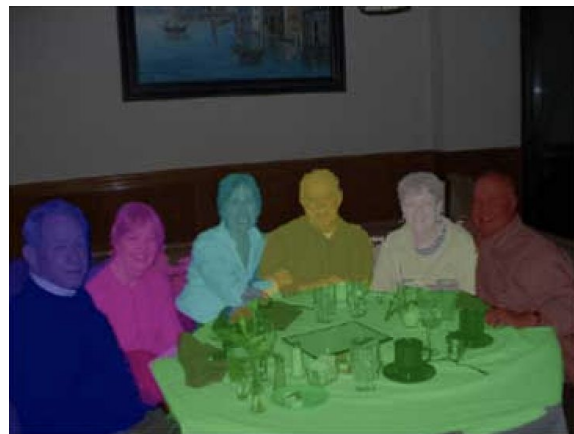


Semantic Segmentation

Semantic vs. Instance Segmentation



Semantic Segmentation



Instance Segmentation

Semantic Segmentation

- Same class \rightarrow one segment.

Semantic Segmentation

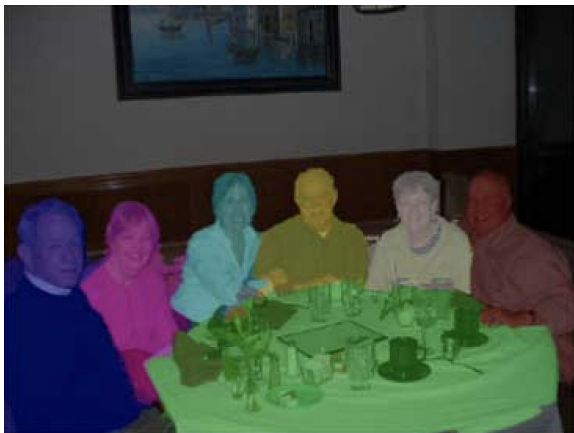
- Same class → one segment.
- Each pixel is associated with one class.
 - All person(s) in an image are treated as one segment

Semantic Segmentation

- Same class → one segment.
- Each pixel is associated with one class.
 - All person(s) in an image are treated as one segment
- Popular models are Fully Convolutional Neural Networks, U-Net, DeepLab

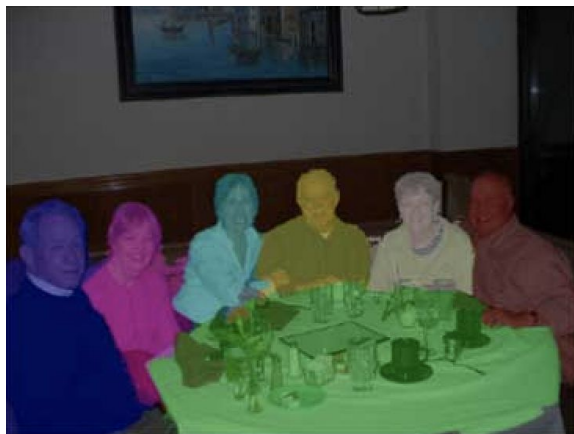
Instance Segmentation

- Multiple “instances” of same class are separate segments.



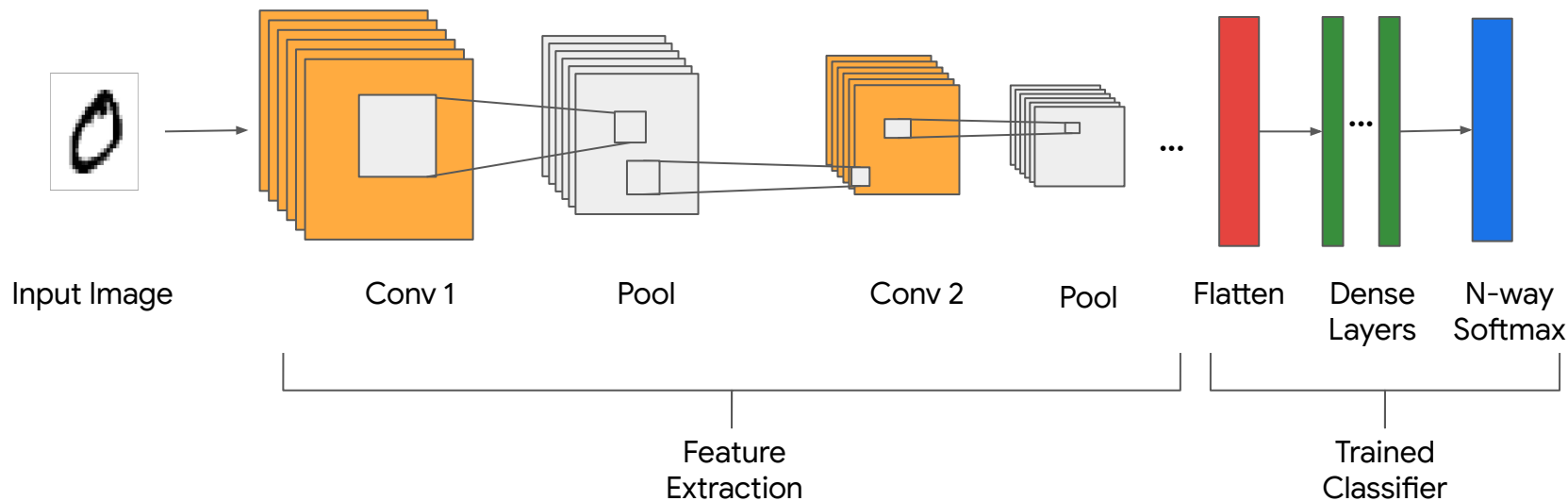
Instance Segmentation

- Multiple “instances” of same class are separate segments.

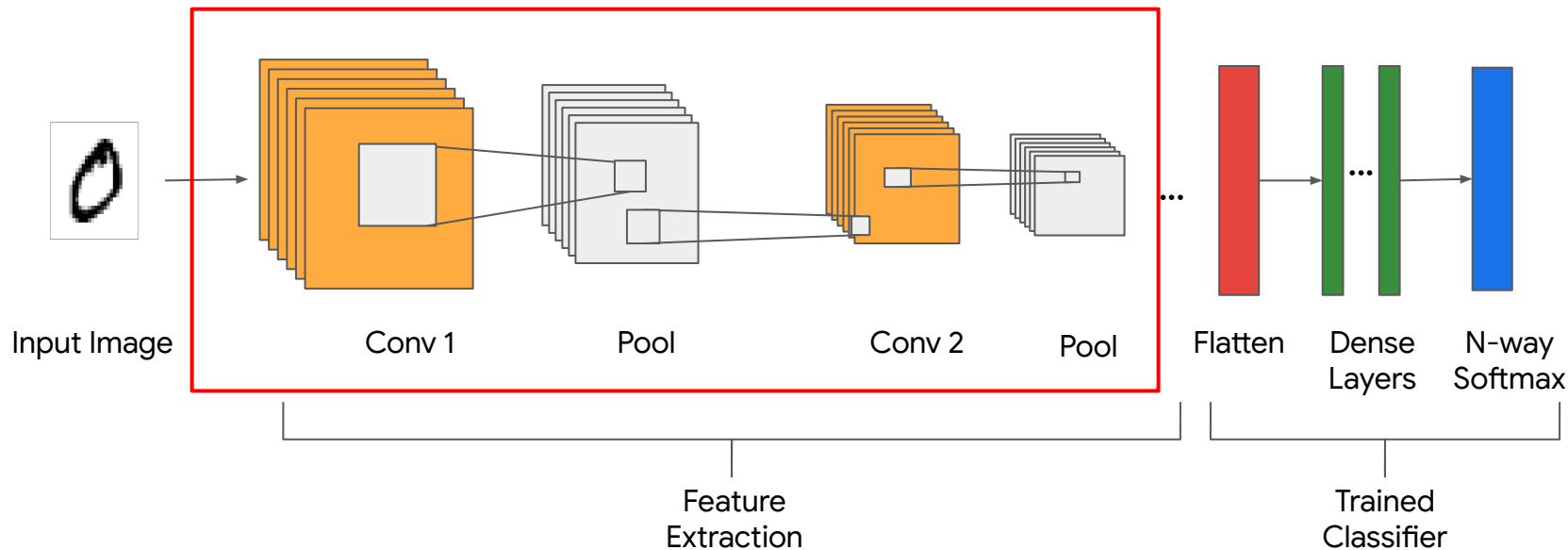


- Popular algorithms are [Mask R-CNN](#).

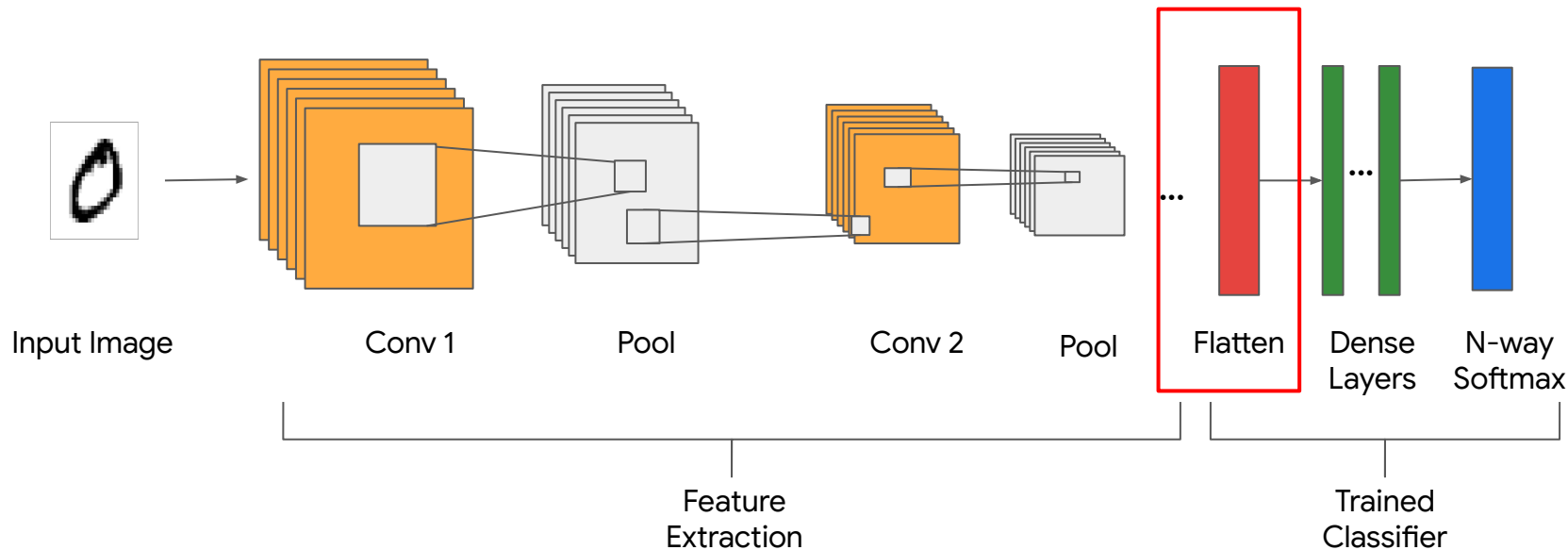
Convolutional Neural Networks Architecture



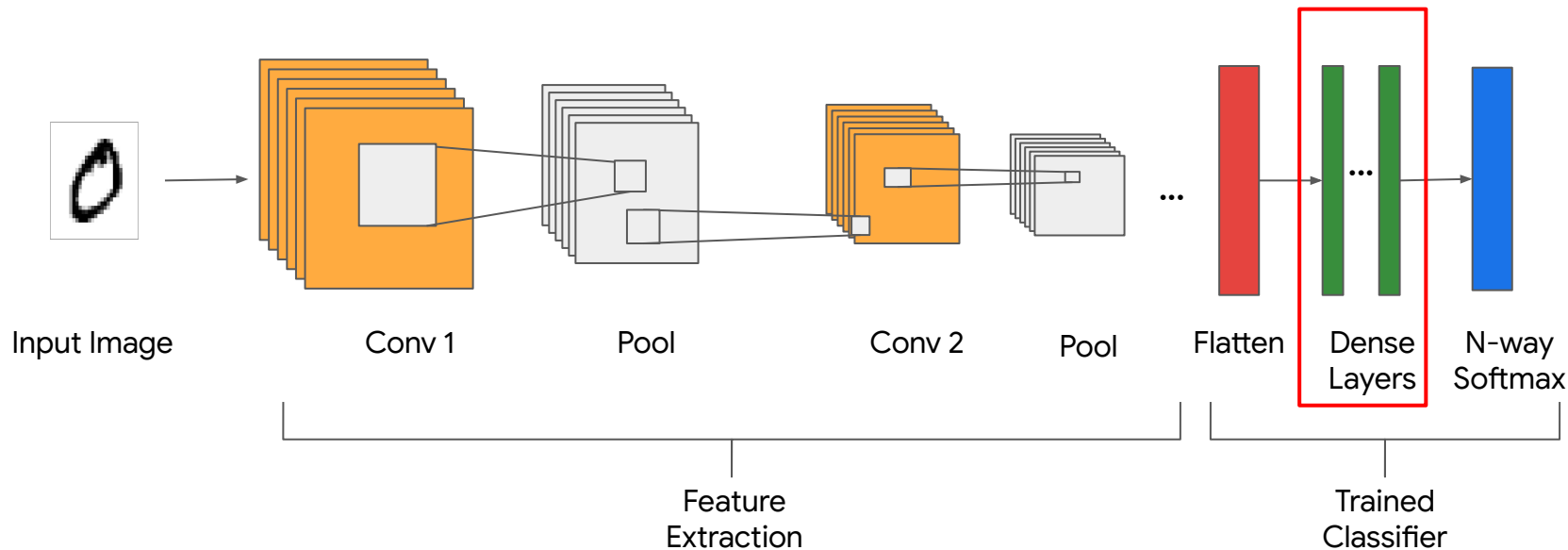
Convolutional Neural Networks Architecture



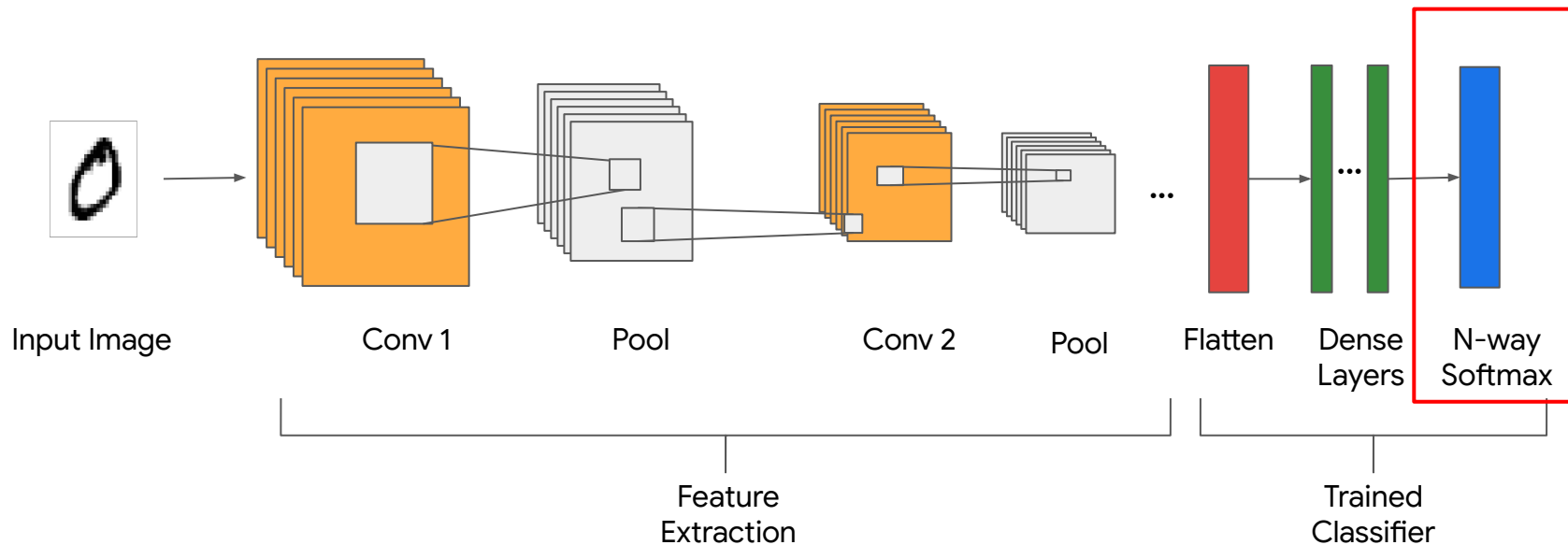
Convolutional Neural Networks Architecture



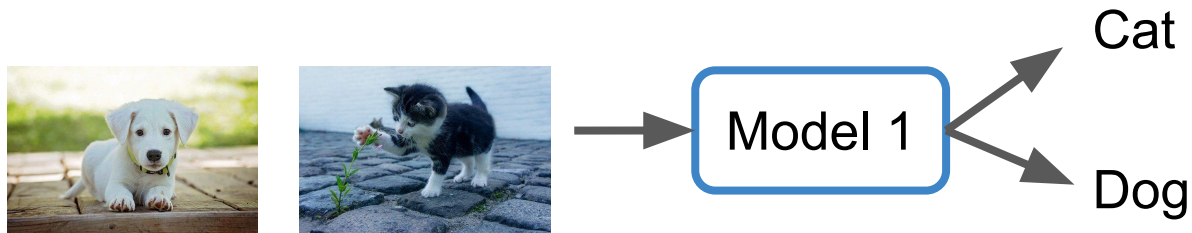
Convolutional Neural Networks Architecture



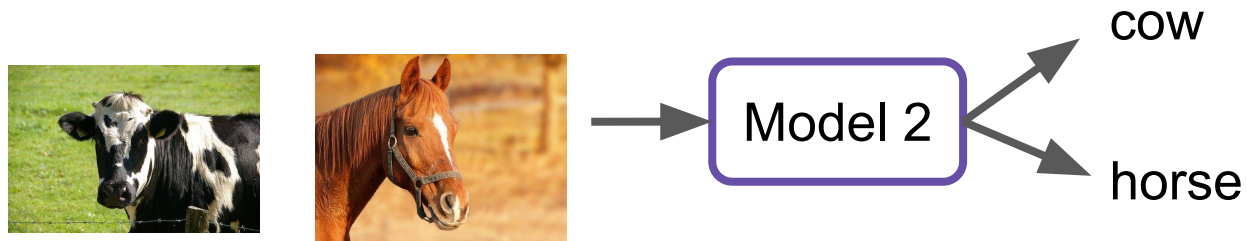
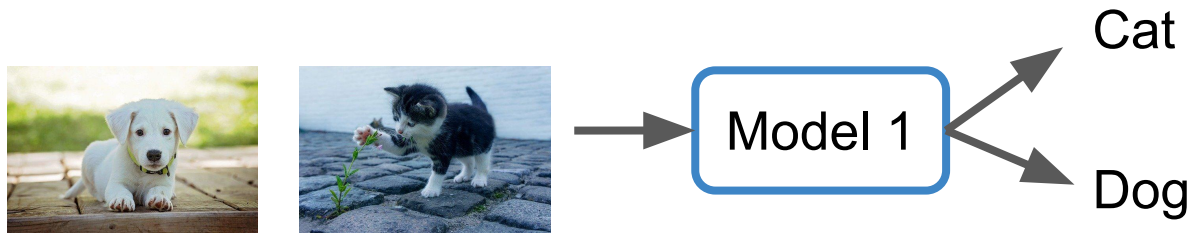
Convolutional Neural Networks Architecture



Without Transfer Learning

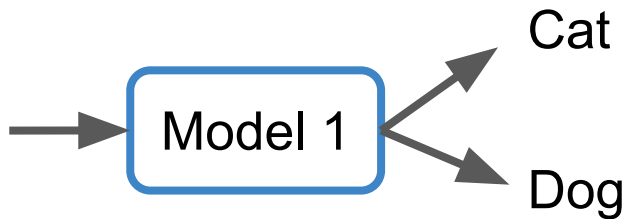


Without Transfer Learning

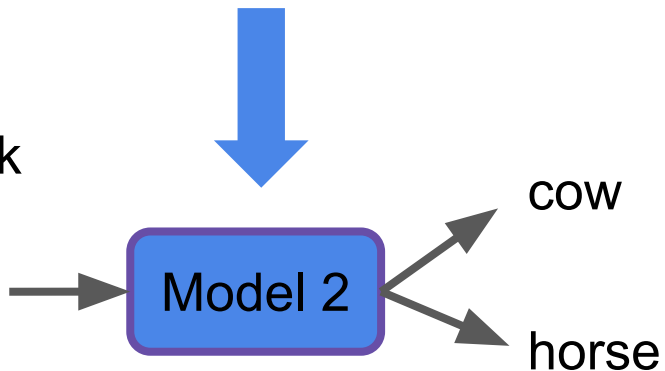


Re-use past learning

Pre-train



Train on downstream task



What is Transfer Learning?

- Pre trained model reused for solving another task.
- Reuse weights and layers.
- Example: pre-trained MobileNetV2 → cats vs dog classifier.

What is Transfer Learning?

- Pre trained model reused for solving downstream task.
- Reuse weights and layers.
- Example: pre-trained MobileNetV2 → cats vs dog classifier.

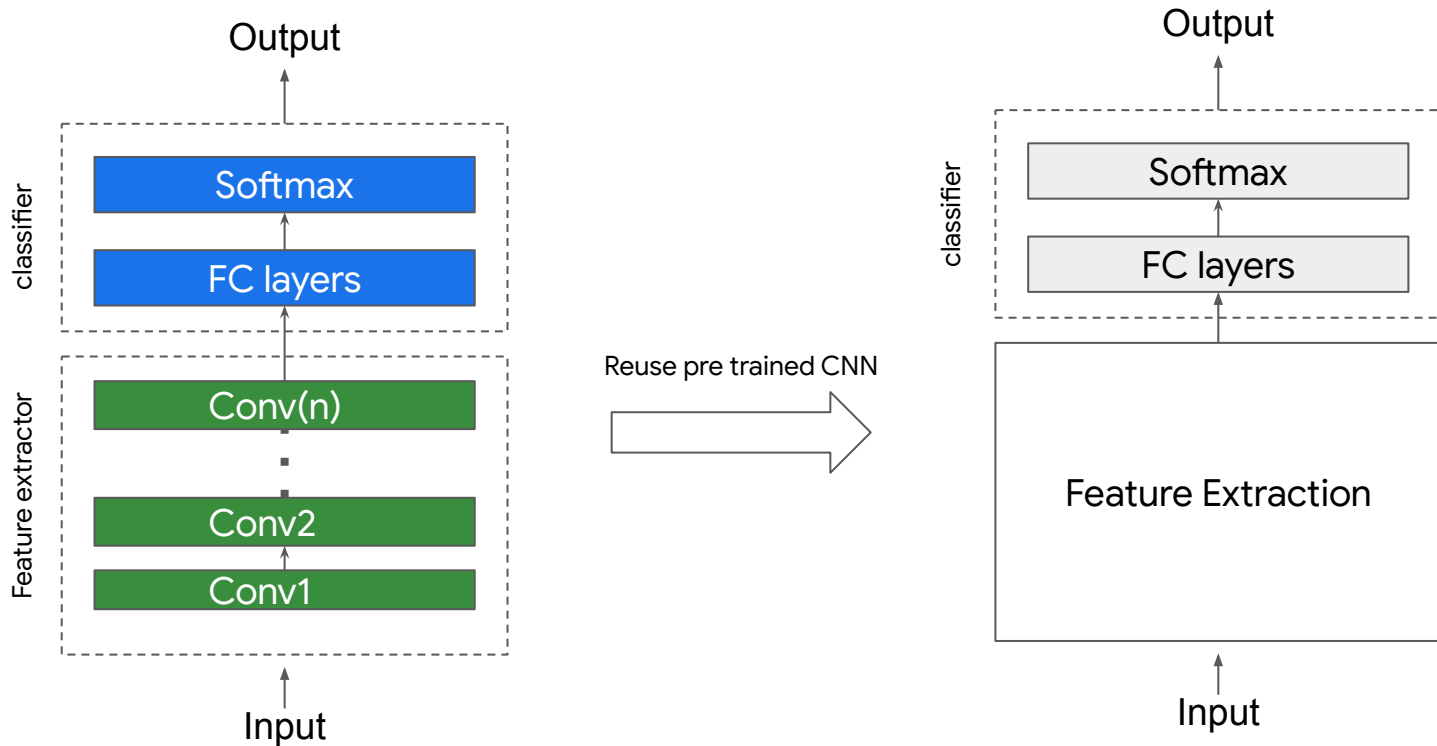
What is Transfer Learning?

- Pre trained model reused for solving downstream task.
- Reuse weights and layers.
- Example: pre-trained MobileNetV2 → cats vs dog classifier.

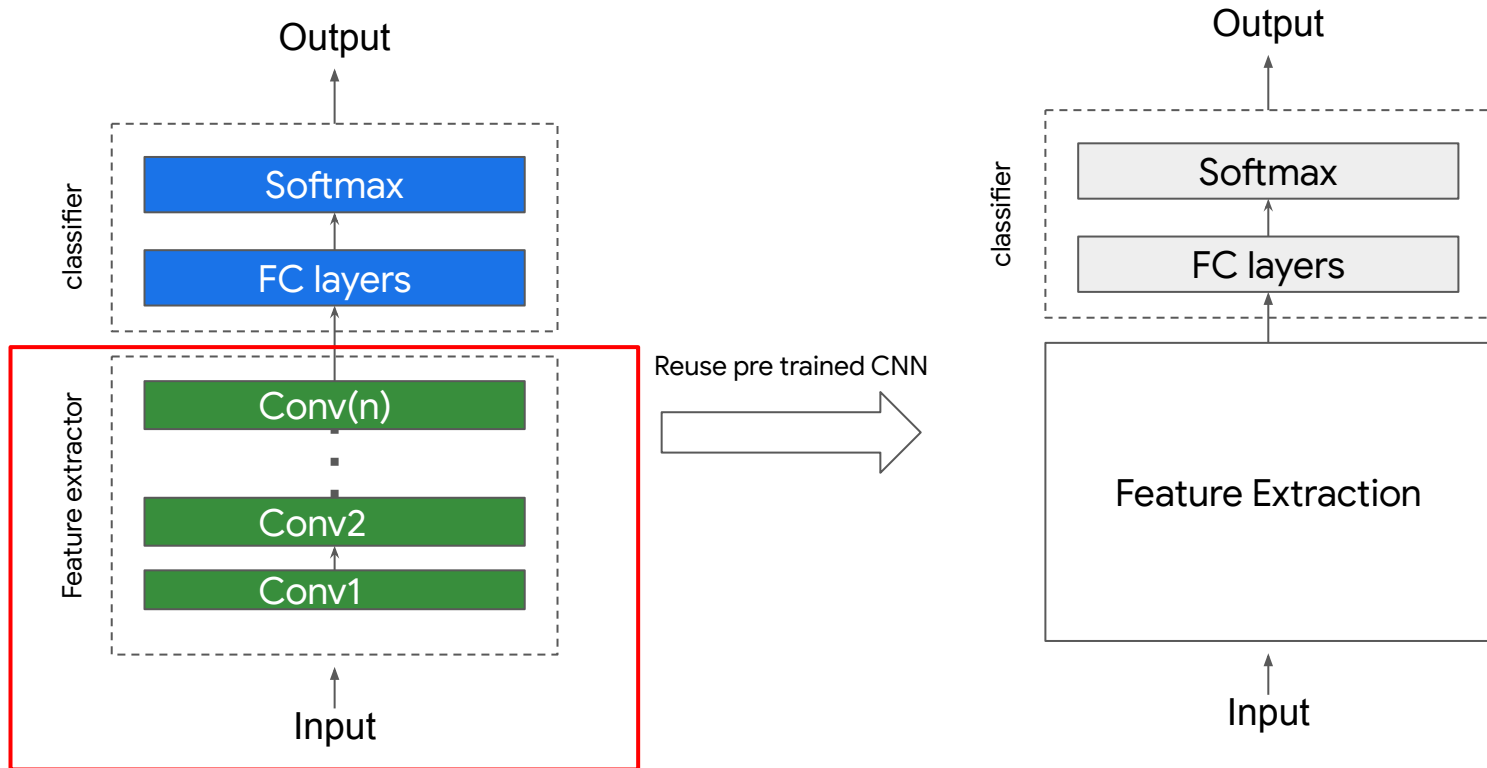
What is Transfer Learning?

- Pre trained model reused for solving downstream task.
- Reuse weights and layers.
- Example: pre-trained MobileNetV2 → cats vs dog classifier.

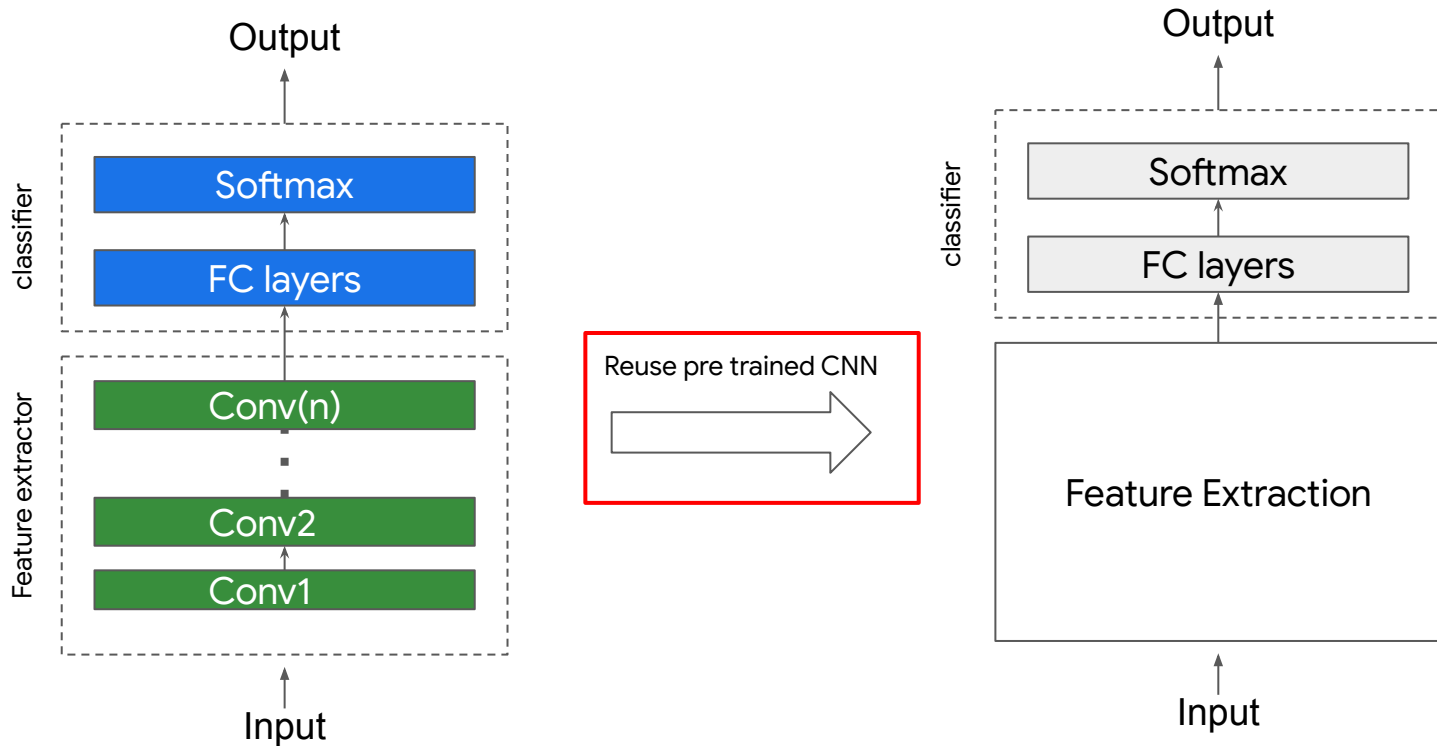
What is Transfer Learning?



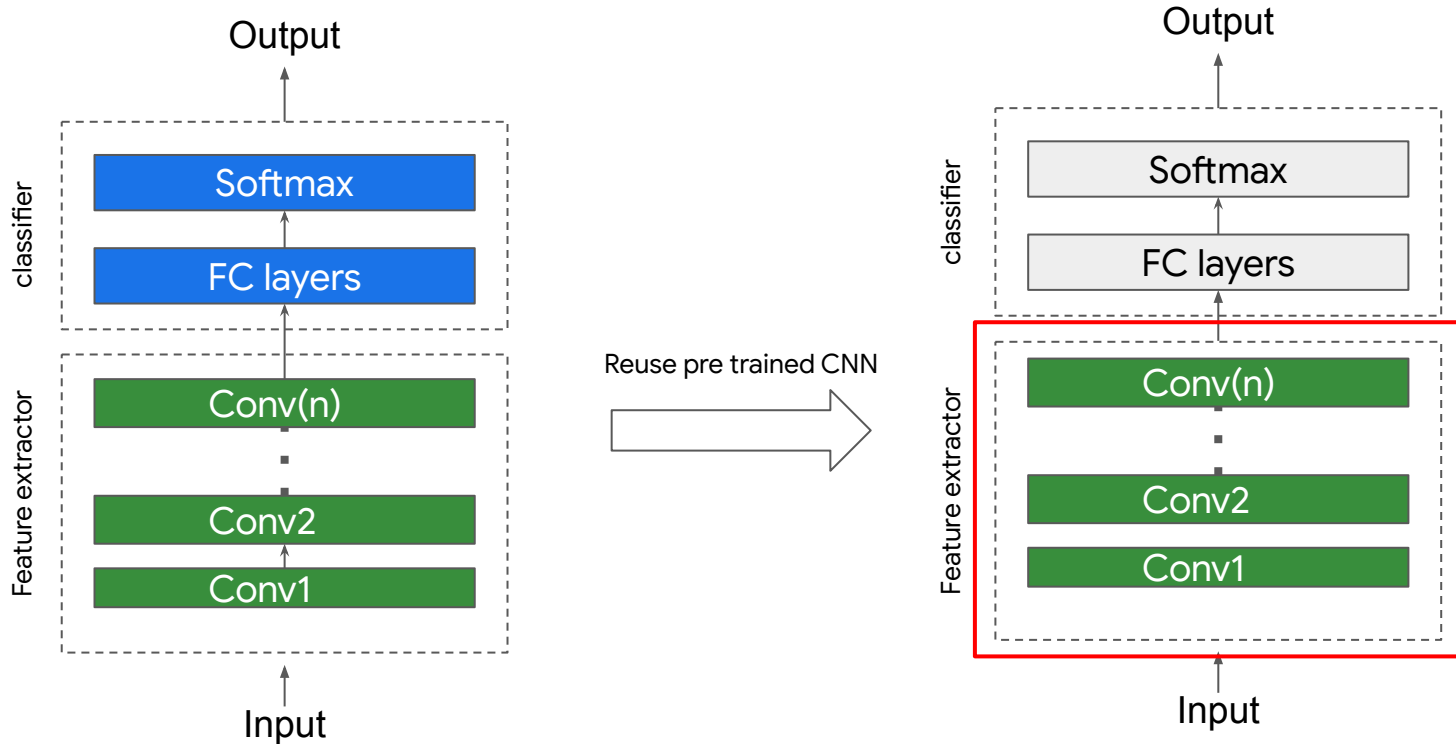
What is Transfer Learning?



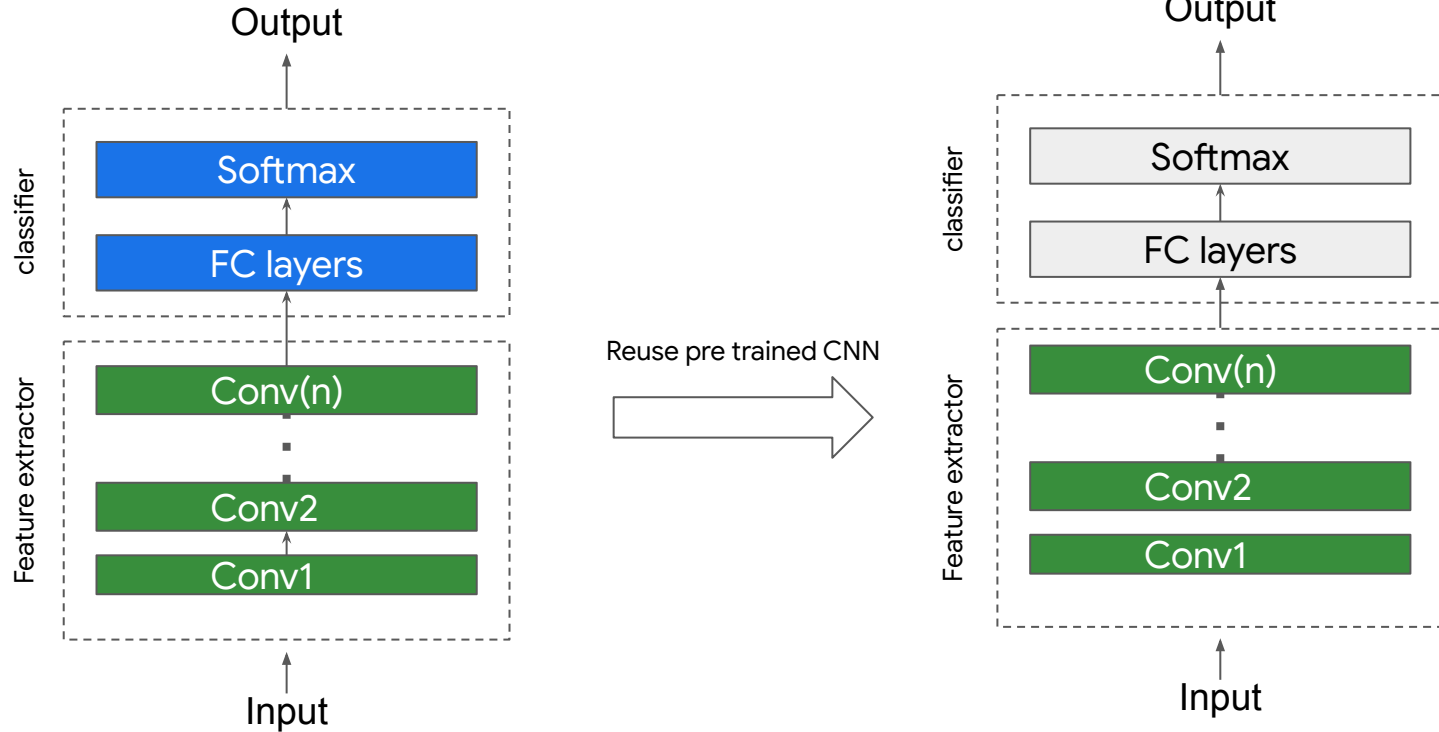
What is Transfer Learning?



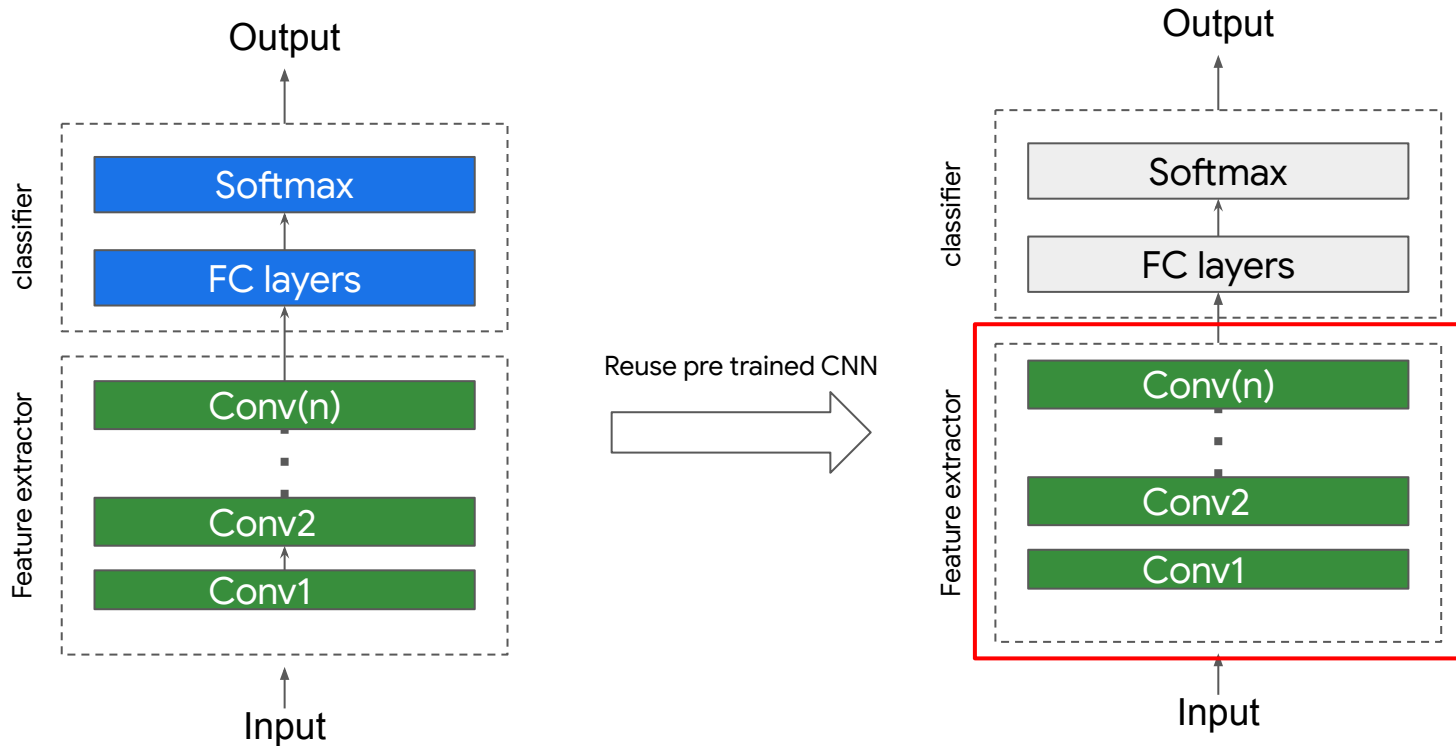
What is Transfer Learning?



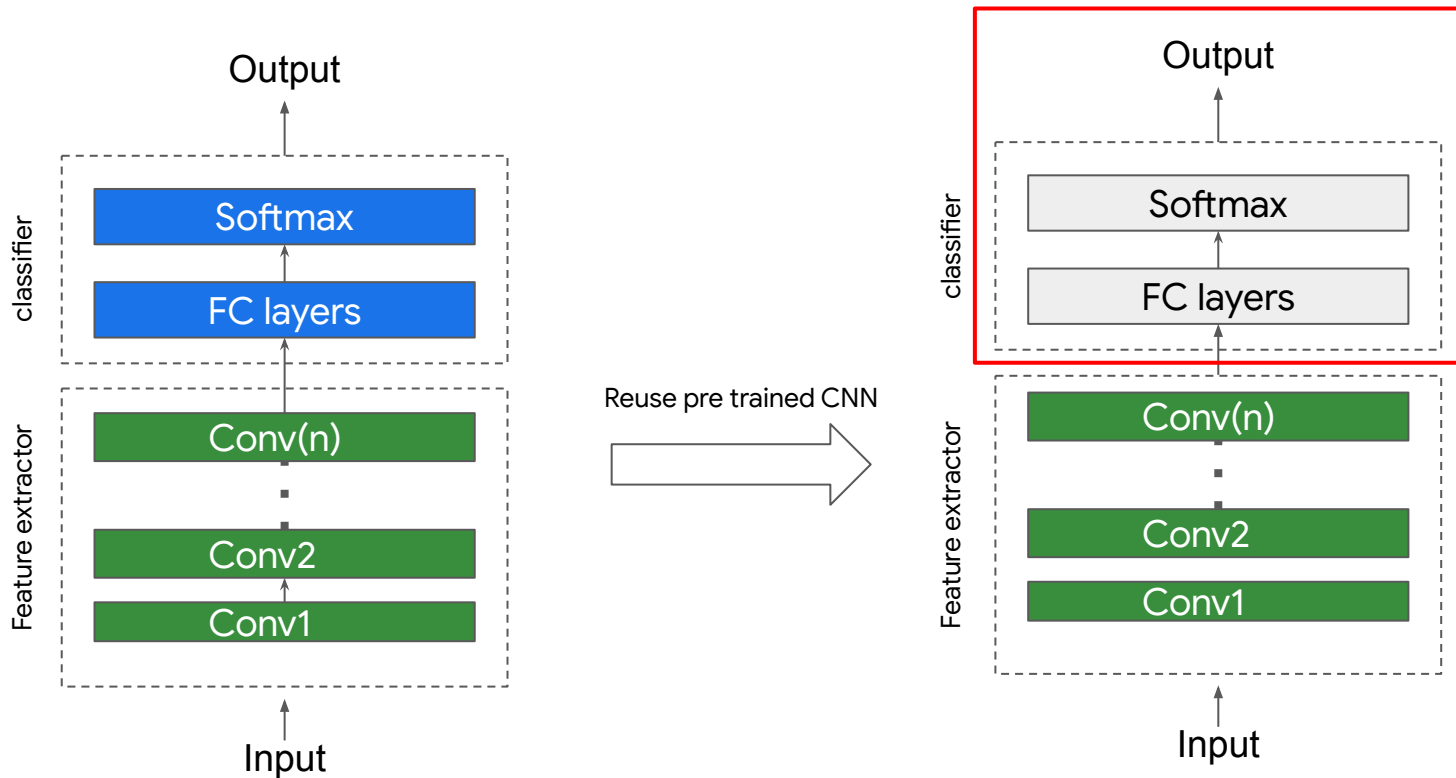
Freeze the Weights



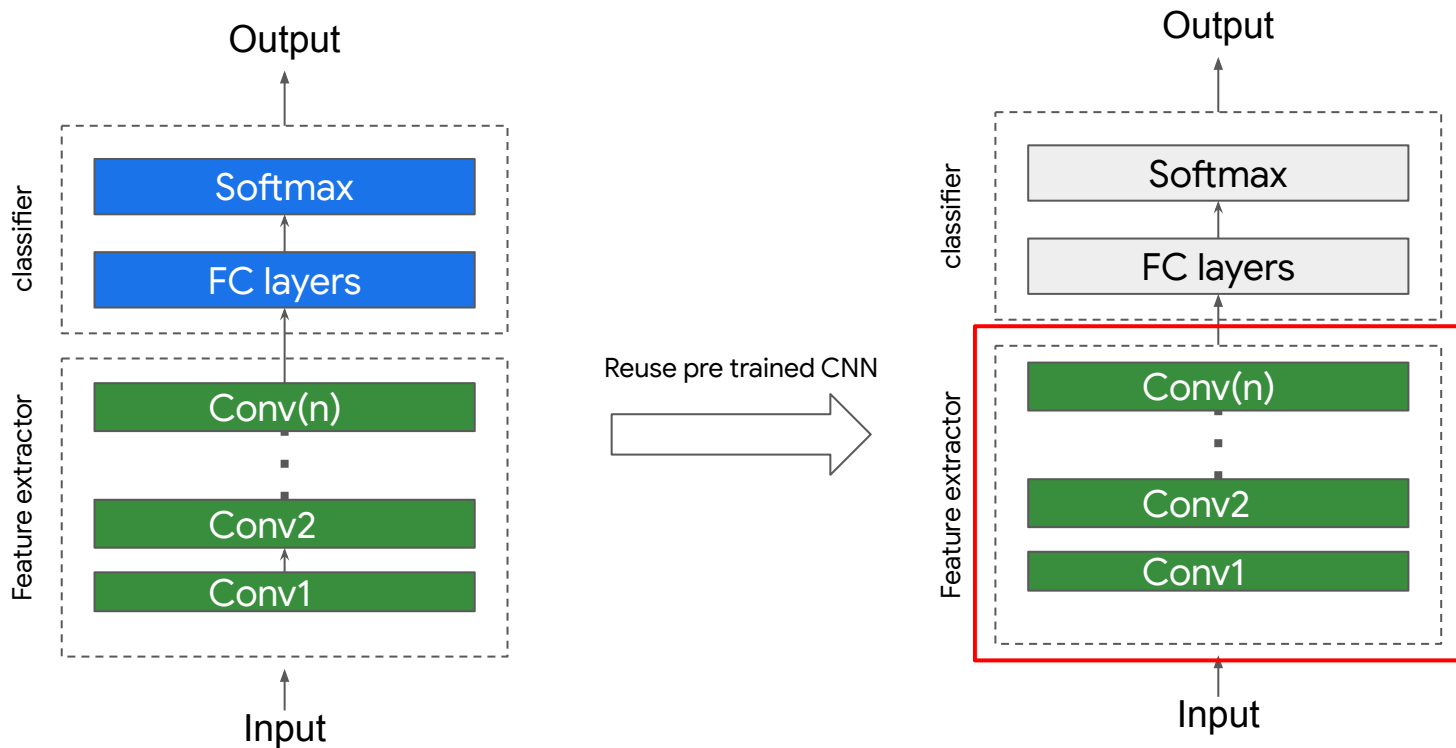
Freeze the Weights



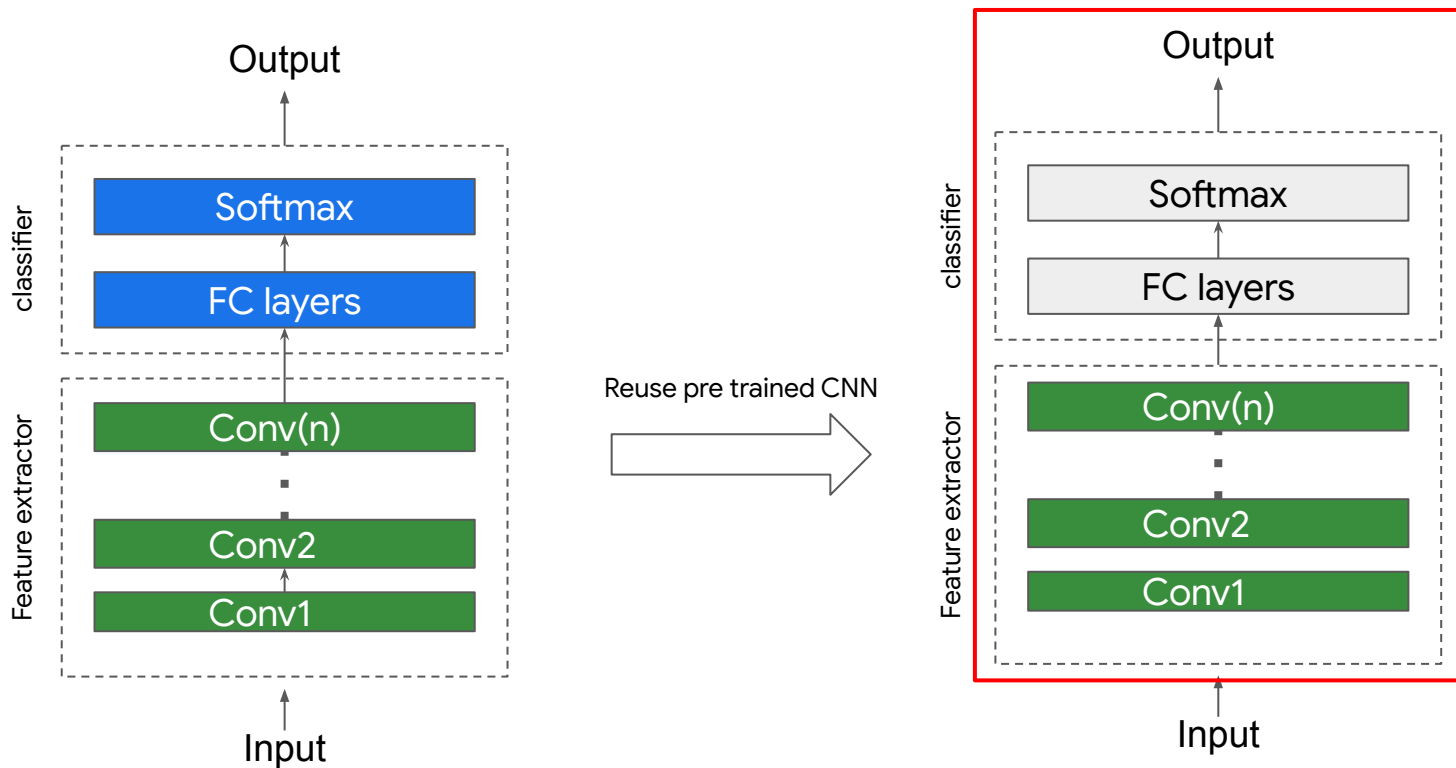
Freeze the Weights



Train with learned values as default



Train with learned values as default



CIFAR - 10 Dataset

10

airplane

automobile

bird

cat

deer

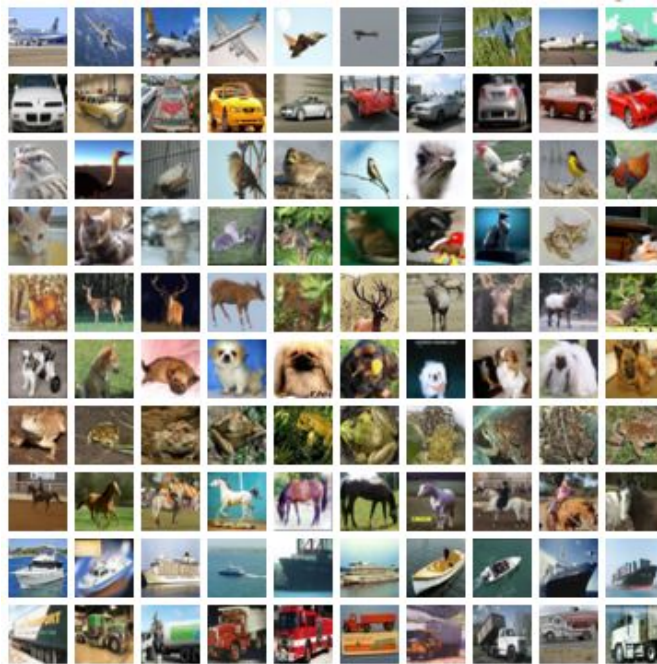
dog

frog

horse

ship

truck



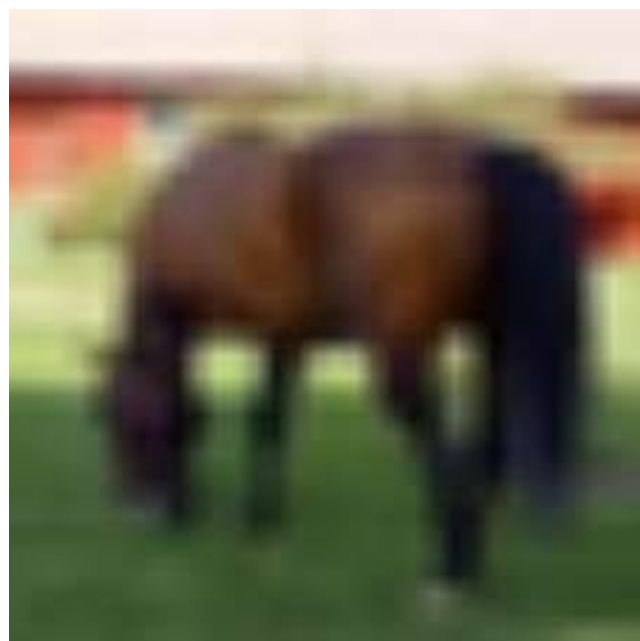
~6,000

...

~6,000

= 60,000

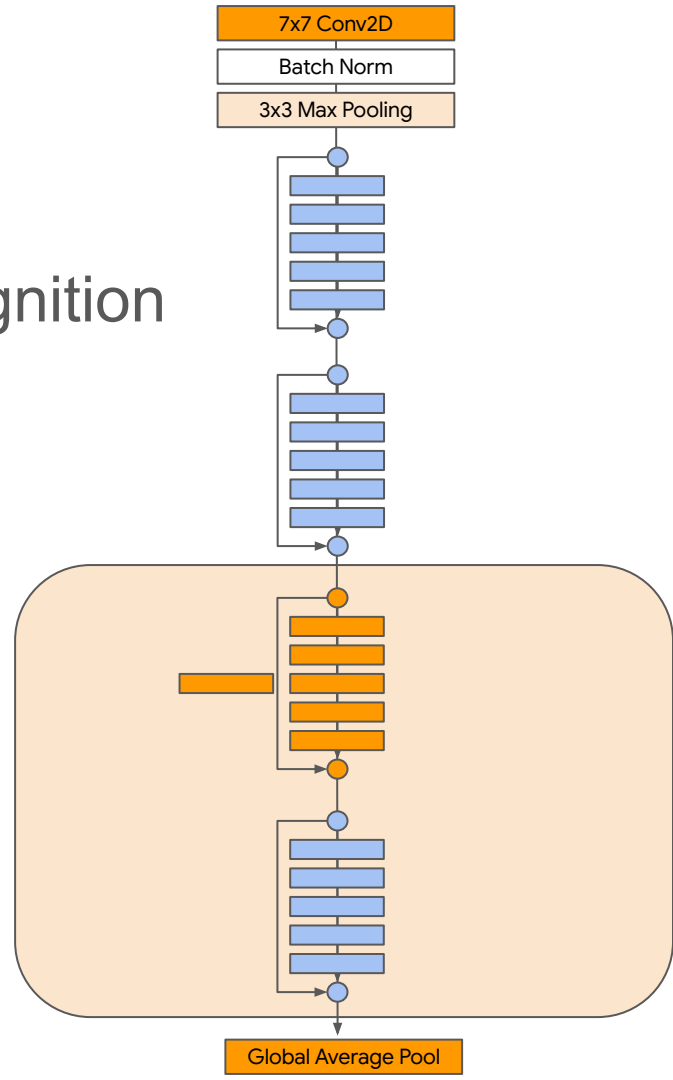
<https://www.cs.toronto.edu/~kriz/cifar.html>



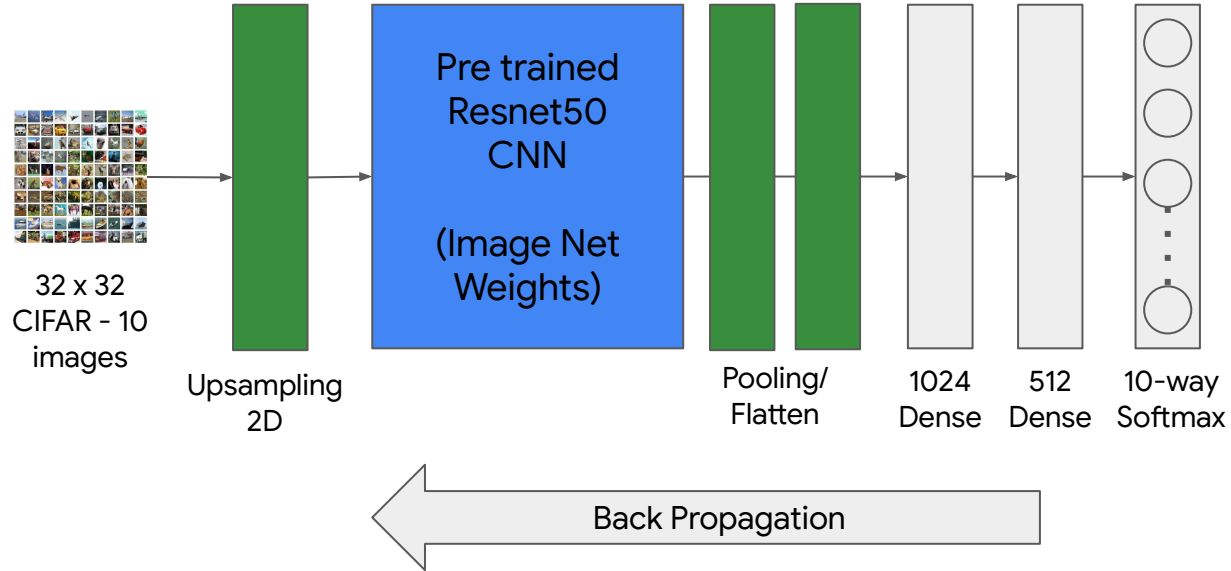
ResNet 50

Deep Residual Learning for Image Recognition

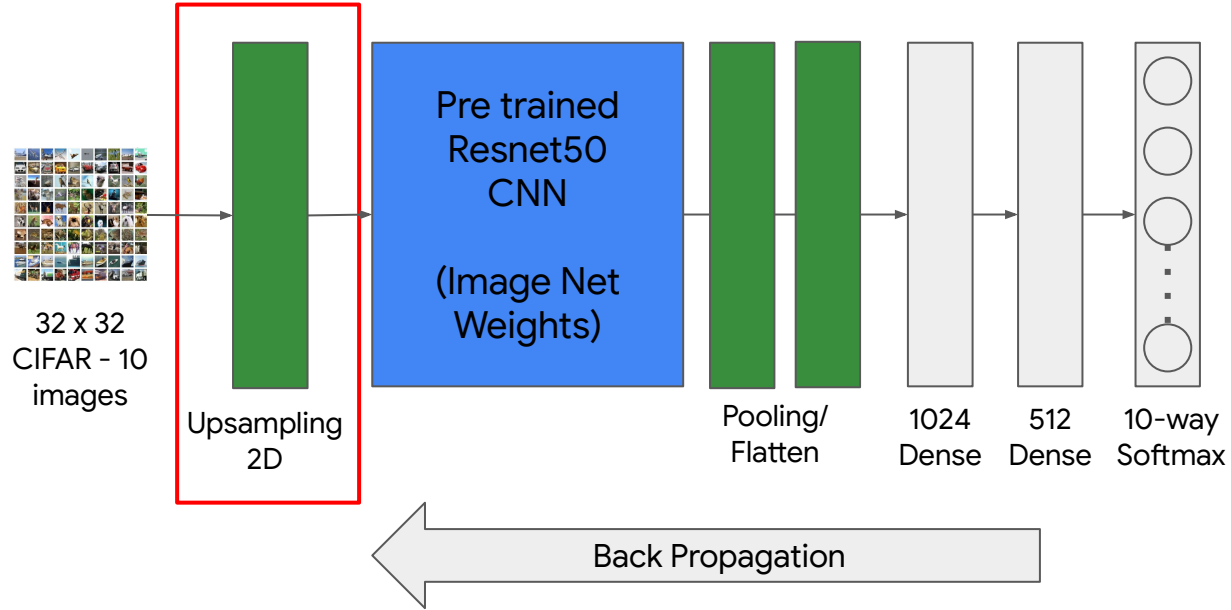
<https://arxiv.org/abs/1512.03385>



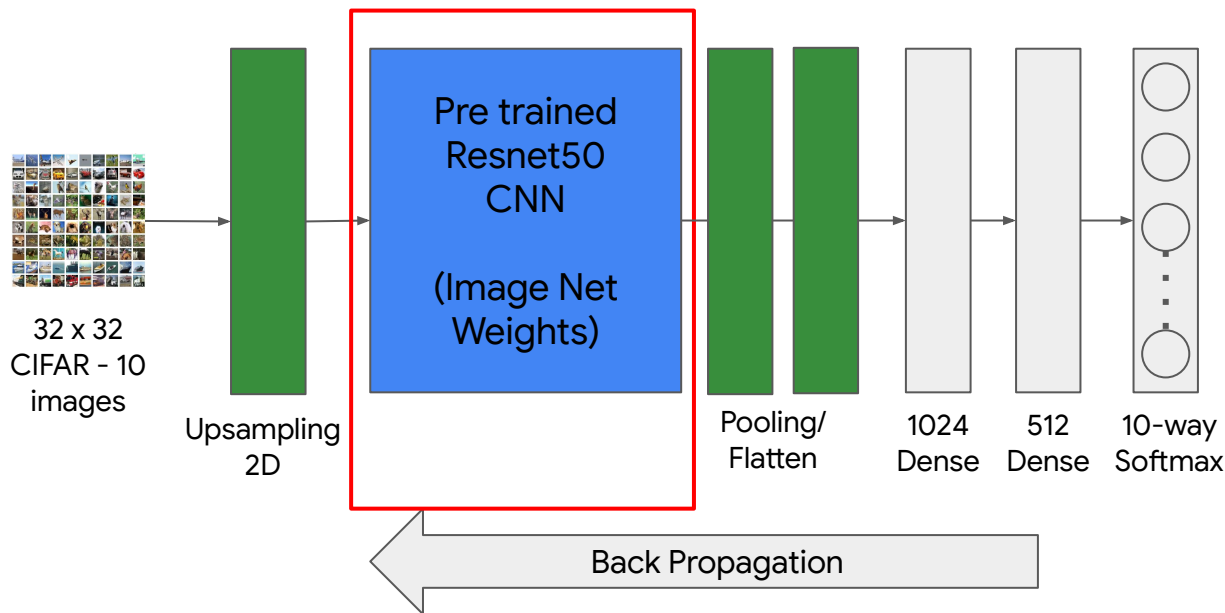
Network Architecture



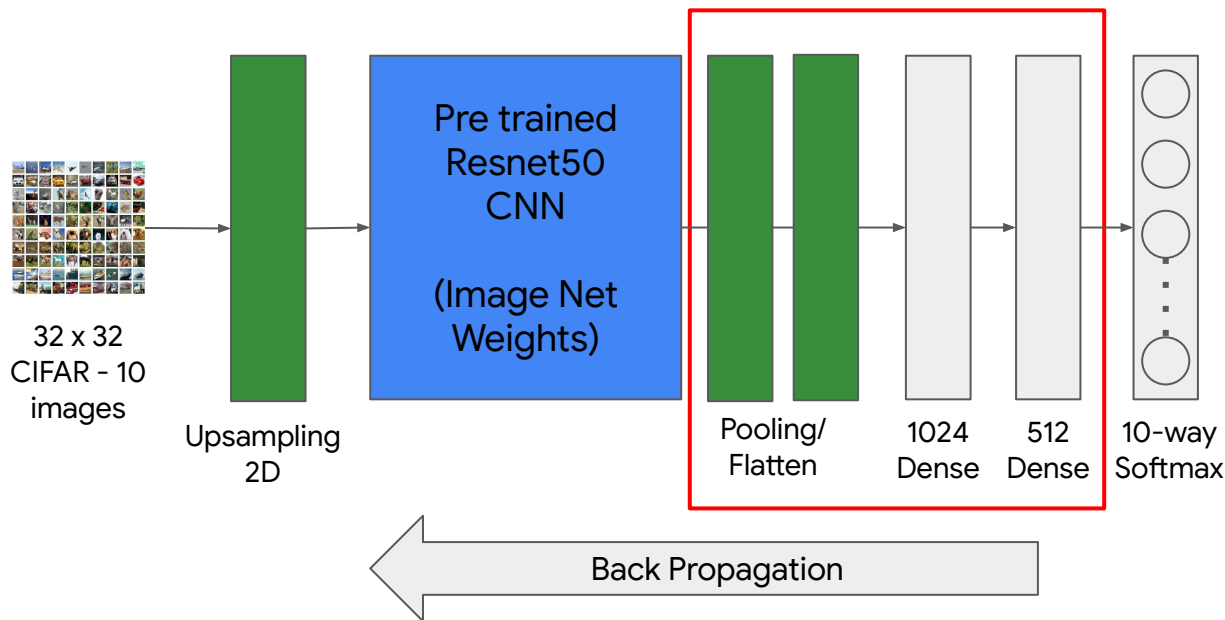
Network Architecture



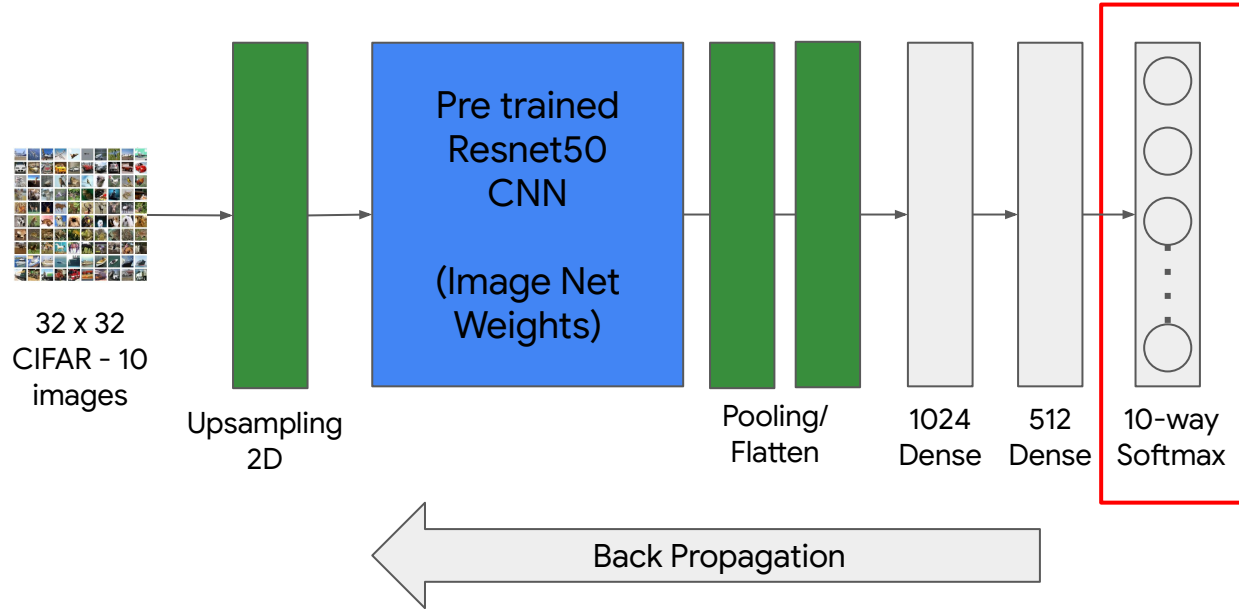
Network Architecture



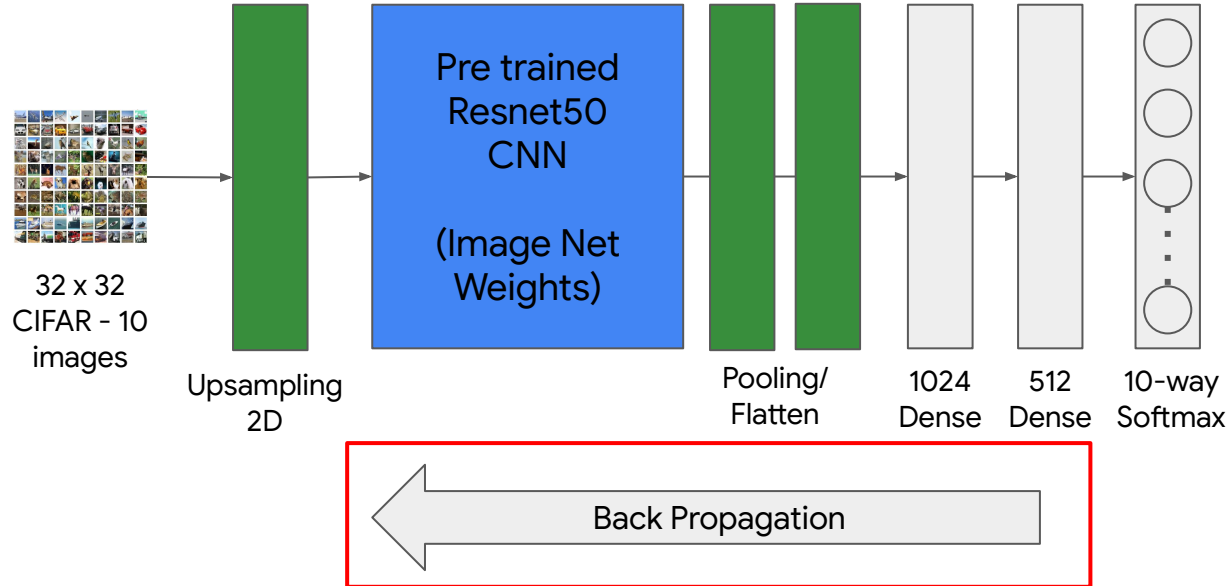
Network Architecture



Network Architecture



Network Architecture



Define Feature Extraction

Pre trained
Resnet50
CNN

(Image Net
Weights)

```
def feature_extractor(inputs):  
    feature_extractor_layer = tf.keras.applications.resnet.ResNet50(  
        input_shape=(224, 224, 3),  
        include_top=False,  
        weights='imagenet')(inputs)  
  
    return feature_extractor_layer
```

Define Feature Extraction

Pre trained
Resnet50
CNN

(Image Net
Weights)

```
def feature_extractor(inputs):  
    feature_extractor_layer = tf.keras.applications.resnet.ResNet50(  
        input_shape=(224, 224, 3),  
        include_top=False,  
        weights='imagenet')(inputs)  
  
    return feature_extractor_layer
```

Define Feature Extraction

Pre trained
Resnet50
CNN

(Image Net
Weights)

```
def feature_extractor(inputs):  
    feature_extractor_layer = tf.keras.applications.resnet.ResNet50(  
        input_shape=(224, 224, 3),  
        include_top=False,  
        weights='imagenet')(inputs)  
  
    return feature_extractor_layer
```

Define Feature Extraction

Pre trained
Resnet50
CNN

(Image Net
Weights)

```
def feature_extractor(inputs):  
    feature_extractor_layer = tf.keras.applications.resnet.ResNet50(  
        input_shape=(224, 224, 3),  
        include_top=False,  
        weights='imagenet')(inputs)  
  
    return feature_extractor_layer
```

Define Feature Extraction

Pre trained
Resnet50
CNN

(Image Net
Weights)

```
def feature_extractor(inputs):  
    feature_extractor_layer = tf.keras.applications.resnet.ResNet50(  
        input_shape=(224, 224, 3),  
        include_top=False,  
        weights='imagenet')(inputs)  
  
    return feature_extractor_layer
```

Define Feature Extraction

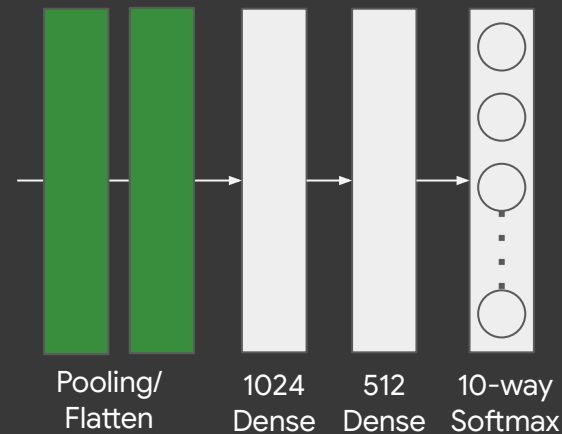
Pre trained
Resnet50
CNN

(Image Net
Weights)

```
def feature_extractor(inputs):  
    feature_extractor_layer = tf.keras.applications.resnet.ResNet50(  
        input_shape=(224, 224, 3),  
        include_top=False,  
        weights='imagenet')(inputs)  
  
    return feature_extractor_layer
```

Define Classifier

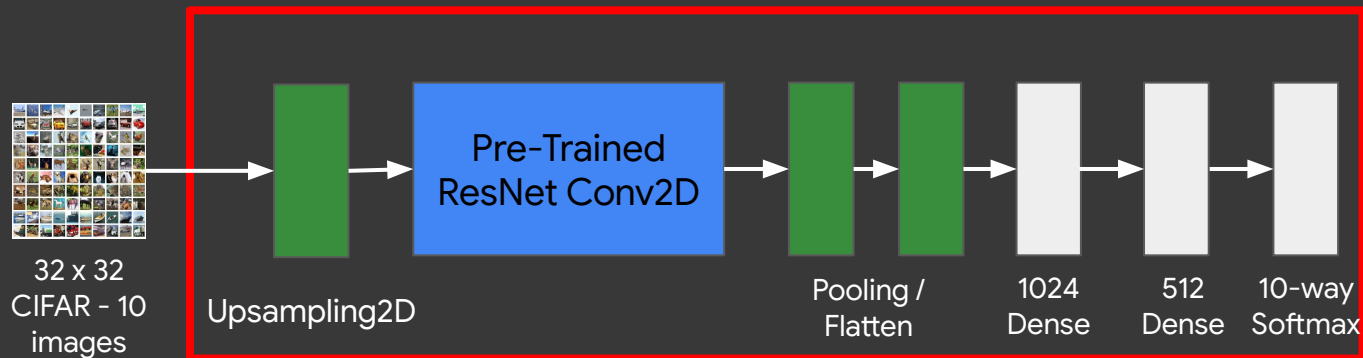
```
def classifier(inputs):  
    x = tf.keras.layers.GlobalAveragePooling2D()(inputs)  
    x = tf.keras.layers.Flatten()(x)  
    x = tf.keras.layers.Dense(1024, activation="relu")(x)  
    x = tf.keras.layers.Dense(512, activation="relu")(x)  
    x = tf.keras.layers.Dense(10, activation="softmax", name="classification")(x)  
    return x
```



Finalize Model

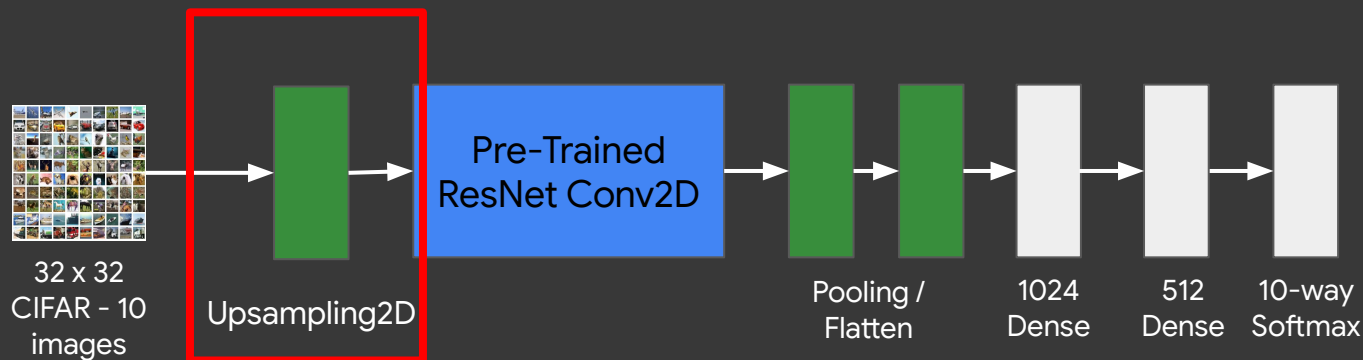
```
def final_model(inputs):
```

```
...
```



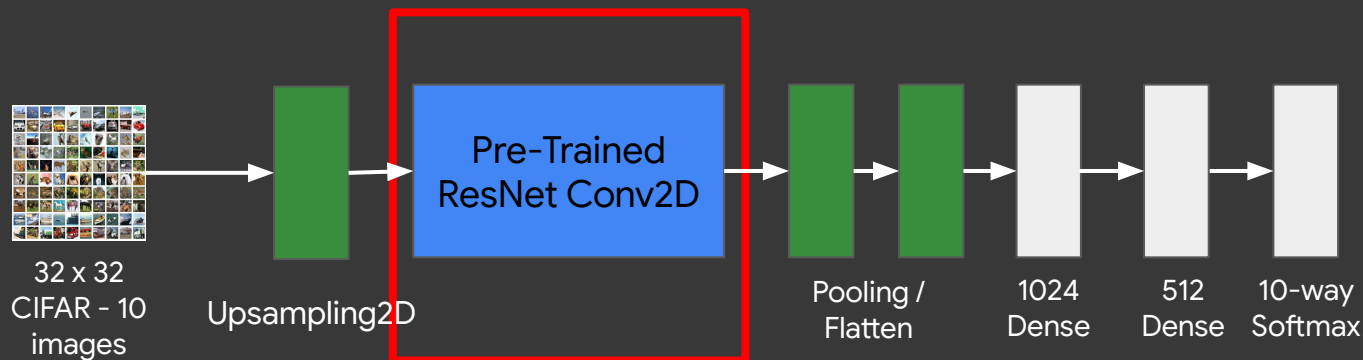
Define Inputs and Outputs

```
def final_model(inputs):  
    resize = tf.keras.layers.UpSampling2D(size=(7,7))(inputs)
```



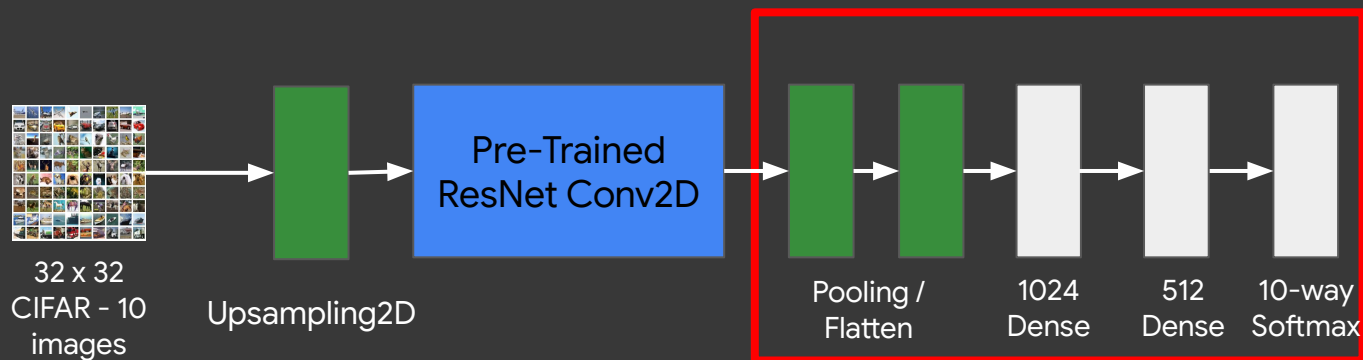
Define Inputs and Outputs

```
def final_model(inputs):  
    resize = tf.keras.layers.UpSampling2D(size=(7,7))(inputs)  
  
    resnet_feature_extractor = feature_extractor(resize)
```



Define Inputs and Outputs

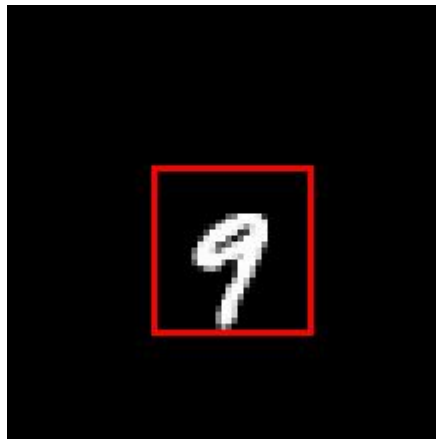
```
def final_model(inputs):  
    resize = tf.keras.layers.UpSampling2D(size=(7,7))(inputs)  
  
    resnet_feature_extractor = feature_extractor(resize)  
    classification_output = classifier(resnet_feature_extractor)  
  
    return classification_output
```



Augmenting MNIST for Object Localization

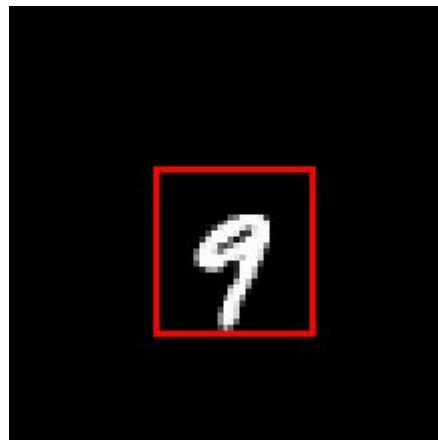


A sample from MNIST



Sample pasted on a 75 x 75
black canvas

Augmenting MNIST for Object Localization

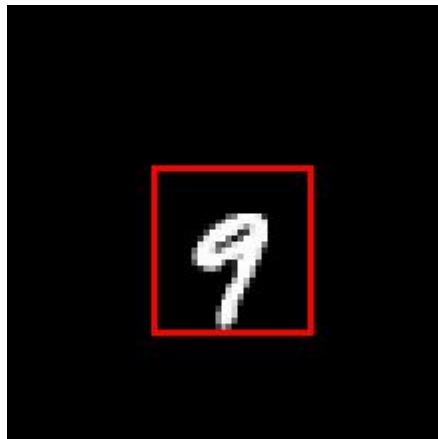


Sample pasted on a 75 x 75
black canvas

Augmenting MNIST for Object Localization

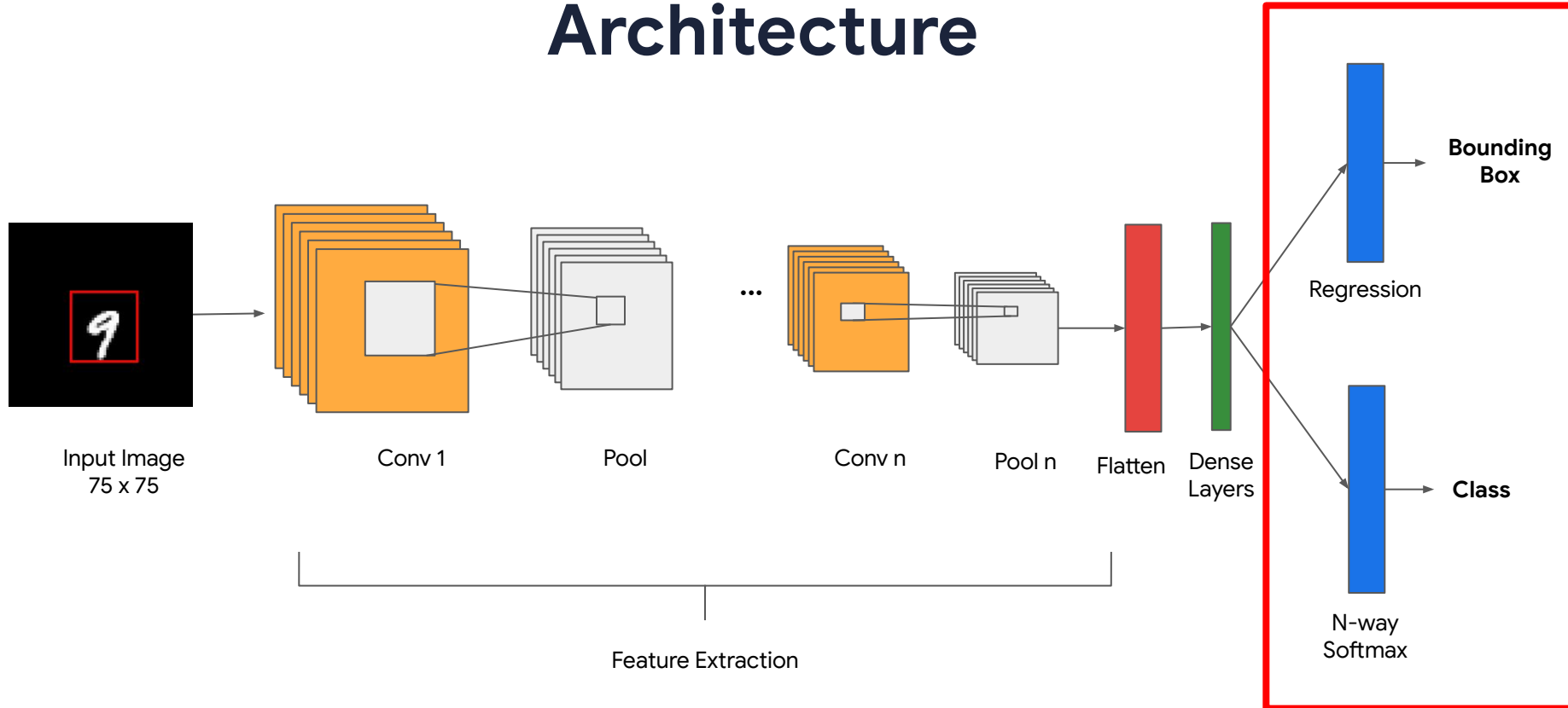


A sample from MNIST

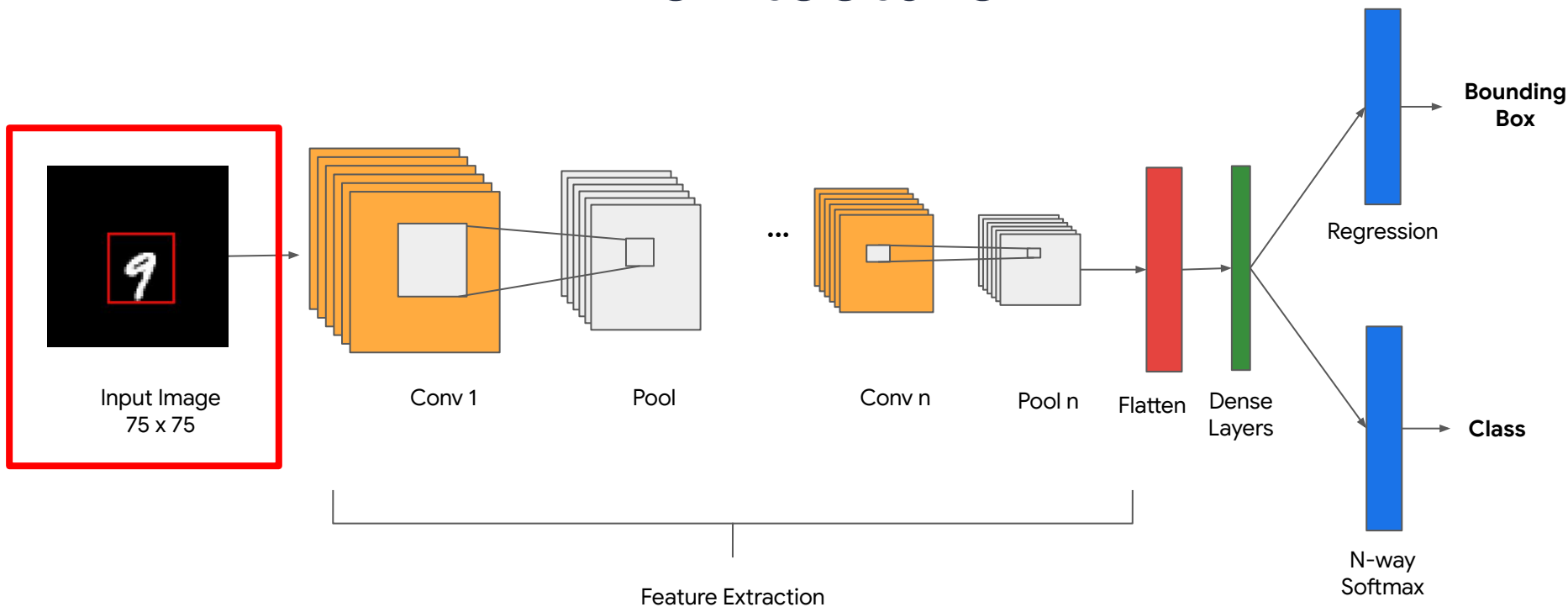


Sample pasted on a 75 x 75
black canvas

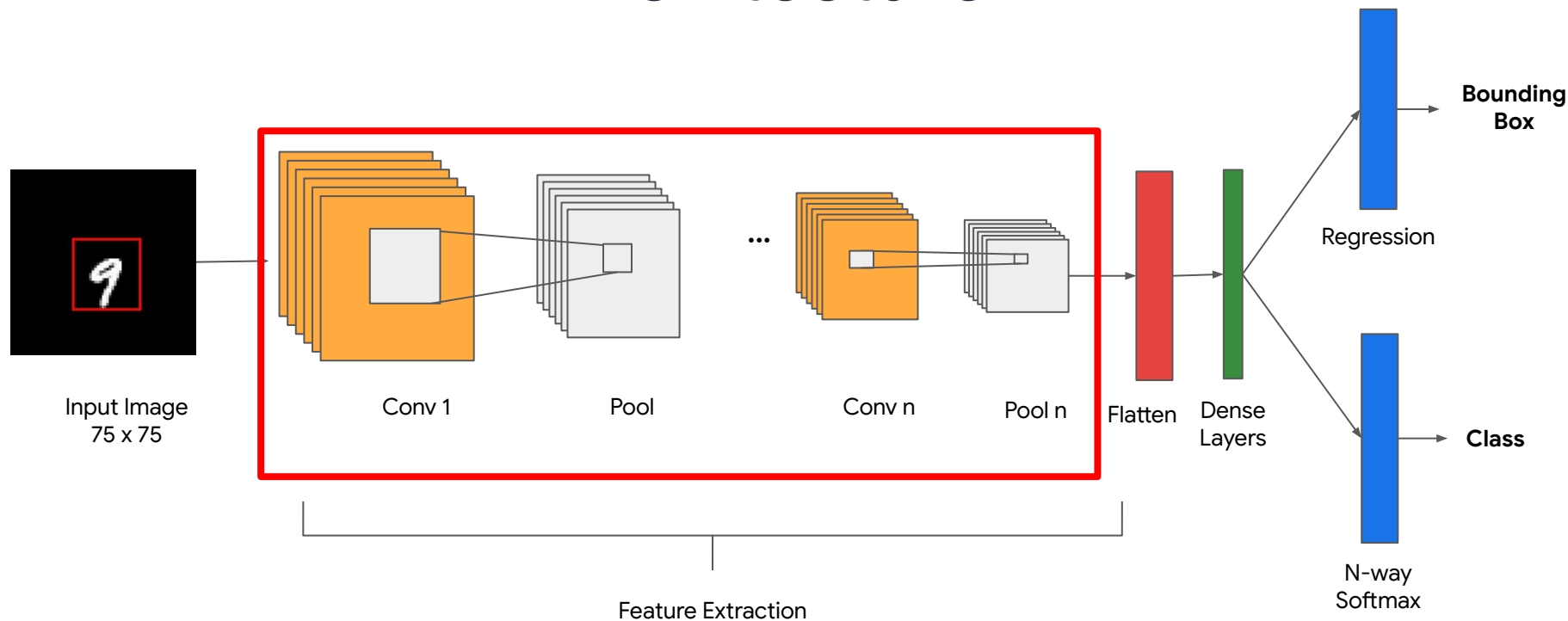
Convolutional Neural Networks Architecture



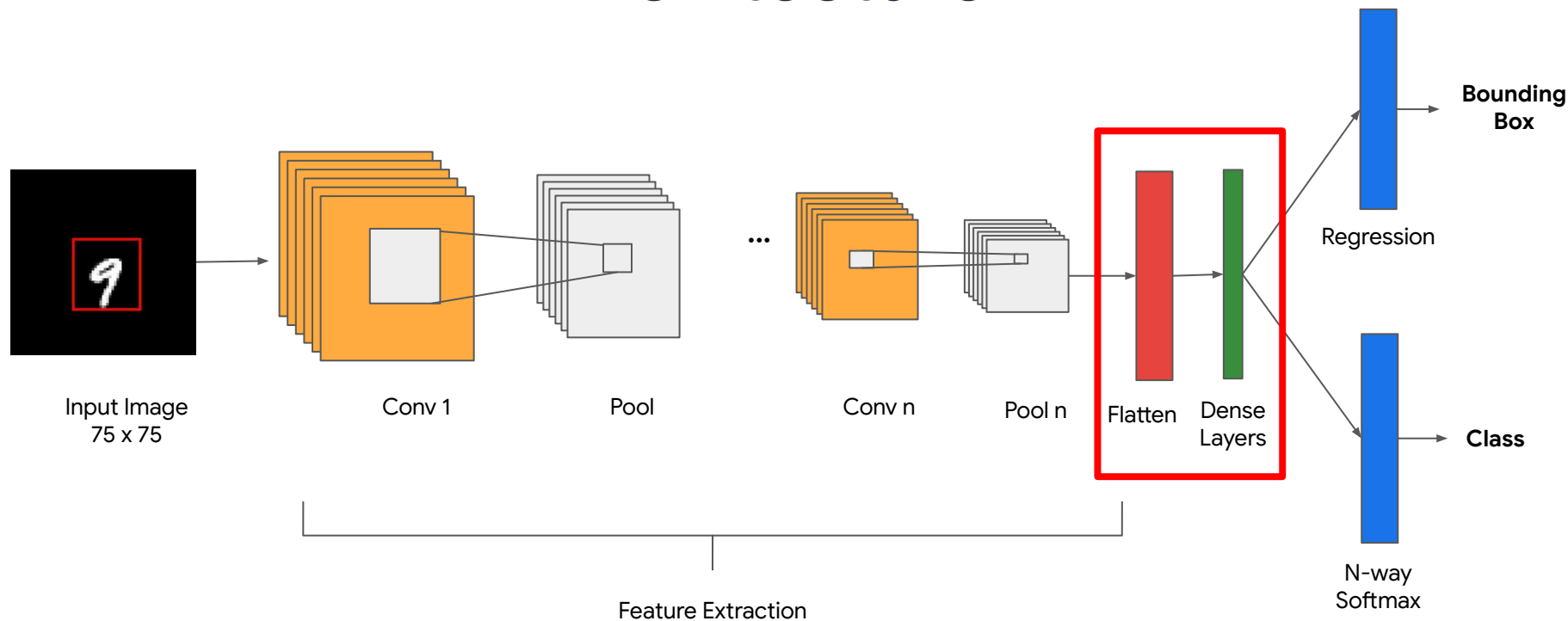
Convolutional Neural Networks Architecture



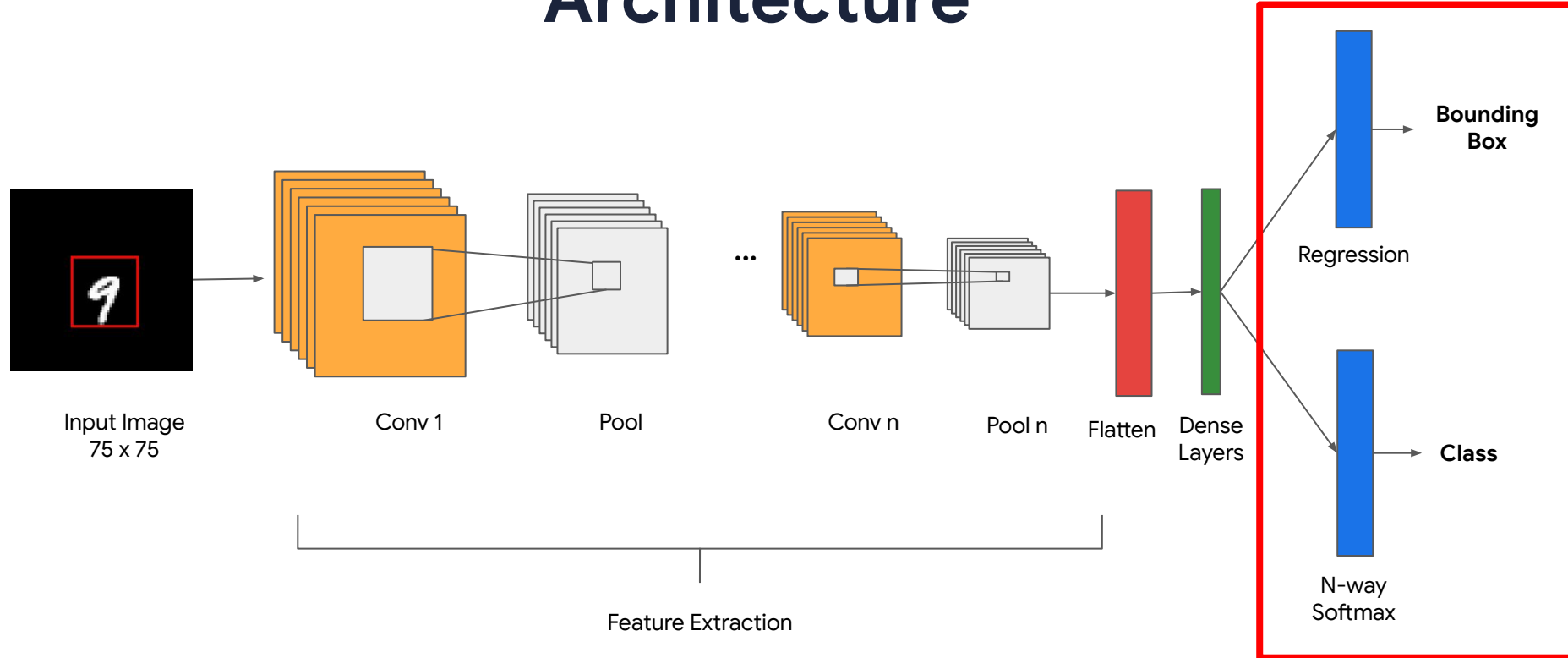
Convolutional Neural Networks Architecture



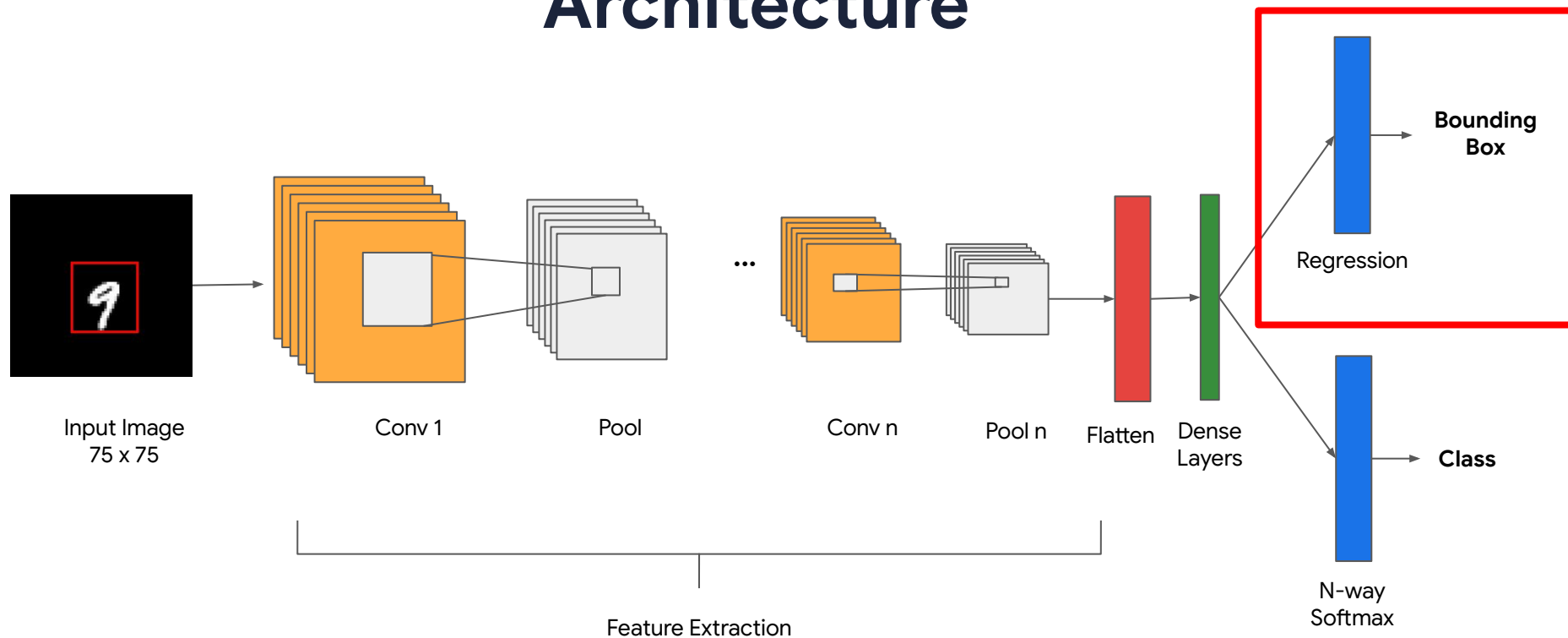
Convolutional Neural Networks Architecture



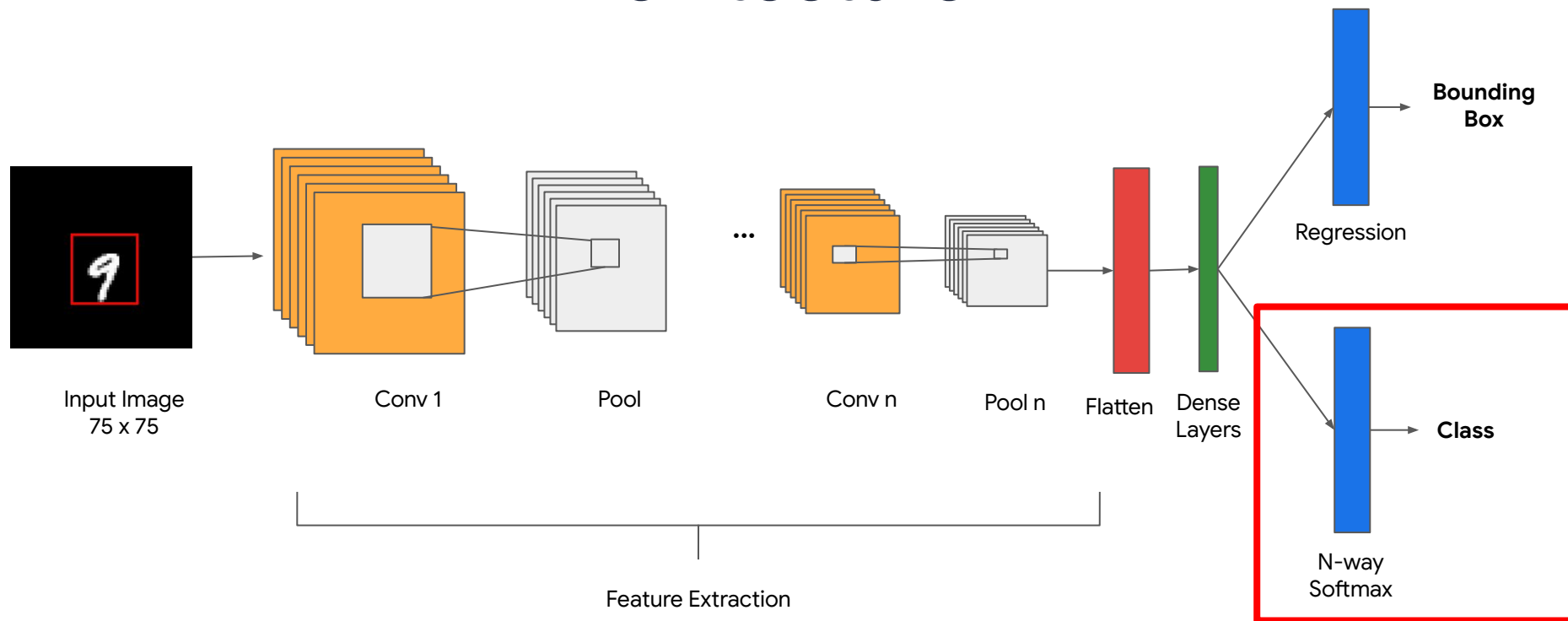
Convolutional Neural Networks Architecture

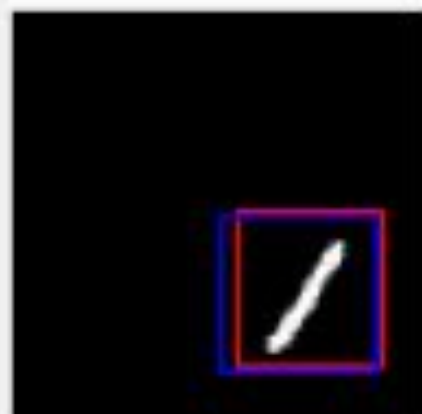


Convolutional Neural Networks Architecture



Convolutional Neural Networks Architecture





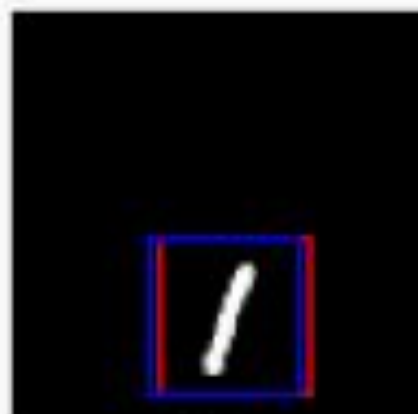
1

iou: 0.90866166



4

iou: 1.1792105



1

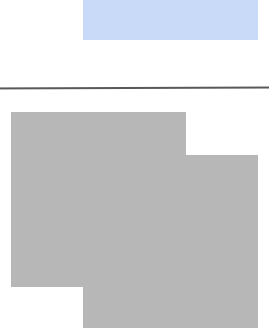
iou: 1.2664019



4

iou: 1.0059167

Intersection Over Union

$$\text{IOU} = \frac{\text{Intersection}}{\text{Union}}$$


The diagram illustrates the concept of Intersection Over Union (IOU) using two overlapping rectangles. The top rectangle is red, and the bottom rectangle is blue. The intersection of the two rectangles is highlighted in purple. The union of the two rectangles is highlighted in gray. The equation shows that IOU is the ratio of the intersection area to the union area.

Intersection Over Union

$$\text{IOU} = \frac{\text{Intersection}}{\text{Union}}$$

