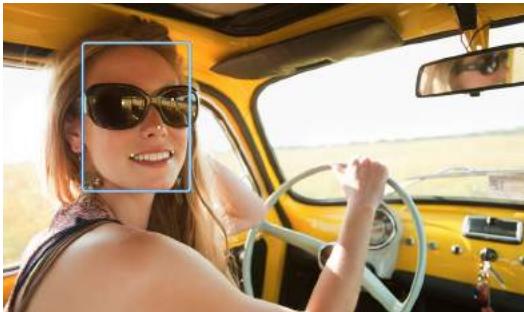




# Object Detection examples



Facial Detection / Recognition  
[Amazon Rekognition](#)

<https://docs.aws.amazon.com/rekognition/latest/dg/faces.html>



Counting vehicles  
[PowerAI](#)

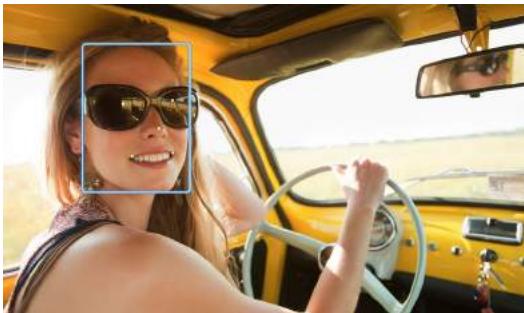
<https://github.com/IBM/powerai-counting-cars>



Analyzing aerial imagery  
[DIGITS](#)

<https://devblogs.nvidia.com/detectnet-deep-neural-network-object-detection-digits/>

# Object Detection examples



Facial Detection / Recognition  
[Amazon Rekognition](#)

<https://docs.aws.amazon.com/rekognition/latest/dg/faces.html>



Counting vehicles  
[PowerAI](#)

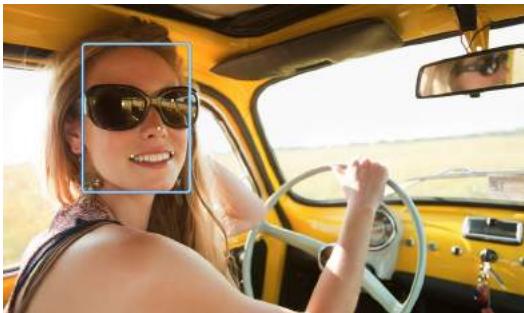
<https://github.com/IBM/powerai-counting-cars>



Analyzing aerial imagery  
[DIGITS](#)

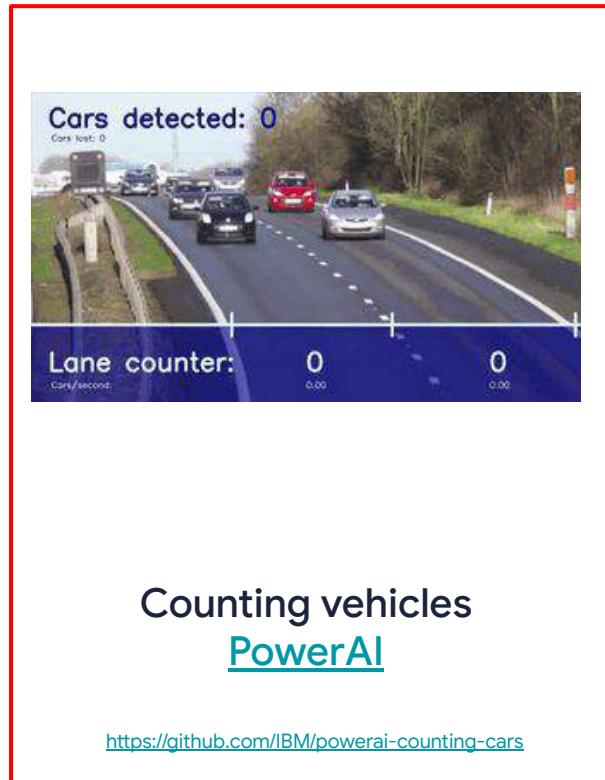
<https://devblogs.nvidia.com/detectnet-deep-neural-network-object-detection-digits/>

# Object Detection examples



Facial Detection / Recognition  
[Amazon Rekognition](#)

<https://docs.aws.amazon.com/rekognition/latest/dg/faces.html>



Counting vehicles  
[PowerAI](#)

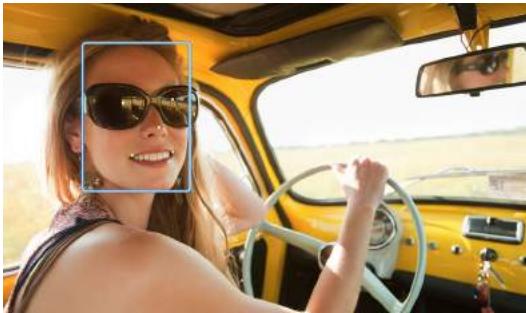
<https://github.com/IBM/powerai-counting-cars>



Analyzing aerial imagery  
[DIGITS](#)

<https://devblogs.nvidia.com/detectnet-deep-neural-network-object-detection-digits/>

# Object Detection examples



Facial Detection / Recognition  
[Amazon Rekognition](#)

<https://docs.aws.amazon.com/rekognition/latest/dg/faces.html>



Counting vehicles  
[PowerAI](#)

<https://github.com/IBM/powerai-counting-cars>

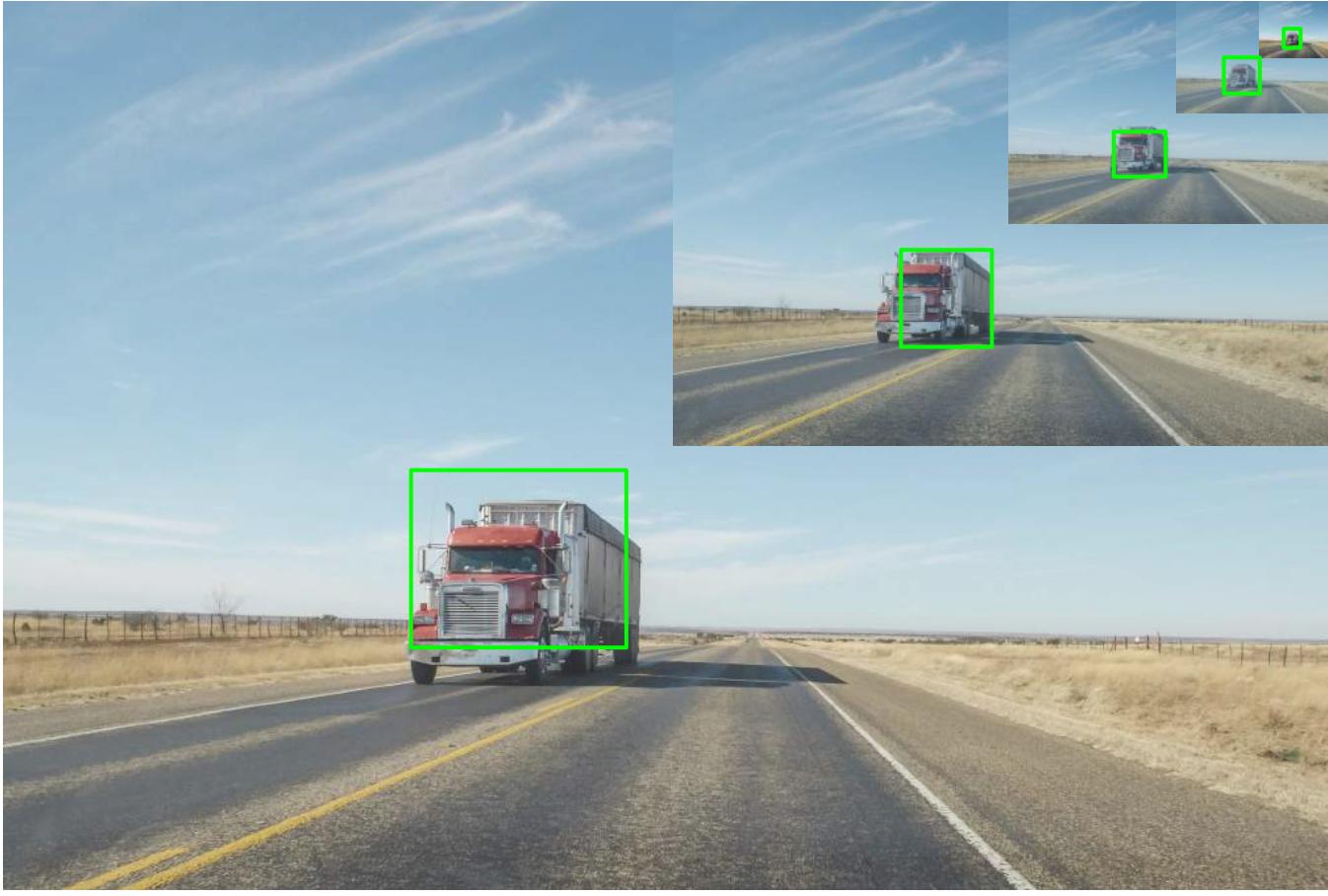


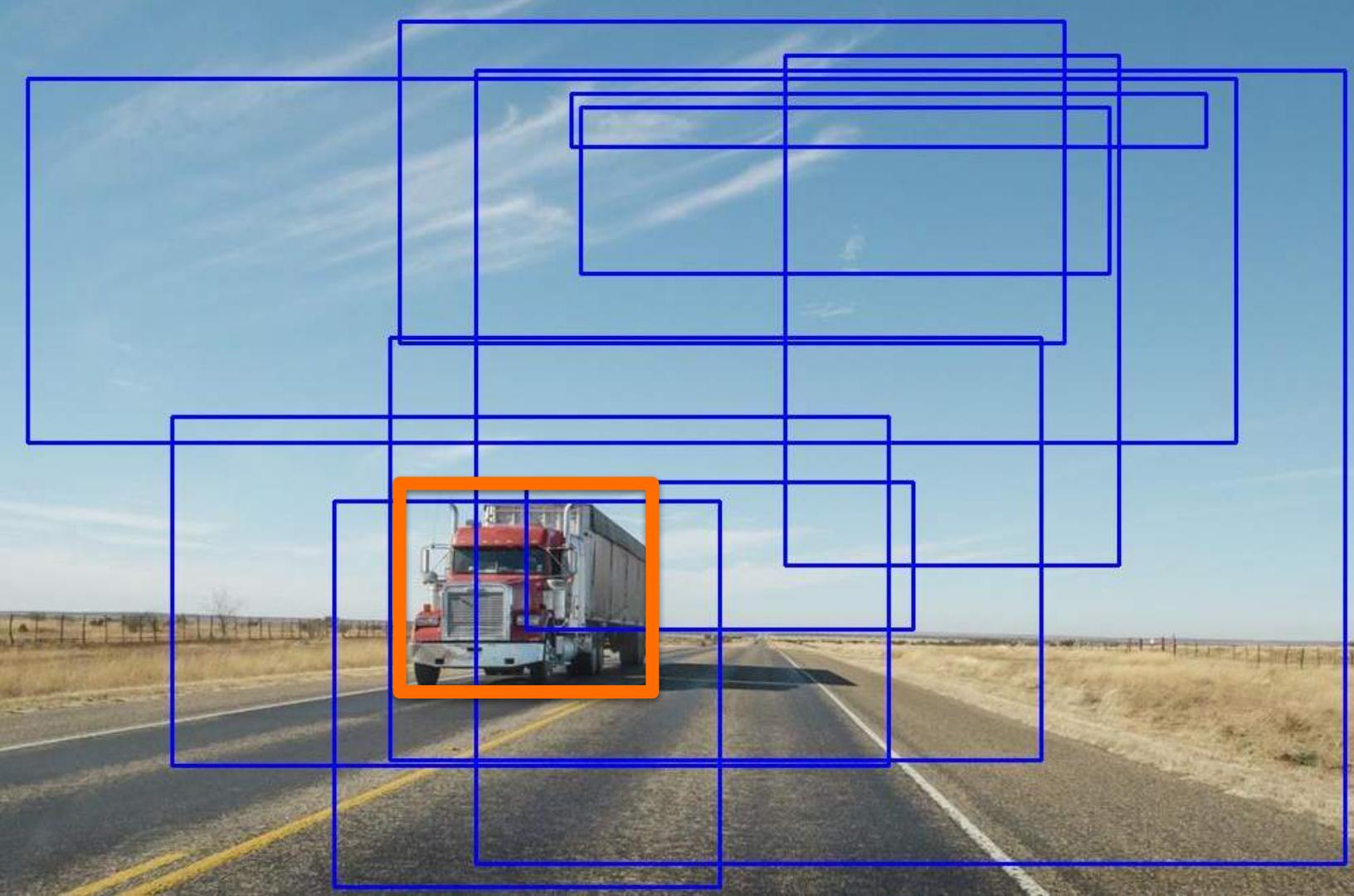
Analyzing aerial imagery  
[DIGITS](#)

<https://devblogs.nvidia.com/detectnet-deep-neural-network-object-detection-digits/>

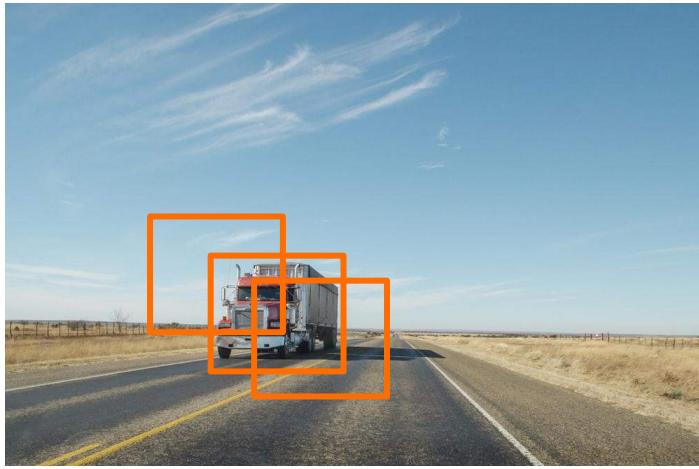






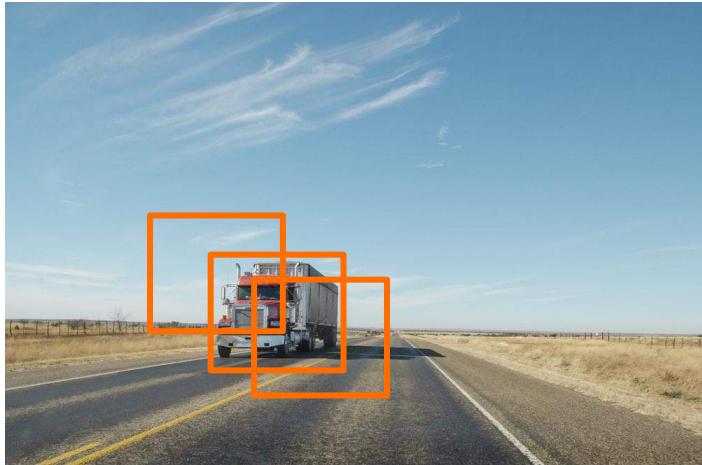


# Sliding Window

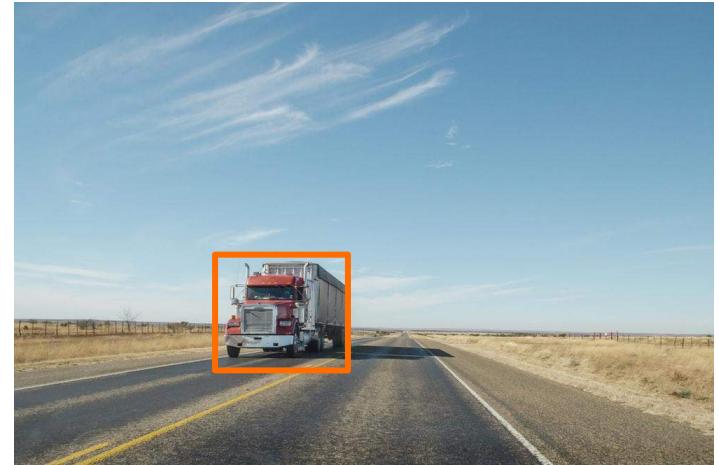


Before NMS

# Non-maximum suppression (NMS)



Before NMS



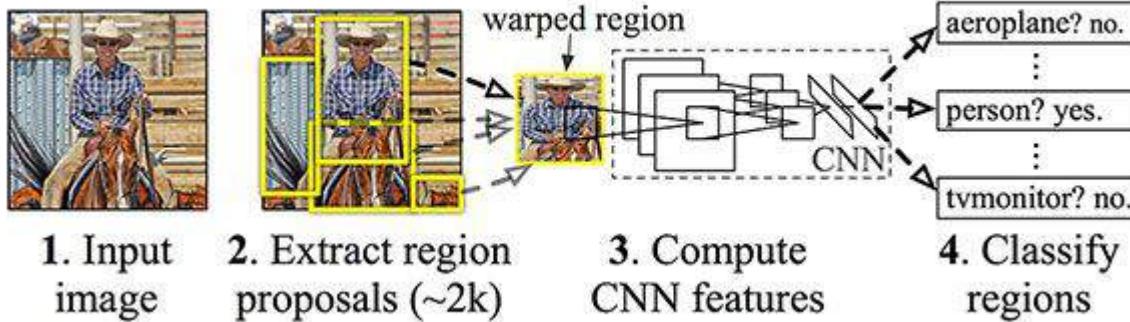
After NMS

# Two Stages to Object Detection:

1. Region proposal
2. Object detection and classification

# R-CNN

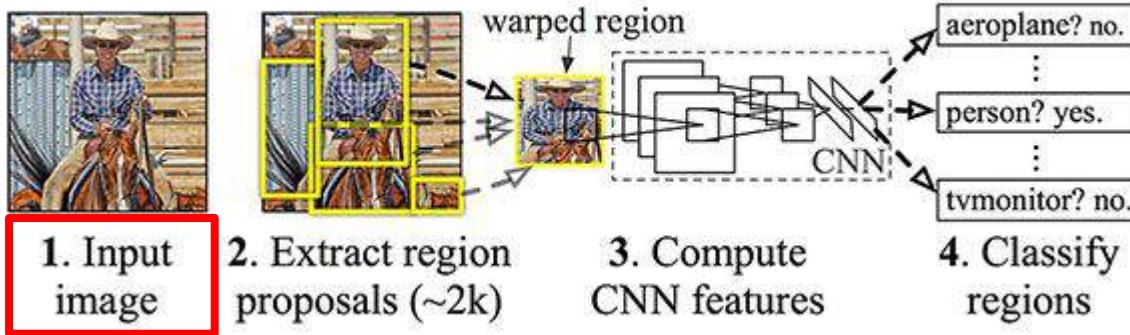
## R-CNN: *Regions with CNN features*



The R-CNN architecture (source: Girshick et al., 2013)

# R-CNN

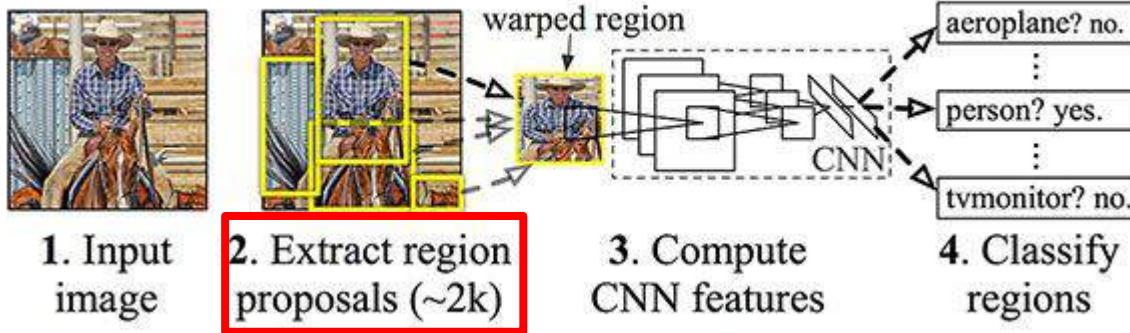
## R-CNN: *Regions with CNN features*



The R-CNN architecture (source: [Girshick et al., 2013](#))

# R-CNN

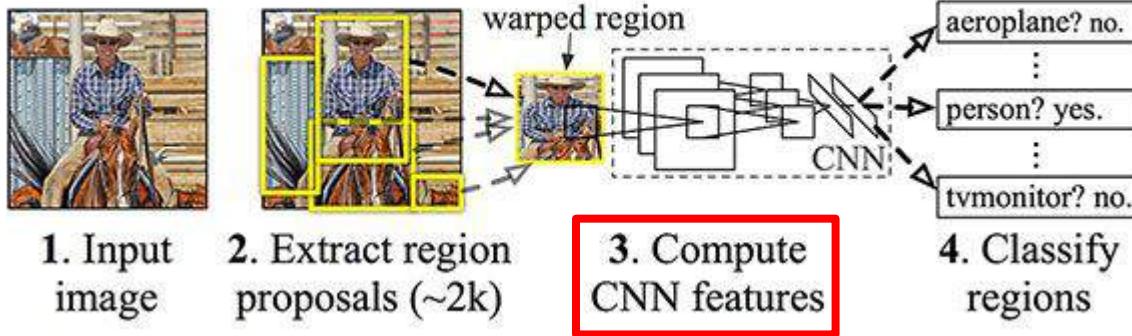
## R-CNN: *Regions with CNN features*



The R-CNN architecture (source: [Girshick et al., 2013](#))

# R-CNN

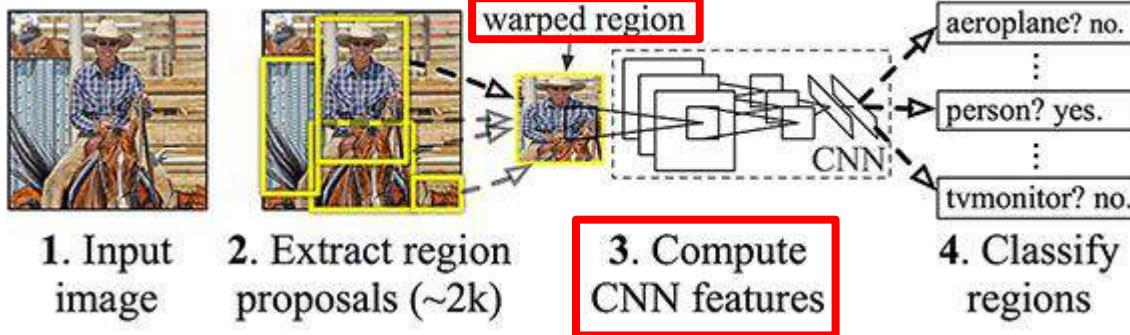
## R-CNN: *Regions with CNN features*



The R-CNN architecture (source: [Girshick et al., 2013](#))

# R-CNN

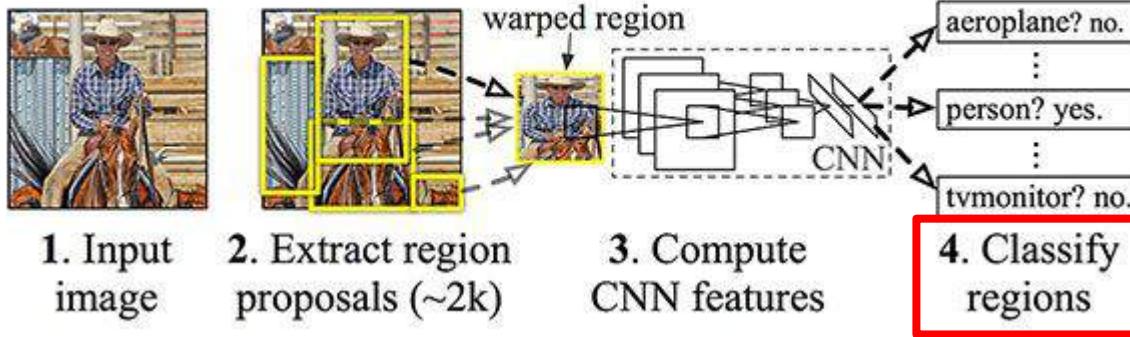
## R-CNN: *Regions with CNN features*



The R-CNN architecture (source: Girshick et al., 2013)

# R-CNN

## R-CNN: *Regions with CNN features*



The R-CNN architecture (source: Girshick et al., 2013)

# Rich feature hierarchies for accurate object detection and semantic segmentation

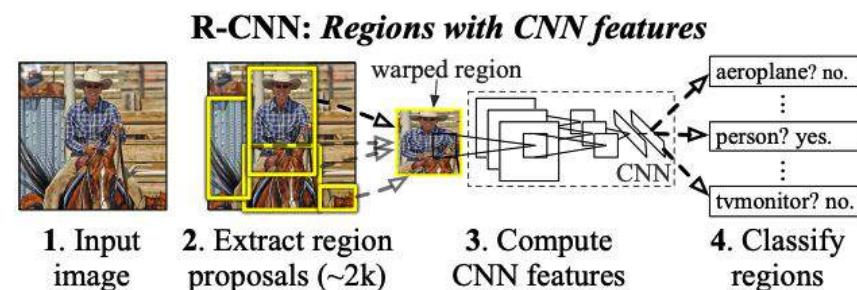
## Tech report (v5)

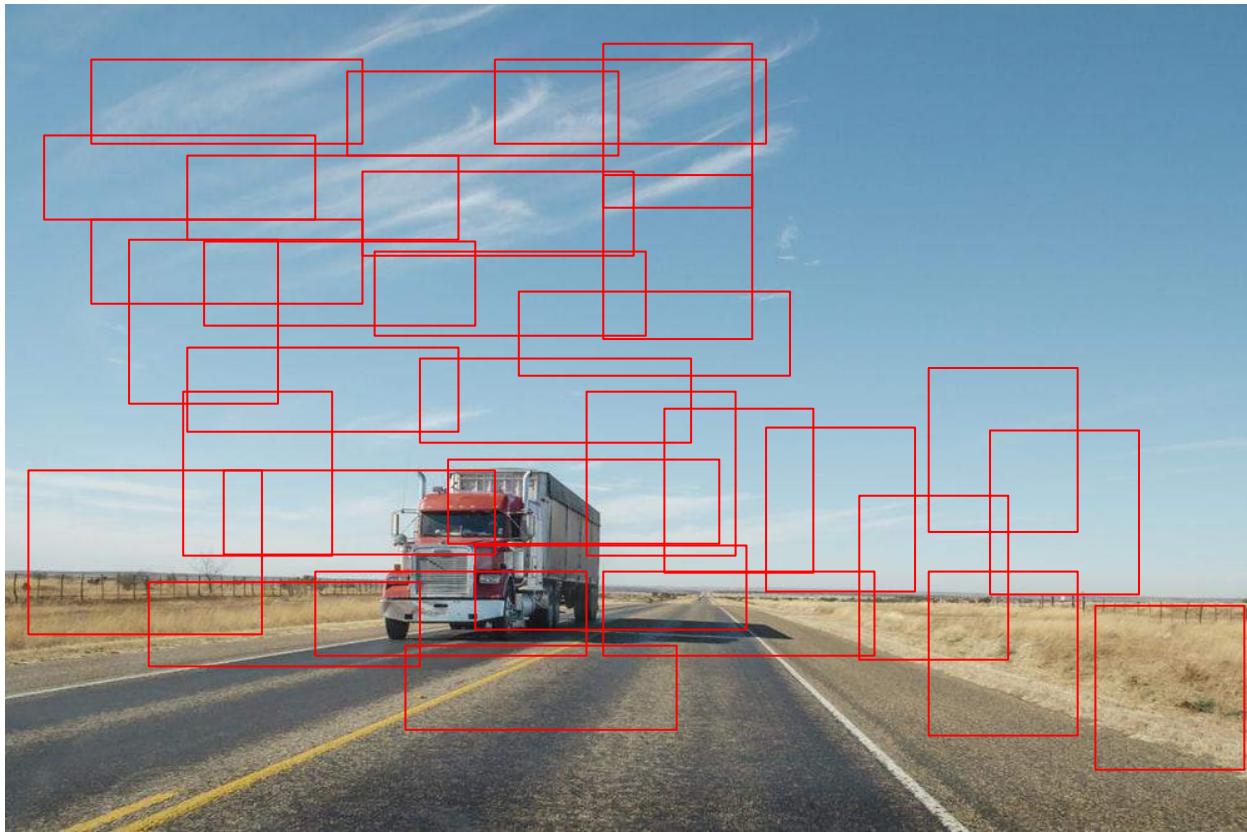
Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik  
UC Berkeley

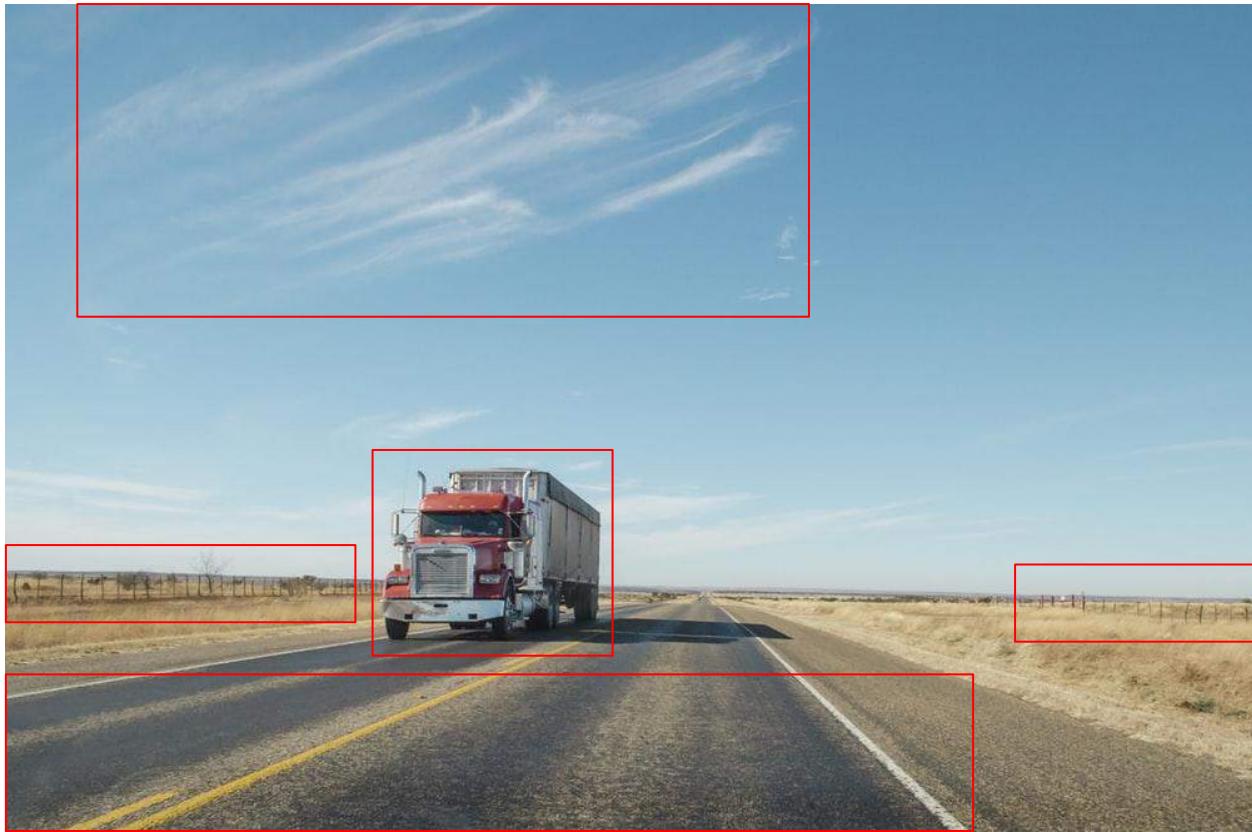
{rbg, jdonahue, trevor, malik}@eecs.berkeley.edu

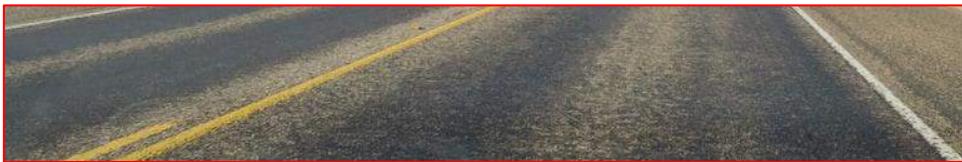
### Abstract

Object detection performance, as measured on the canonical PASCAL VOC dataset, has plateaued in the last few years. The best-performing methods are complex ensemble systems that typically combine multiple low-level image features with high-level context. In this paper, we

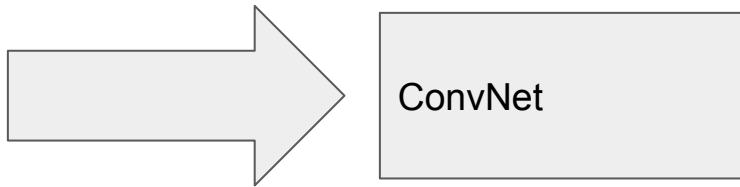




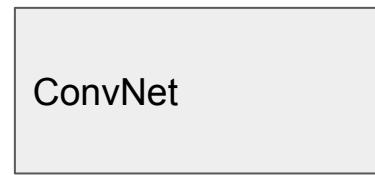




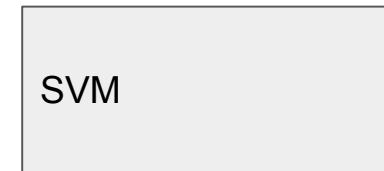
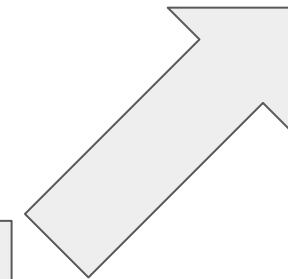




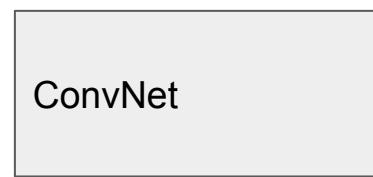
Feature Extraction



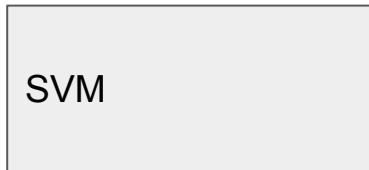
Feature Extraction



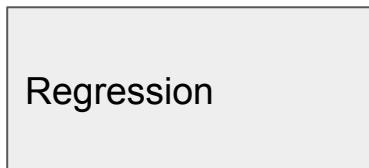
Classification



Feature Extraction



Classification



Bounding Boxes

# Transfer Learning for R-CNN

Pre-train: auxiliary task



Large auxiliary Data Set

Fine-tune: domain specific task



Warped region proposals

# Transfer Learning for R-CNN

Pre-train: auxiliary task



Large auxiliary Data Set

Fine-tune: domain specific task

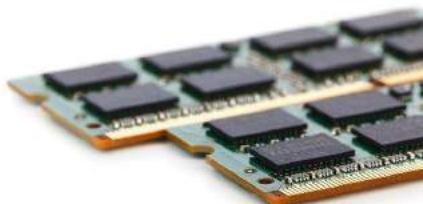


Warped region proposals

# R-CNN disadvantages

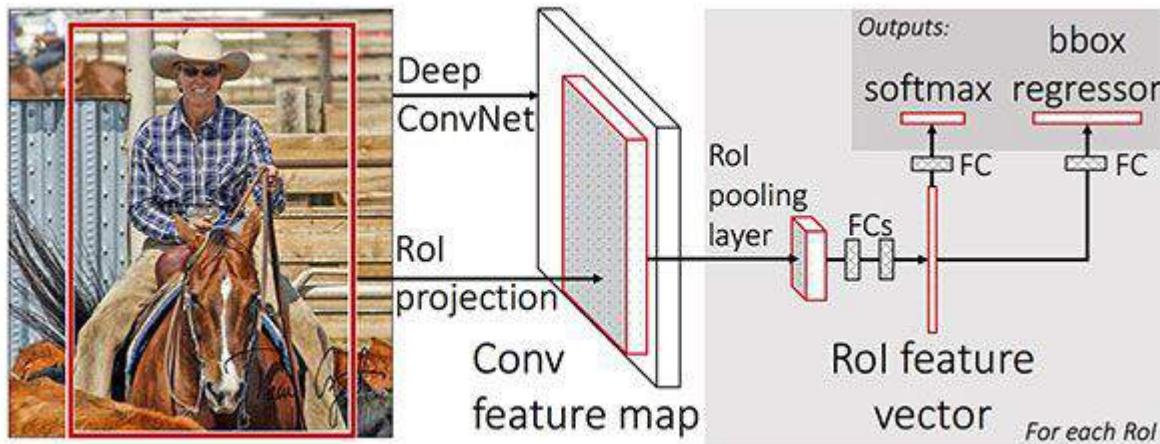


Slow



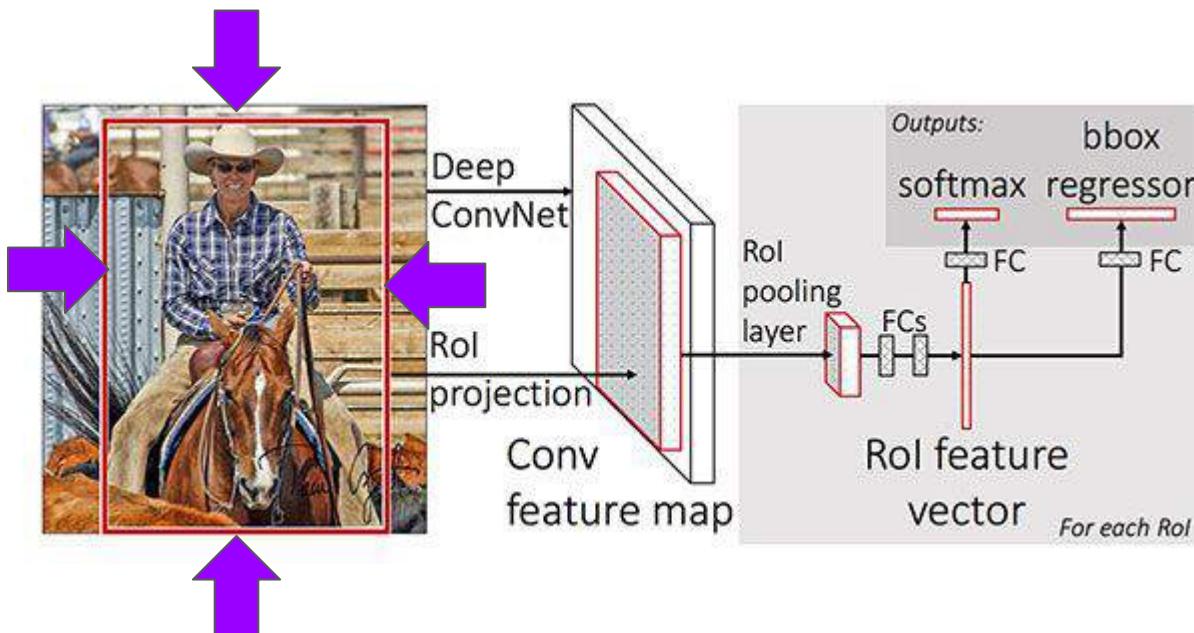
memory

# Fast R-CNN



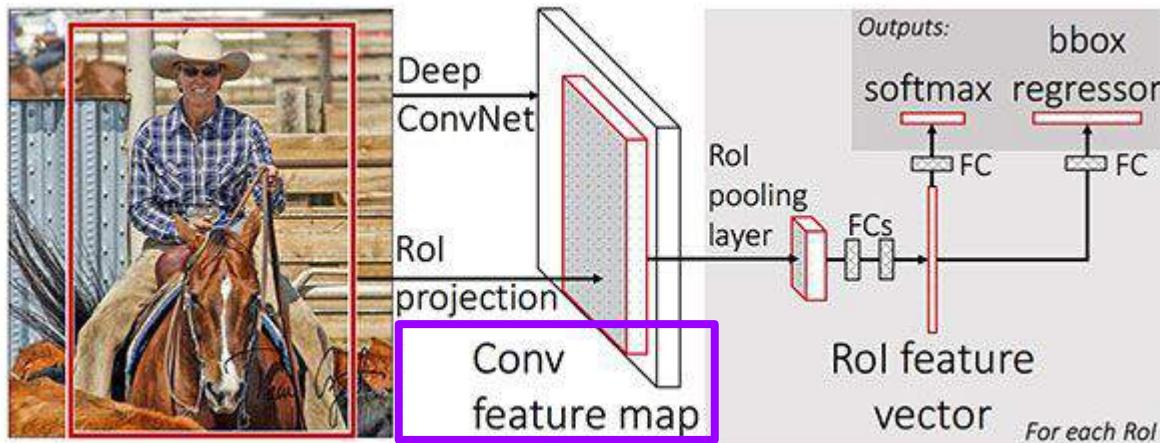
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



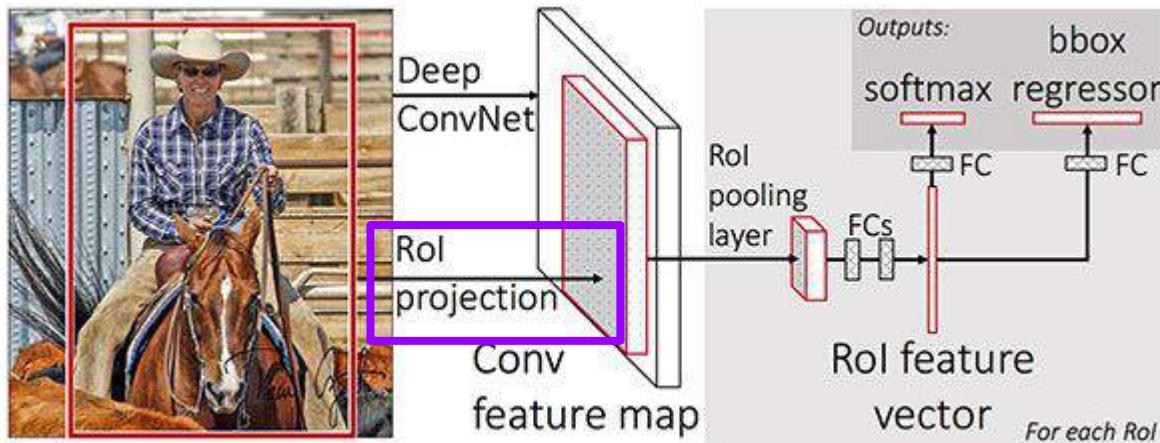
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



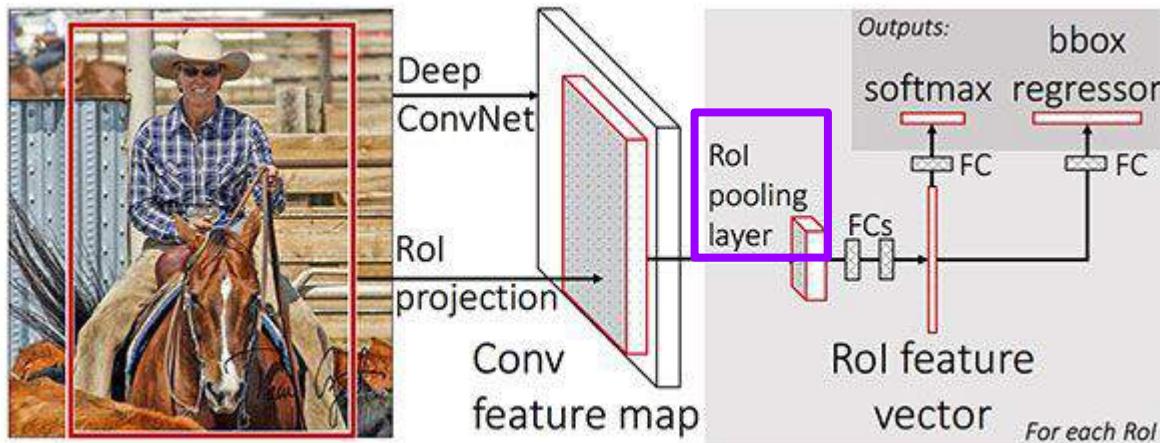
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



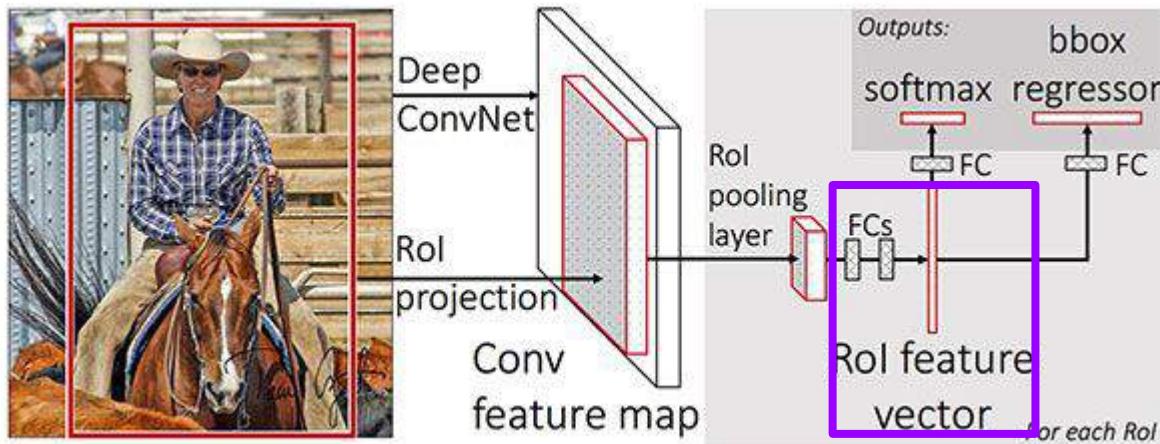
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



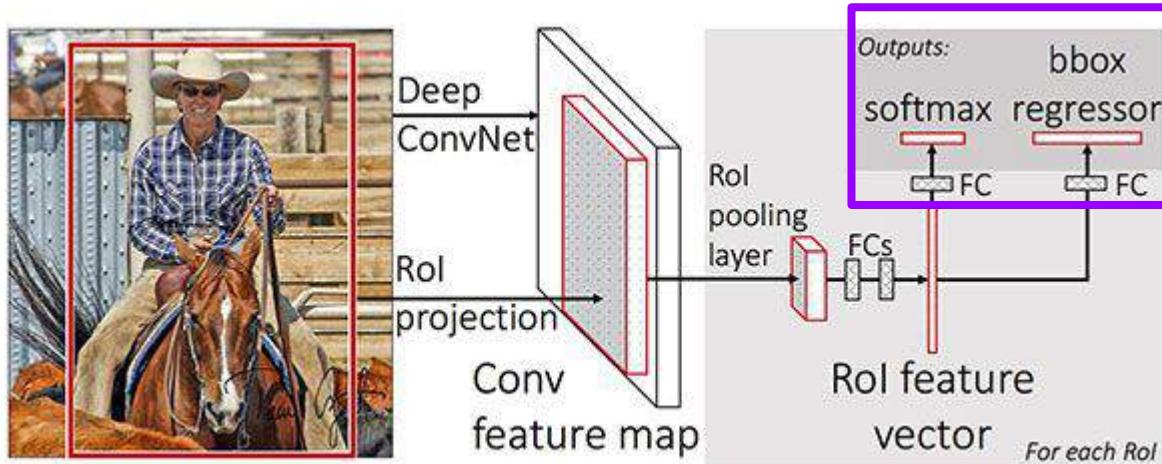
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



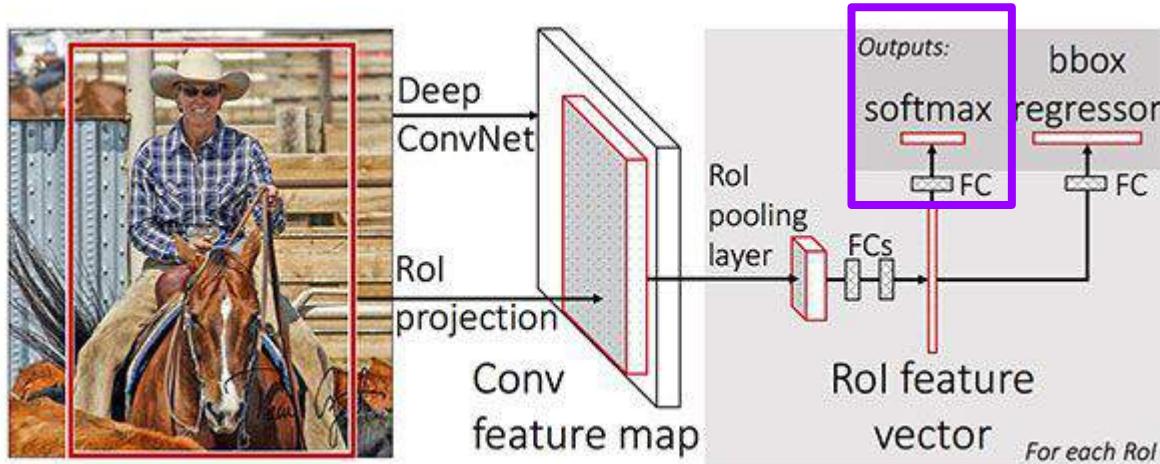
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



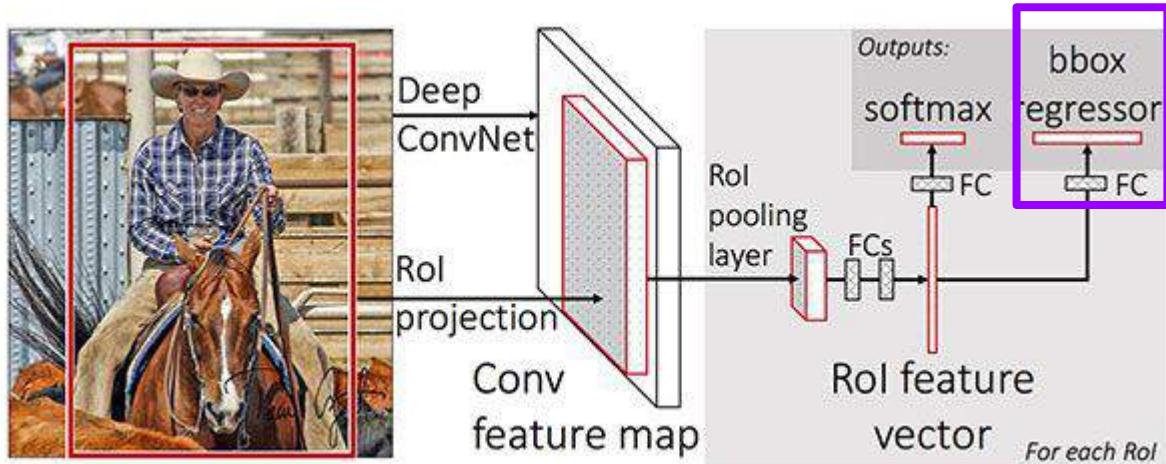
The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

# Fast R-CNN



The Fast R-CNN architecture (source: [Girshick et al., 2015](#)).

<https://arxiv.org/pdf/1504.08083.pdf>

# Fast R-CNN

Ross Girshick  
Microsoft Research

rbg@microsoft.com

## Abstract

*This paper proposes a Fast Region-based Convolutional Network method (Fast R-CNN) for object detection. Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. Fast R-CNN trains the very deep VGG16 network 9× faster than R-CNN, is 213× faster at test-time, and achieves a higher mAP on PASCAL VOC 2012. Compared to SPPnet, Fast R-CNN trains VGG16 3×*

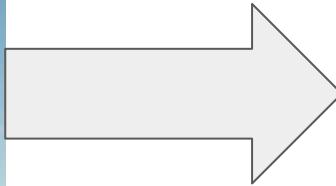
while achieving top accuracy on PASCAL VOC 2012 [7] with a mAP of 66% (vs. 62% for R-CNN).<sup>1</sup>

## 1.1. R-CNN and SPPnet

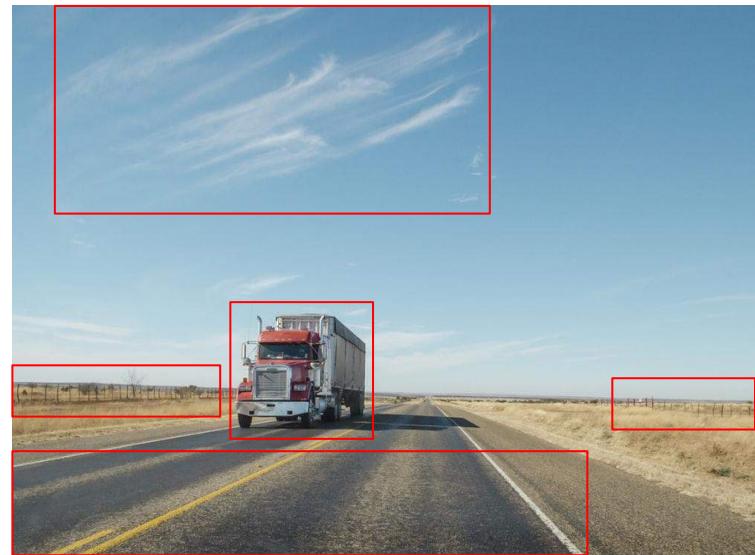
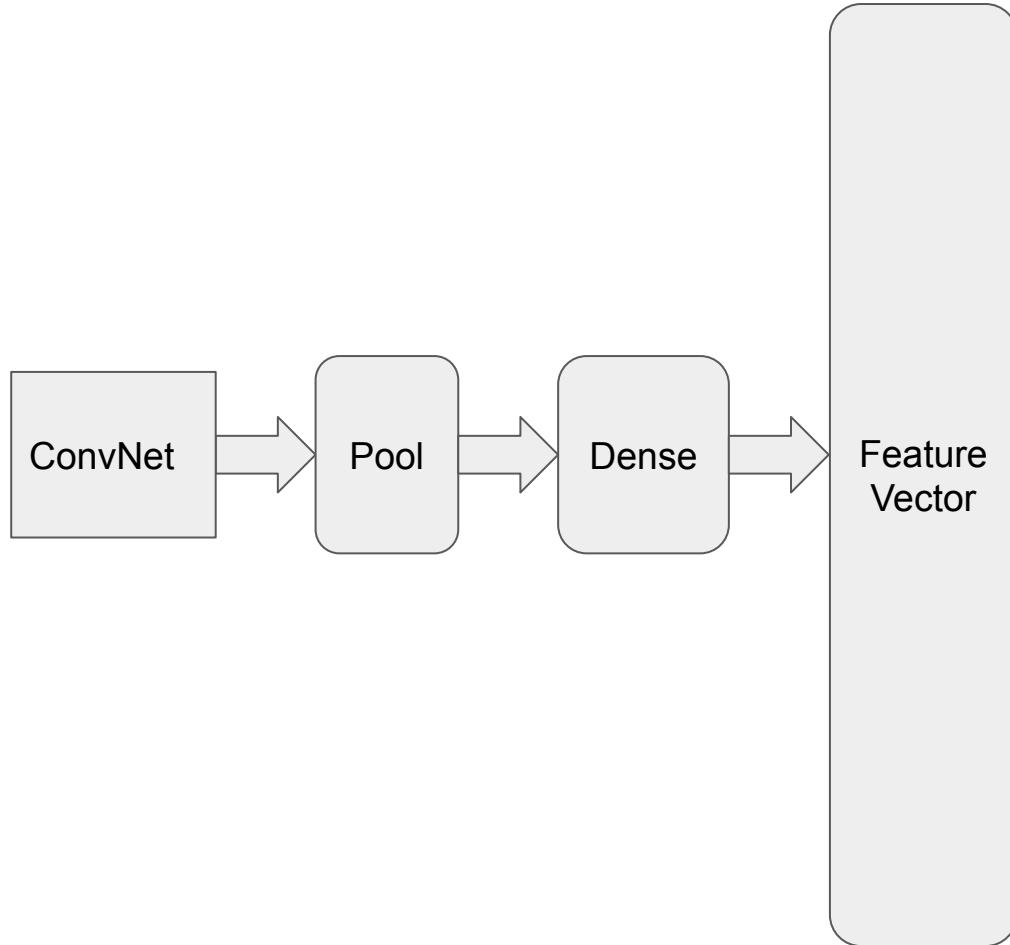
The Region-based Convolutional Network method (R-CNN) [9] achieves excellent object detection accuracy by using a deep ConvNet to classify object proposals. R-CNN, however, has notable drawbacks:

1. **Training is a multi-stage pipeline.** R-CNN first fine-tunes a ConvNet on object proposals using log loss. Then, it fits SVMs to ConvNet features. These SVMs





Feature Extraction

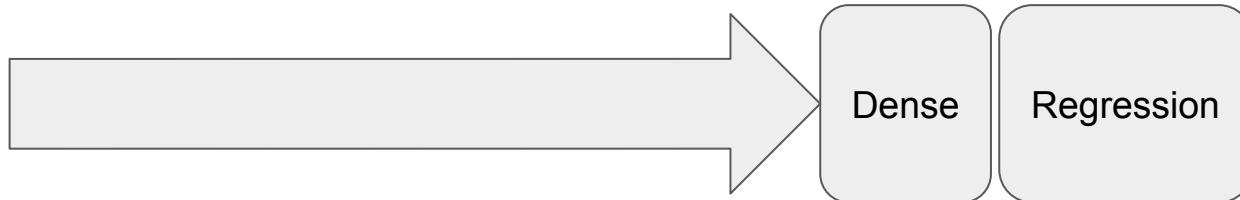




Feature  
Vector

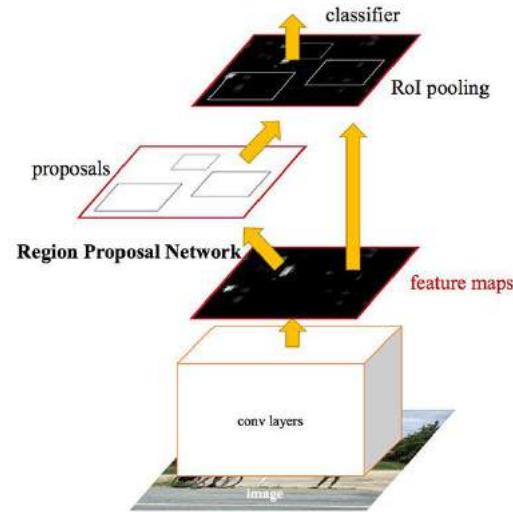


Classification

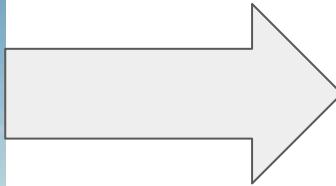


Bounding Boxes

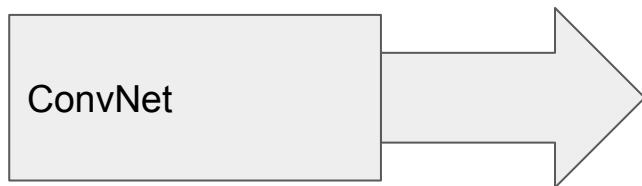
# Faster R-CNN



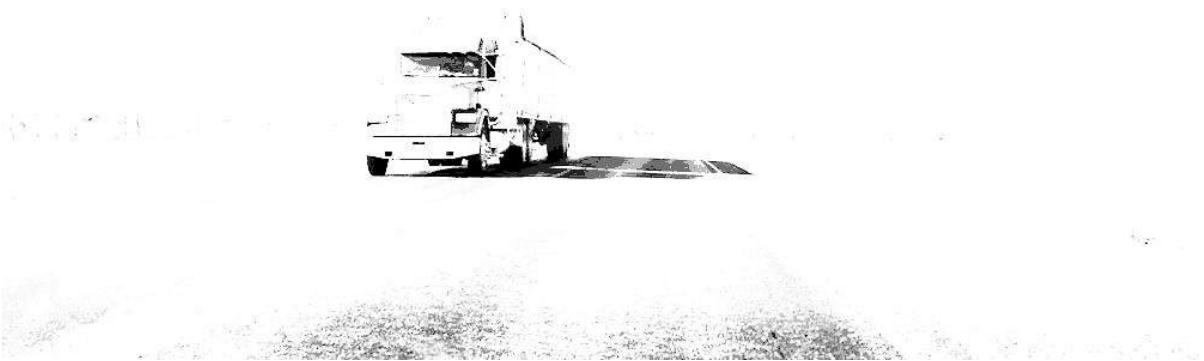
source: Deep Learning for Computer Vision with Python

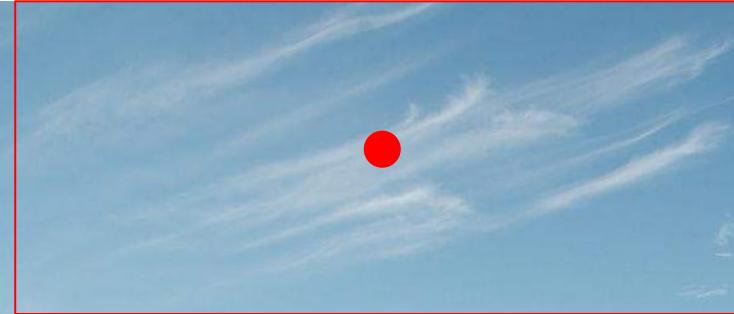
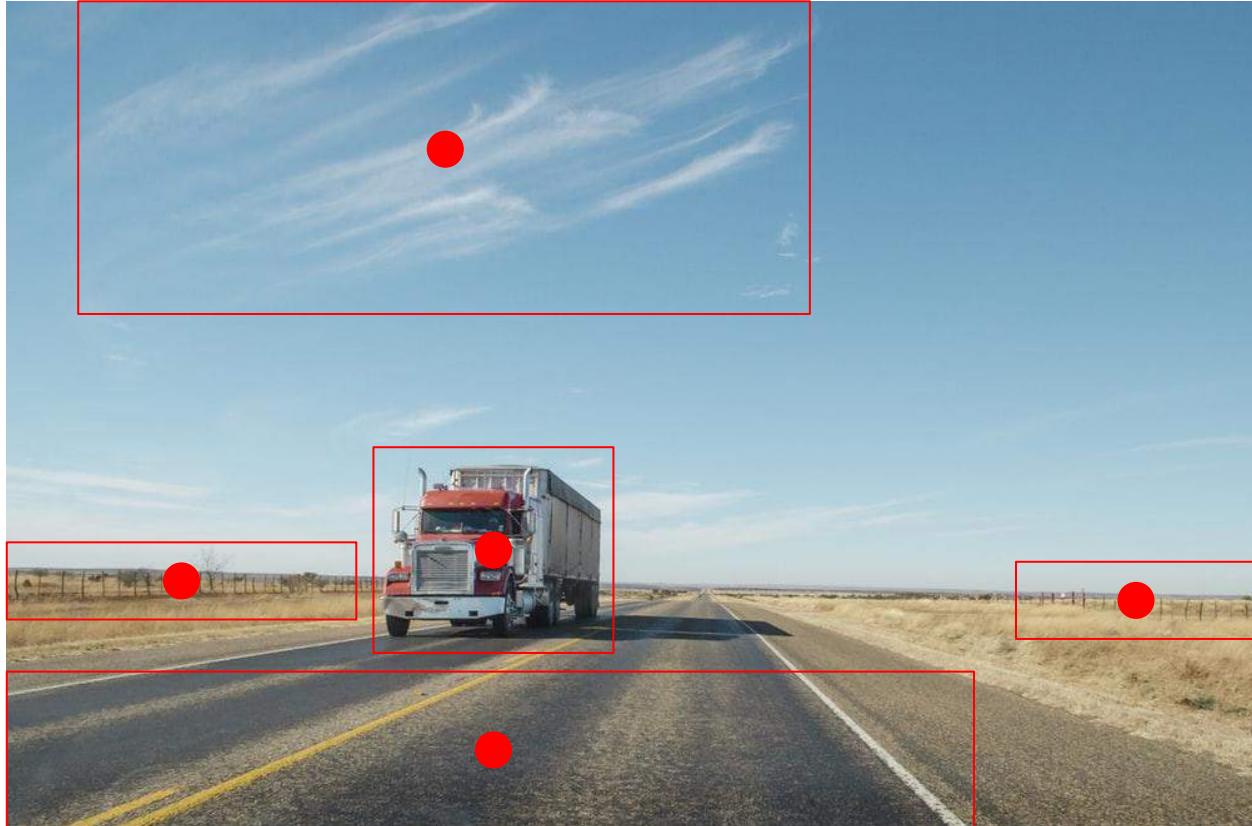


Feature Extraction



Feature Extraction







Classification



Bounding Boxes



```
import tensorflow as tf  
import tensorflow_hub as hub
```

```
module_handle = ...
```

```
detector = ...
```

```
import tensorflow as tf  
import tensorflow_hub as hub
```

```
module_handle = ...
```

```
detector = ...
```

Back

Image object detection

## faster\_rcnn/openimages\_v4/inception\_resnet\_v2

Object detection model trained on Open Images V4 with ImageNet pre-trained Inception Resnet V2 as image feature extractor.

Publisher: Google Updated: 08/25/2020 License: Apache-2.0

Architecture: Dataset:

Faster R-CNN

OpenImagesV4

Go to:

<https://www.tensorflow.org/hub>



### FasterRCNN Openimages v4

Results

Label: Boat

Label: Person

Drag and drop an image

Browse

URL

← Back



Image object detection

**faster\_rcnn/openimages\_v4/i**

# <https://tfhub.dev/s?module-type=image-object-detection>

Filters [Clear all](#)

**Problem domain** [Image object detection](#) [X](#)

**Model format** [TF.js](#) [TFLite](#) [Coral](#)

**TF Version** [?](#) [TF1](#) [TF2](#)

**Fine tunable** [Toggle](#)

**Architecture** [?](#)

**Publisher** [?](#)

**Dataset** [?](#)

**Language** [?](#)

-  Collection

### object\_detection

Publisher: [TensorFlow](#) Models: 39 Updated: 08/14/2020  
Collection of TensorFlow 2 Object Detection models trained on COCO 2017 dataset.
-  Collection

### lite/task-library/object-detector

Publisher: [TensorFlow](#) Models: 2 Updated: 09/09/2020  
Collection of TensorFlow Lite Task Library compatible models for object detection.
-  Image object detection

### faster\_rcnn/openimages\_v4 /inception\_resnet\_v2

Publisher: [Google](#) Updated: 08/25/2020  
Object detection model trained on Open Images V4 with ImageNet pre-trained Inception Resnet V2 as image feature extractor.
- Architecture: [Faster R-CNN](#) | Dataset: [OpenImagesV4](#)
-  Image object detection

### mask\_rcnn /inception\_resnet\_v2\_1024x1024

Publisher: [TensorFlow](#) Updated: 08/25/2020  
Mask R-CNN Object detection model, trained on COCO 2017 dataset.
- Architecture: [Faster R-CNN](#)
-  Image object detection

### faster\_rcnn /inception\_resnet\_v2\_1024x1024

Publisher: [TensorFlow](#) Updated: 08/25/2020  
Faster R-CNN with Resnet V2 Object detection model, trained on COCO 2017 dataset with training images scaled to 1024x1024.
- Architecture: [Faster R-CNN](#)

# Hub module (v1)

---

Fine tunable: No License: [Apache-2.0](#)

Last updated: 08/25/2020

Format: Hub module

Object detection model trained on Open Images V4 with ImageNet pre-trained Inception Resnet V2 as image feature extractor.

...ges\_v4/inception\_resnet\_v2/1

[Copy URL](#) 

[Download](#)  228.63MB

[Open Colab Notebook](#) 

```
import tensorflow as tf
import tensorflow_hub as hub

module_handle =
"https://tfhub.dev/google/faster_rcnn/openimages_v4/
    inception_resnet_v2/1"

detector = ...
```

```
import tensorflow as tf
import tensorflow_hub as hub

module_handle =
"tfhub.dev/google/faster_rcnn/openimages_v4/
 inception_resnet_v2/1"

detector = ...
```

```
import tensorflow as tf
import tensorflow_hub as hub

module_handle =
"tfhub.dev/google/faster_rcnn/openimages_v4/
    inception_resnet_v2/1"

detector = hub.load(module_handle).signatures['default']
```

# Hub module (v1)

---

Fine tunable: No License: [Apache-2.0](#)

Last updated: 08/25/2020

Format: Hub module

Object detection model trained on Open Images V4 with ImageNet pre-trained Inception Resnet V2 as image feature extractor.

...ges\_v4/inception\_resnet\_v2/1

[Copy URL](#) 

[Download](#)  228.63MB

[Open Colab Notebook](#) 

# File:20130807 dublino14.JPG

From Wikimedia Commons, the free media repository

[File](#) [File history](#) [File usage on Commons](#) [Metadata](#)



- [!\[\]\(1a2e2185252feb68138797d44af7a2ee\_img.jpg\) Download  
all sizes](#)
- [!\[\]\(37536ec7013eab330b9dd872965b19a6\_img.jpg\) Use this file  
on the web](#)
- [!\[\]\(a9996f890c060f34257b6ba9290de0f7\_img.jpg\) Use this file  
on a wiki](#)
- [!\[\]\(3d2249a53f3b6448bd62e14ad28d6762\_img.jpg\) Email a link  
to this file](#)
- [!\[\]\(07331d43c7976a43973f36f1dc53e5ef\_img.jpg\) Information  
about reusing](#)

# File:20130807 dublino14.JPG

From Wikimedia Commons, the free media repository

File    File history    File usage on Commons    Metadata



**Use this file on the web**

**Page URL:** [https://commons.wikimedia.org/wiki/File:20130807\\_dublino14.JPG](https://commons.wikimedia.org/wiki/File:20130807_dublino14.JPG)

**File URL:** [https://upload.wikimedia.org/wikipedia/commons/f/fb/20130807\\_dublino14.JPG](https://upload.wikimedia.org/wikipedia/commons/f/fb/20130807_dublino14.JPG)

**Attribution:** Jean Housen / CC BY-SA

Jean Housen / CC BY-SA (<https://creativecommons.org/licenses/by-sa/3.0>)  HTML

**Embed this file**  HTML  BBCode 512px wide ▾

**Download**  
all sizes

**Use this file on the web** (Red box)

**Use this file on a wiki**

**Email a link to this file**

**Information about reusing**

```
image_url =  
"https://upload.wikimedia.org/wikipedia/commons/f/fb/20130807_dublin014.JPG"
```

```
downloaded_image_path = ...
```

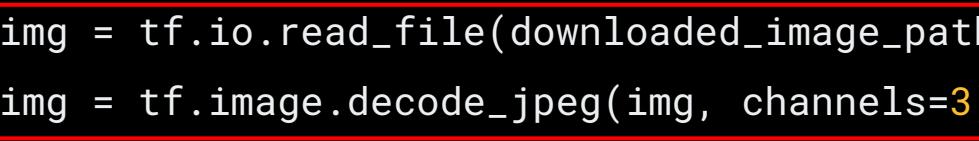
```
image_url =  
"https://upload.wikimedia.org/wikipedia/commons/f/fb/20130807_dublin014.JPG"  
  
downloaded_image_path = download_and_resize_image(url=image_url  
                                                 new_width=3872,  
                                                 new_height=2592)
```



Size of this preview: 800 × 536 pixels. Other resolutions: 320 × 214 pixels | 640 × 428 pixels | 1,024 × 685 pixels | 1,280 × 857 pixels.  
Original file (3,872 × 2,592 pixels, file size: 4.26 MB, MIME type: image/jpeg); ZoomViewer: flash/no flash

```
img = tf.io.read_file(downloaded_image_path)
img = tf.image.decode_jpeg(img, channels=3)
converted_img = tf.image.convert_image_dtype(img, tf.float32)[tf.newaxis, ...]

result = detector(converted_img)
```

```
img = tf.io.read_file(downloaded_image_path)

    img = tf.image.decode_jpeg(img, channels=3)
converted_img = tf.image.convert_image_dtype(img, tf.float32)[tf.newaxis, ...]

result = detector(converted_img)
```

```
img = tf.io.read_file(downloaded_image_path)
img = tf.image.decode_jpeg(img, channels=3)
converted_img = tf.image.convert_image_dtype(img, tf.float32)[tf.newaxis, ...]
```

```
result = detector(converted_img)
```

```
img = tf.io.read_file(downloaded_image_path)
img = tf.image.decode_jpeg(img, channels=3)
converted_img = tf.image.convert_image_dtype(img, tf.float32)[tf.newaxis, ...]

result = detector(converted_img)
```

Found 100 objects.

```
[0.43670595 0.34758776 0.2438663  0.23315561 0.22782972 0.21416378  
 0.2057755  0.20488328 0.20278934 0.19843656 0.18925622 0.18167153  
...]  
[b'Person' b'Footwear' b'Footwear' b'Building' b'Person' b'Footwear'  
 b'Window' b'Building' b'Person' b'Window' b'Window' b'Window' b'Window'  
...]  
[[0.5130533  0.9170097  0.82187796 0.99240506]  
 [0.80095136 0.9544444   0.83115625 0.98134536]  
 [0.79767334 0.94279504 0.8265182  0.9654046 ]
```

Found 100 objects.

Probability	Class	Bounding box
0.43670595	b'Person'	[0.5130533 0.9170097 0.82187796 0.99240506]
0.34758776	b'Footwear'	[0.80095136 0.954444 0.83115625 0.98134536]
0.2438663	b'Footwear'	[0.79767334 0.94279504 0.8265182 0.9654046 ]
0.23315561	b'Building'	
0.22782972	b'Person'	
0.21416378	b'Footwear'	
0.2057755	b'Window'	
0.20488328	b'Building'	
0.20278934	b'Person'	
0.19843656	b'Window'	
0.18925622	b'Window'	
0.18167153	b'Window'	
...	...	...

```
# Clone the tensorflow models repository  
!git clone --depth 1 https://github.com/tensorflow/models
```

# # Clone the tensorflow models repository

```
!git clone --depth 1 https://github.com/tensorflow/models
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Shows the title "git clone tensorflow models.ipynb" and a star icon.
- File Menu:** File, Edit, View, Insert, Runtime, Tools, Help.
- Status Bar:** "All changes saved".
- Toolbar:** Includes icons for search, upload, download, and file operations.
- Left Sidebar (Files):** A tree view of the directory structure:
  - bin
  - boot
  - content
    - models
      - community
      - official
    - orbit
    - research
  - a3c\_blogpostA folder icon in the sidebar is highlighted with a red box.
- Code Cell Output:** Shows the command run and its output:

```
[1] !git clone --depth 1 https://github.com/tensorflow/models
Cloning into 'models'...
remote: Enumerating objects...
remote: Counting objects...
remote: Compressing objects...
remote: Total 2243 (delta 0)
Receiving objects: 100% (2243/2243)
Resolving deltas: 100% (55/55)
```
- Bottom Cell:** Shows the command "!pwd" and its output "/content".

# # Clone the tensorflow models repository

```
!git clone --depth 1 https://github.com/tensorflow/models
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Shows the title "git clone tensorflow models.ipynb" and a star icon.
- File Menu:** File, Edit, View, Insert, Runtime, Tools, Help.
- Status Bar:** Shows "All changes saved".
- Toolbar:** Includes icons for file operations like upload, download, and open.
- Files Panel:** Shows a tree view of the directory structure:
  - bin
  - boot
  - content (highlighted with a red box)
  - models (highlighted with a red box)
  - community
  - official
  - orbit
  - research
  - a3c\_blogpost
- Code Cell:** Displays the command: [1] `!git clone --depth 1 https://github.com/tensorflow/models`.  
Output: Cloning into 'models'...  
remote: Enumerating objects...  
remote: Counting objects...  
remote: Compressing objects...  
remote: Total 2243 (delta 0)  
Receiving objects: 100% (2243/2243)  
Resolving deltas: 100% (55/55)
- Code Cell:** Displays the command: !pwd  
Output: /content

# # Clone the tensorflow models repository

```
!git clone --depth 1 https://github.com/tensorflow/models
```

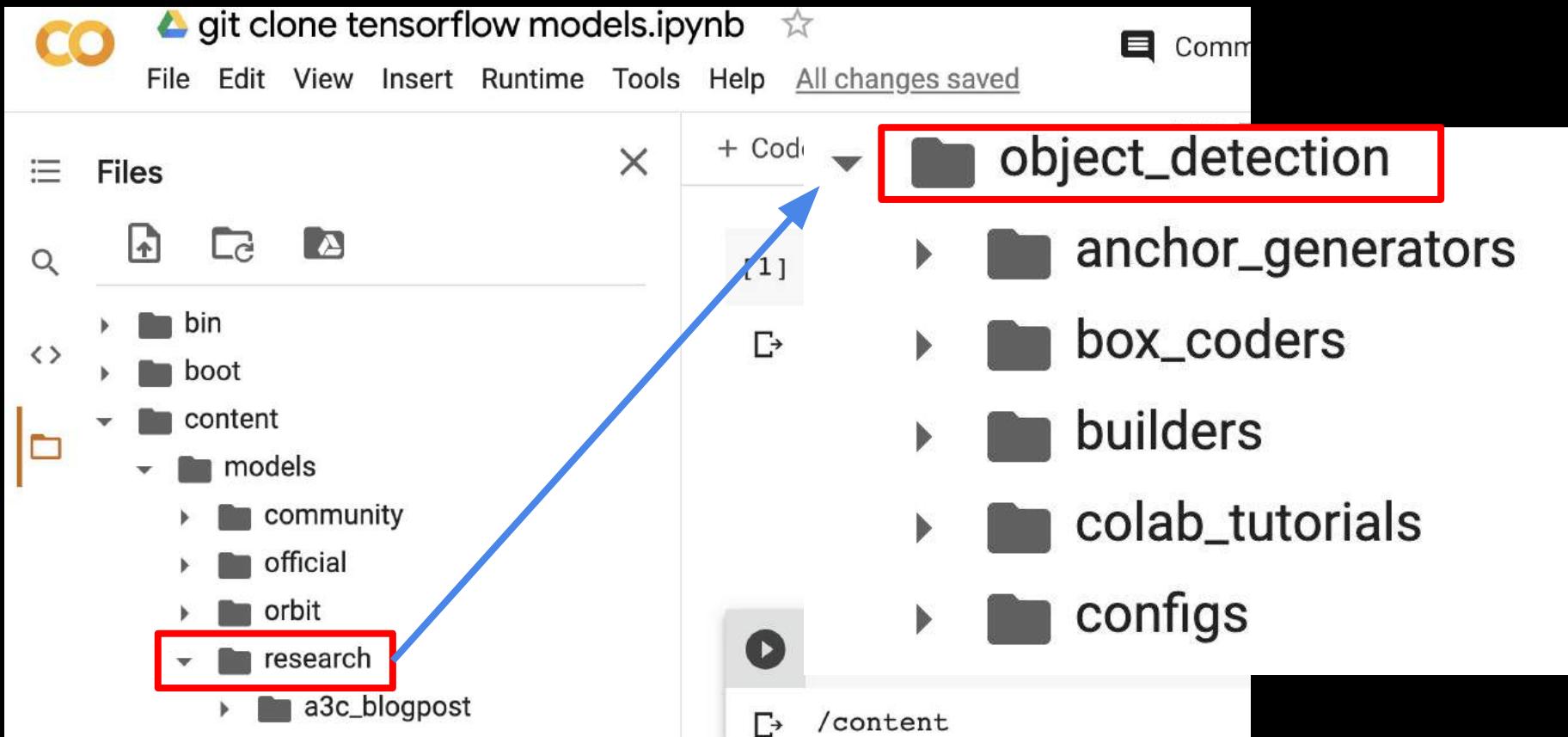
The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Shows the title "git clone tensorflow models.ipynb" and a star icon.
- File Menu:** File, Edit, View, Insert, Runtime, Tools, Help, All changes saved.
- Comm:** A message icon indicating communication status.
- RAM Disk:** RAM is checked.
- Files Panel:** Shows a tree view of the directory structure:
  - bin
  - boot
  - content
    - models
      - community
      - official
      - orbit
    - research
  - a3c\_blogpostA red box highlights the "research" folder.
- Code Cell Output:** Displays the command run and its output:

```
[1] !git clone --depth 1 https://github.com/tensorflow/models
Cloning into 'models'...
remote: Enumerating objects...
remote: Counting objects...
remote: Compressing objects...
remote: Total 2243 (delta 0)
Receiving objects: 100% (2243/2243)
Resolving deltas: 100% (55/55)
```
- Play Button:** A play button icon followed by the command "!pwd".
- Output Cell:** Displays the current working directory: "/content".

```
# Clone the tensorflow models repository
```

```
!git clone --depth 1 https://github.com/tensorflow/models
```



```
%%bash
sudo apt install -y protobuf-compiler
cd models/research/
protoc object_detection/protos/*.proto --python_out=.
cp object_detection/packages/tf2/setup.py .
python -m pip install .
```

```
%%bash
```

```
sudo apt install -y protobuf-compiler
```

```
cd models/research/
```

```
protoc object_detection/protos/*.proto --python_out=.
```

```
cp object_detection/packages/tf2/setup.py .
```

```
python -m pip install .
```

```
%%bash
```

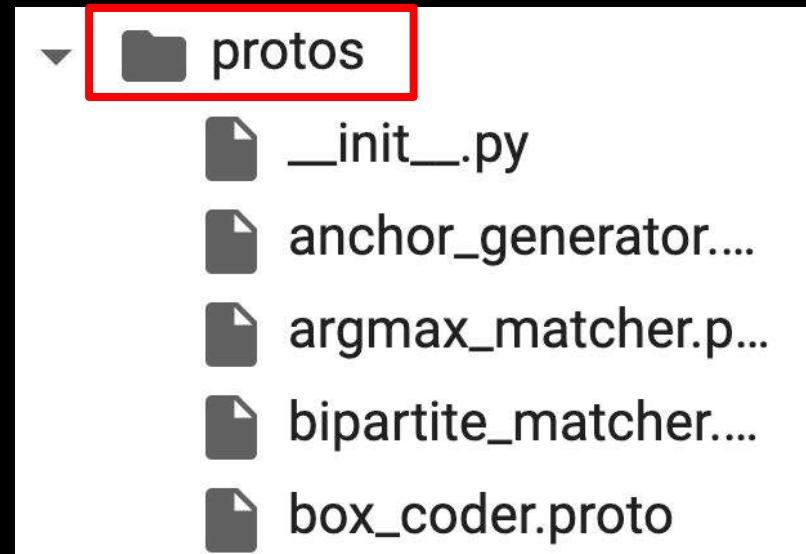
```
sudo apt install -y protobuf-compiler
```

```
cd models/research/
```

```
protoc object_detection/protos/*.proto --python_out=.
```

```
cp object_detection/packages/tf2/setup.py .
```

```
python -m pip install .
```



```
%%bash
```

```
sudo apt install -y protobuf-compiler
```

```
cd models/research/
```

```
protoc object_detection/protos/*.proto --python_out=.
```

```
cp object_detection/packages/tf2/setup.py .
```

```
python -m pip install .
```

```
%%bash
```

```
sudo apt install -y protobuf-compiler
```

```
cd models/research/
```

```
protoc object_detection/protos/*.proto --python_out=.
```

```
cp object_detection/packages/tf2/setup.py .
```

```
python -m pip install .
```

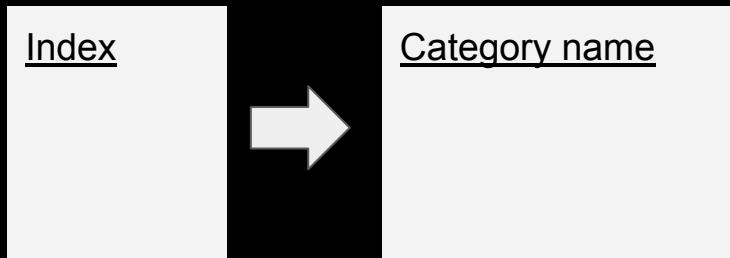
```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

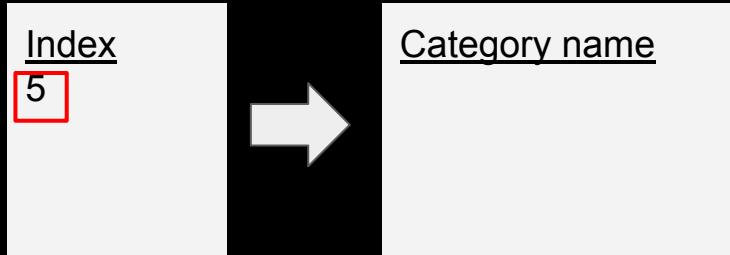
```
from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as viz_utils
from object_detection.utils import ops as utils_ops
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

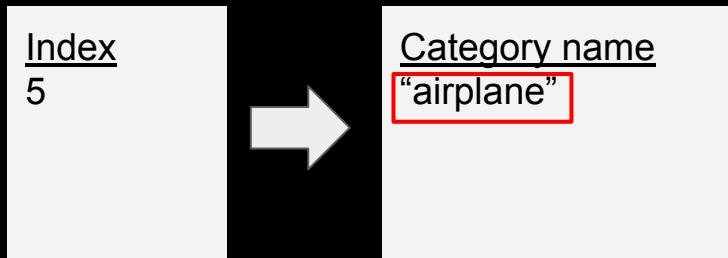
```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```



```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```



```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```



```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

pbtxt

```
    item {  
        name: "/m/04_sv"  
        id: 4  
        display_name: "motorcycle"  
    }  
    item {  
        name: "/m/05czz6l"  
        id: 5  
        display_name: "airplane"  
    }
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

pbtxt

```
item {  
    name: "/m/04_sv"  
    id: 4  
    display_name: "motorcycle"  
}  
item {  
    name: "/m/05czz6l"  
    id: 5  
    display_name: "airplane"  
}
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
PATH_TO_LABELS =  
'./models/research/object_detection/data/mscoco_label_map.pbtxt'
```

```
category_index = label_map_util.create_category_index_from_labelmap(  
    PATH_TO_LABELS, use_display_name=True)
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
PATH_TO_LABELS =  
'./models/research/object_detection/data/mscoco_label_map.pbtxt'
```

```
category_index = label_map_util.create_category_index_from_labelmap(  
    PATH_TO_LABELS, use_display_name=True)
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
PATH_TO_LABELS =  
'./models/research/object_detection/data/mscoco_label_map.pbtxt'
```

```
category_index = label_map_util.create_category_index_from_labelmap(  
    PATH_TO_LABELS, use_display_name=True)
```

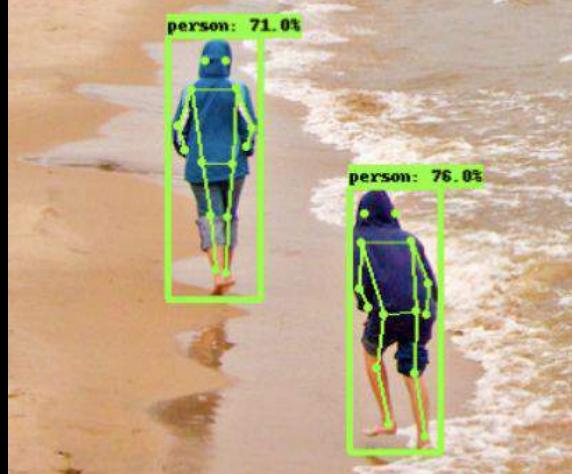
```
category_index
```

```
{1: {'id': 1, 'name': 'person'},  
 2: {'id': 2, 'name': 'bicycle'},  
 3: {'id': 3, 'name': 'car'},  
 4: {'id': 4, 'name': 'motorcycle'},  
 5: {'id': 5, 'name': 'airplane'},
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=...,  
    boxes=...,  
    classes=...,  
    scores=...,  
    ...)
```



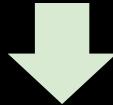
```
from object_detection.utils import label_map_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import ops as utils_ops
```

```
results[ 'detection_scores' ]
```

```
<tf.Tensor: shape=(1, 100), dtype=float32, numpy=
array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
```

```
results['detection_scores']
```

```
<tf.Tensor: shape=(1, 100), dtype=float32, numpy=  
array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
```

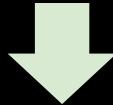


```
result['detection_scores']
```

```
array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
```

```
results[ 'detection_scores' ]
```

```
<tf.Tensor: shape=(1, 100), dtype=float32, numpy=  
array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
```



```
result[ 'detection_scores' ]
```

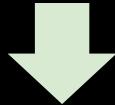
```
array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
```

```
results = hub_model(image_np)
```

```
result = {key:value.numpy() for key,value in results.items()}
```

```
results['detection_scores']

<tf.Tensor: shape=(1, 100), dtype=float32, numpy=
array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
```



```
result['detection_scores']

array([[0.78741133, 0.7599586 , 0.7120807 , 0.7035178 ,
results = hub_model(image_np)

result = {key:value.numpy() for key,value in results.items()}
```

```
result.keys()
```

```
detection_scores
```

```
detection_keypoint_scores
```

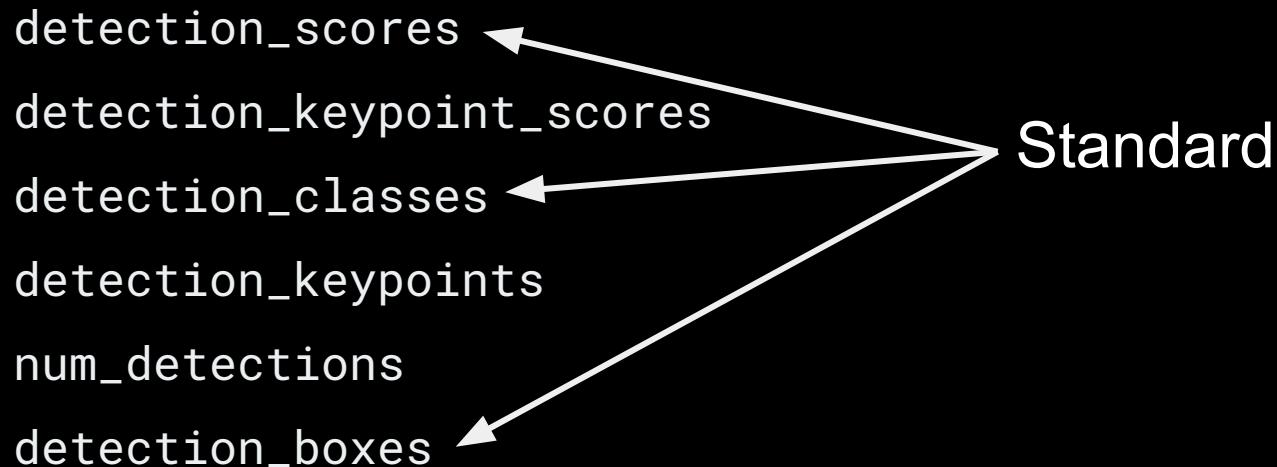
```
detection_classes
```

```
detection_keypoints
```

```
num_detections
```

```
detection_boxes
```

```
result.keys()
```



```
result.keys()
```

detection\_scores

detection\_keypoint\_scores

detection\_classes

detection\_keypoints

num\_detections

detection\_boxes

Standard



```
result.keys()
```

detection\_scores

detection\_keypoint\_scores

detection\_classes

detection\_keypoints

num\_detections

detection\_boxes

Standard

```
result['detection_classes']
```

```
array([[38., 1., 1., 38., 38., 38.,
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=  
    boxes=  
    classes=  
  
    scores=  
    category_index=  
    use_normalized_coordinates=  
    min_score_thresh=  
)  
)
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=  
    classes=  
    scores=  
    category_index=  
    use_normalized_coordinates=  
    min_score_thresh=  
    )
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=  
    scores=  
    category_index=  
    use_normalized_coordinates=  
    min_score_thresh=  
    )
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=  
    category_index=  
    use_normalized_coordinates=  
    min_score_thresh=  
    )
```

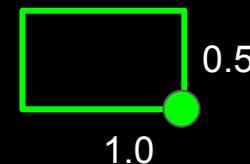
```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=result['detection_scores'][0],  
    category_index=  
    use_normalized_coordinates=  
    min_score_thresh=  
    )
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=result['detection_scores'][0],  
    category_index=category_index,  
    use_normalized_coordinates=  
    min_score_thresh=  
    )
```

```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=result['detection_scores'][0],  
    category_index=category_index,  
    use_normalized_coordinates=True  
    min_score_thresh=  
    )
```

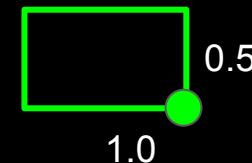
```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=result['detection_scores'][0],  
    category_index=category_index,  
    use_normalized_coordinates=True  
    min_score_thresh=  
    )
```

Normalized

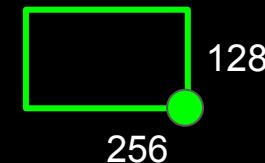


```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=result['detection_scores'][0],  
    category_index=category_index,  
    use_normalized_coordinates=True,  
    min_score_thresh=  
    )
```

Normalized



Denormalized



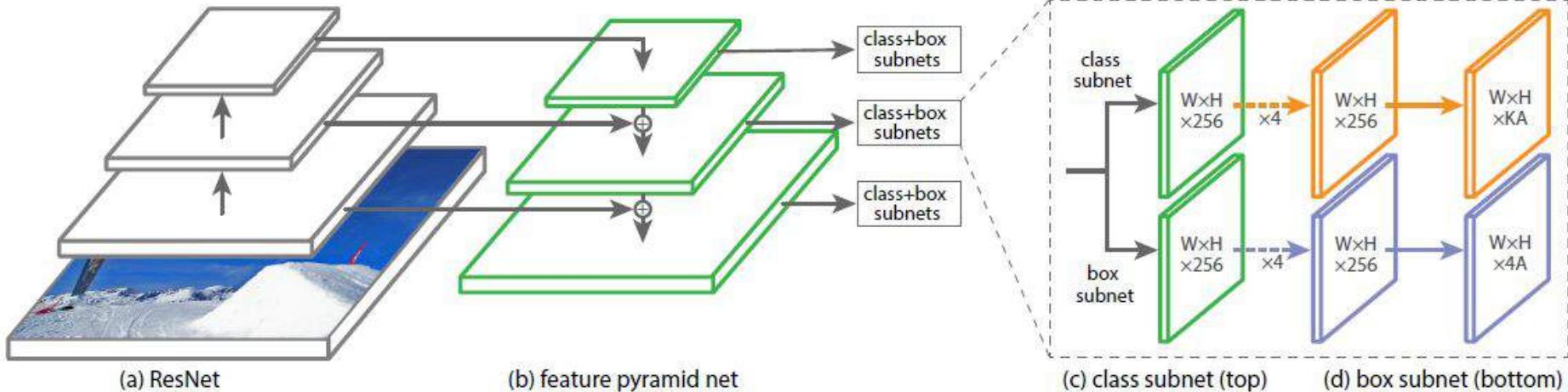
```
viz_utils.visualize_boxes_and_labels_on_image_array(  
    image=image_np_with_detections[0],  
    boxes=result['detection_boxes'][0],  
    classes=(result['detection_classes'][0] +  
             label_id_offset).astype(int),  
    scores=result['detection_scores'][0],  
    category_index=category_index,  
    use_normalized_coordinates=True,  
    min_score_thresh=.40  
)
```

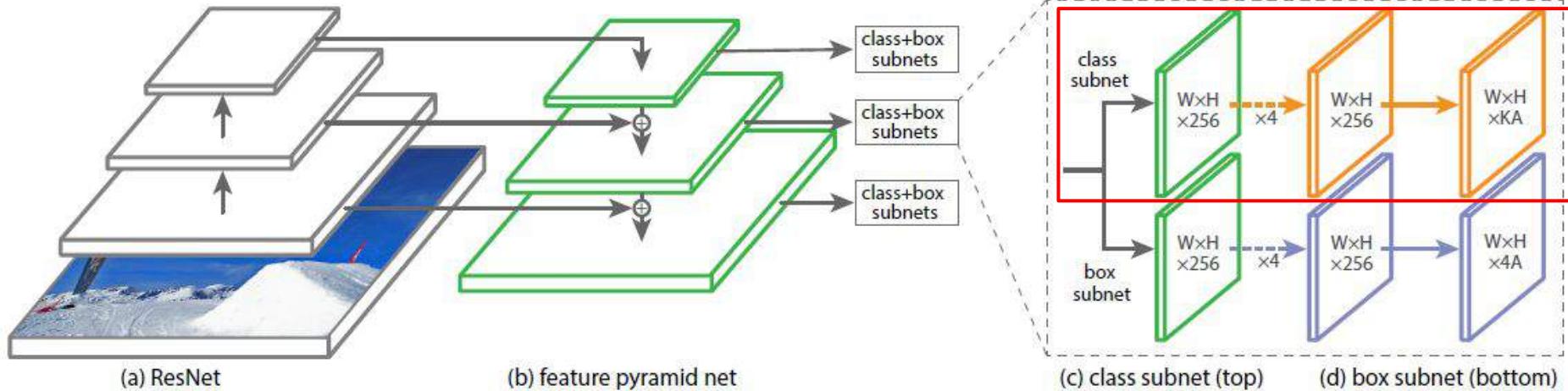


# “Focal Loss for Dense Object Detection”

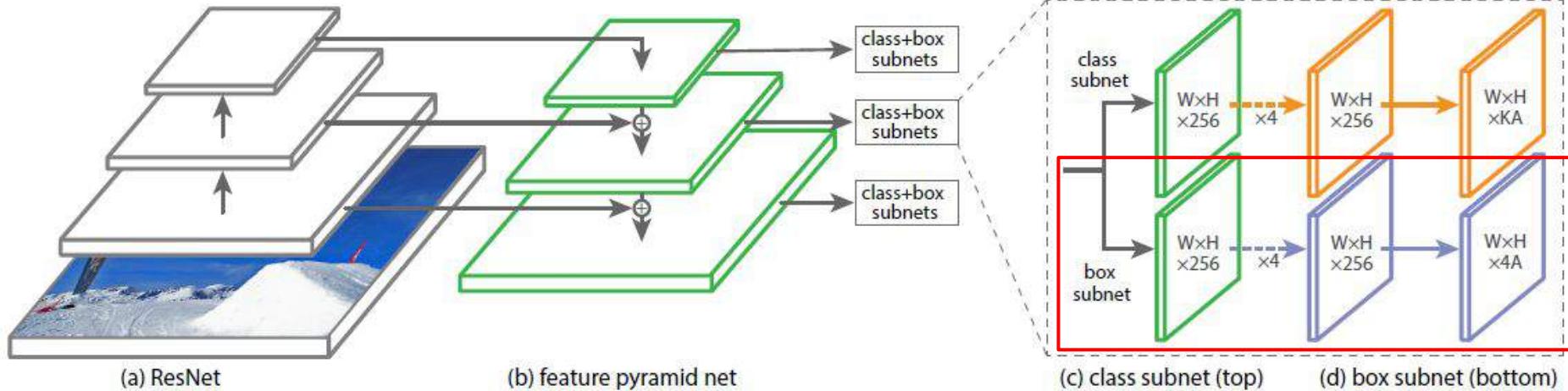
By: Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár

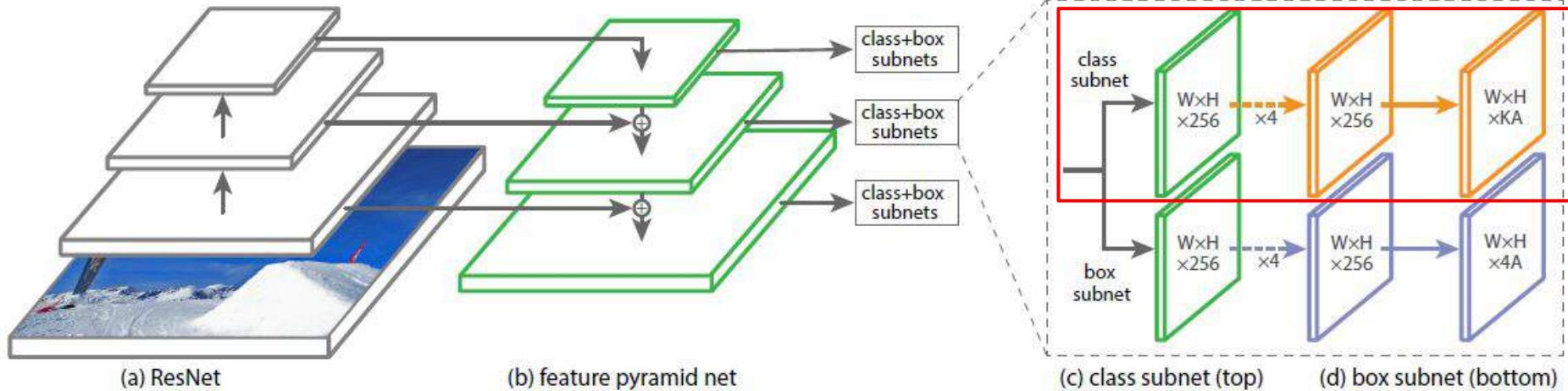
<https://arxiv.org/abs/1708.02002>



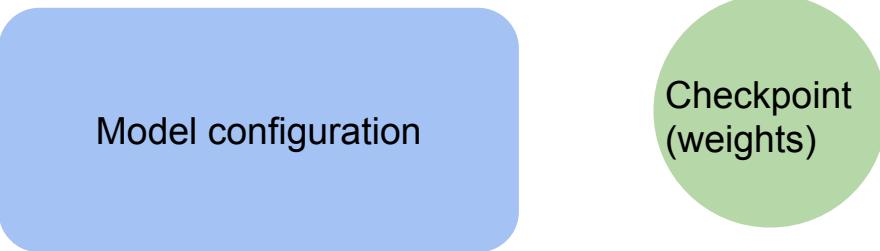


<https://arxiv.org/abs/1708.02002>





Model configuration



Model configuration

Checkpoint  
(weights)

Model configuration

Checkpoint  
(weights)

The screenshot shows a Jupyter Notebook environment. On the left, there's a sidebar titled "Files" with a tree view of directory contents. A red box highlights the "content" folder, which contains a "models" folder. In the main workspace, there are two code cells. The top cell runs a Git command to clone a repository:

```
[1] !git clone --depth 1 https://github.com/tensorflow/models.git
```

The output of this command shows the cloning process into the "models" directory:

```
Cloning into 'models'...
remote: Enumerating objects...
remote: Counting objects...
remote: Compressing objects...
remote: Total 2243 (delta 0)
Receiving objects: 100% (2243/2243)
Resolving deltas: 100% (55/55)
```

The bottom cell runs a "pwd" command to show the current working directory:

```
[2] !pwd
```

The output shows the path "/content":

```
/content
```

```
!wget http://download.tensorflow.org/models/object_detection/tf2/20200711/  
      ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

```
!tar -xf ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

A green circular icon with a thin black border. Inside the circle, the text "Checkpoint (weights)" is written in white, sans-serif font, centered vertically and horizontally.

```
!mv ssd_resnet50_v1_fpn_640x640_coco17_tpu-8/checkpoint
```

```
models/research/object_detection/test_data/
```

```
!wget http://download.tensorflow.org/models/object_detection/tf2/20200711/  
      ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

```
!tar -xf ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

Checkpoint  
(weights)

```
!mv ssd_resnet50_v1_fpn_640x640_coco17_tpu-8/checkpoint  
      models/research/object_detection/test_data/
```

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

```
!wget http://download.tensorflow.org/models/object_detection/tf2/20200711/  
      ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

```
!tar -xf ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

Checkpoint  
(weights)

```
!mv ssd_resnet50_v1_fpn_640x640_coco17_tpu-8/checkpoint  
      models/research/object_detection/test_data/
```

```
!wget http://download.tensorflow.org/models/object_detection/tf2/20200711/  
      ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

```
!tar -xf ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

Checkpoint  
(weights)

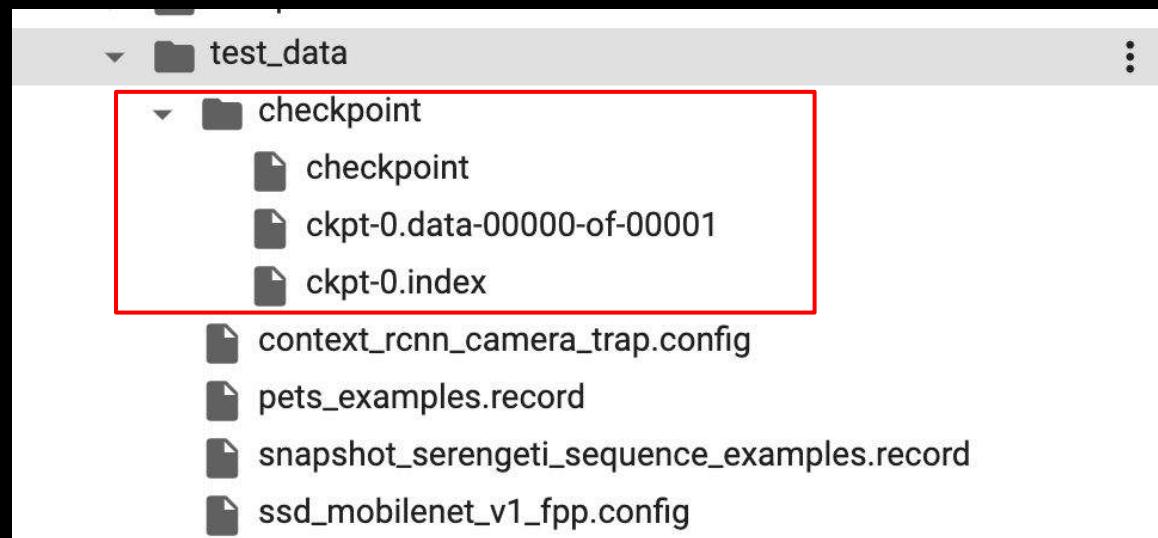
```
!mv ssd_resnet50_v1_fpn_640x640_coco17_tpu-8/checkpoint  
      models/research/object_detection/test_data/
```

```
!wget http://download.tensorflow.org/models/object_detection/tf2/20200711/  
      ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

```
!tar -xf ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
```

Checkpoint  
(weights)

```
!mv ssd_resnet50_v1_fpn_640x640_coco17_tpu-8/checkpoint  
      models/research/object_detection/test_data/
```



```
from object_detection.utils import label_map_util  
from object_detection.utils import config_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import colab_utils  
from object_detection.builders import model_builder
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import config_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import colab_utils  
from object_detection.builders import model_builder
```

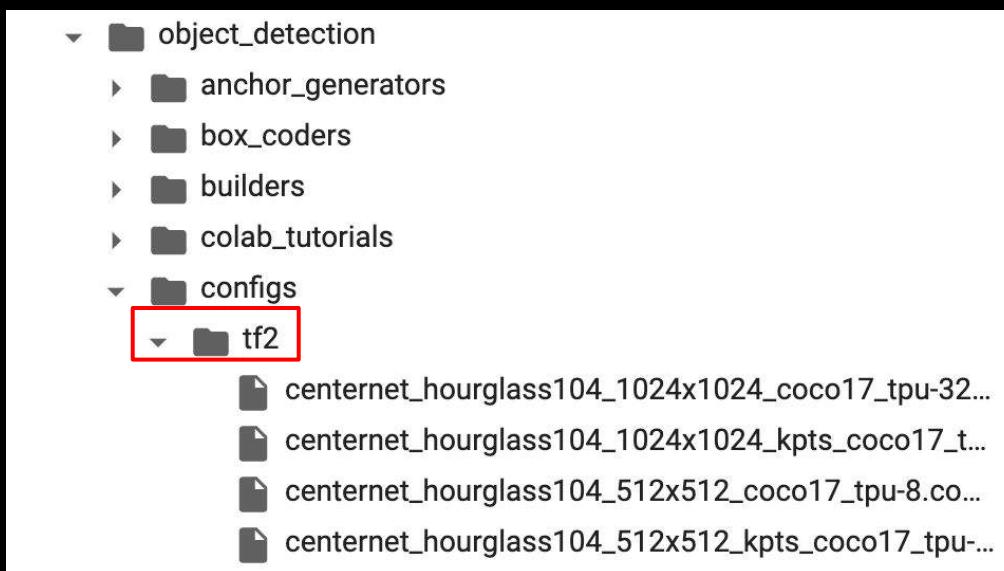
```
from object_detection.utils import label_map_util
from object_detection.utils import config_util
from object_detection.utils import visualization_utils as viz_utils
from object_detection.utils import colab_utils
from object_detection.builders import model_builder
```

```
from object_detection.utils import label_map_util  
from object_detection.utils import config_util  
from object_detection.utils import visualization_utils as viz_utils  
from object_detection.utils import colab_utils  
from object_detection.builders import model_builder
```

```
pipeline_config = 'models/research/object_detection/configs/tf2/  
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config'  
  
checkpoint_path = 'models/research/object_detection/  
test_data/checkpoint/ckpt-0'
```

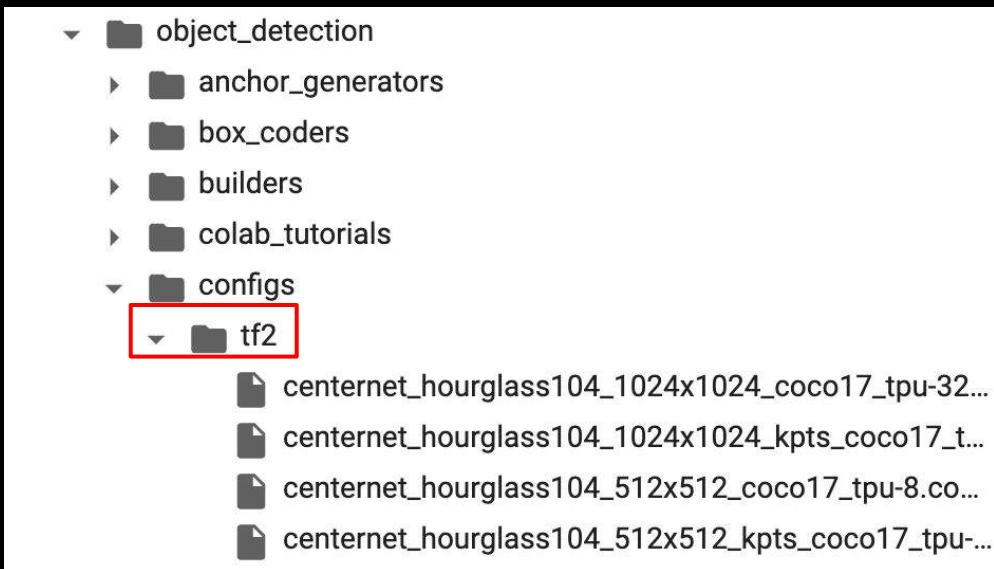
```
pipeline_config = 'models/research/object_detection/configs/tf2/  
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config'
```

```
checkpoint_path = 'models/research/object_detection/  
test_data/checkpoint/ckpt-0'
```



```
pipeline_config = 'models/research/object_detection/configs/tf2/  
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config'
```

```
checkpoint_path = 'models/research/object_detection/  
test_data/checkpoint/ckpt-0'
```



```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X  
1 # SSD with Resnet 50 v1 FPN feature extracto  
2 # loss (a.k.a Retinanet).  
3 # See Lin et al, https://arxiv.org/abs/1708.01178.  
4 # Trained on COCO, initialized from Imagenet  
5 # Train on TPU-8  
6 #  
7 # Achieves 34.3 mAP on COCO17 Val  
8  
9 model {  
10   ssd {  
11     inplace_batchnorm_update: true  
12     freeze_batchnorm: false  
13     num_classes: 90  
14     box_coder {  
15       faster_rcnn_box_coder {  
16         y_scale: 10.0  
17         x_scale: 10.0  
18         height_scale: 5.0  
19         width_scale: 5.0  
20       }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)
```

```
model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True
```

```
detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)
```

```
model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True
```

```
detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X
1 # SSD with Resnet 50 v1 FPN feature extractor.
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01178
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X
1 # SSD with Resnet 50 v1 FPN feature extractor
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01178
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)
```

```
model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True
```

```
detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X
1 # SSD with Resnet 50 v1 FPN feature extractor.
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01102
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config ×
1 # SSD with Resnet 50 v1 FPN feature extractor
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01178
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X
1 # SSD with Resnet 50 v1 FPN feature extractor.
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01102
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X
1 # SSD with Resnet 50 v1 FPN feature extractor
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01178
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)
```

```
model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True
```

```
detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

Model configuration

```
ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config X
1 # SSD with Resnet 50 v1 FPN feature extractor.
2 # loss (a.k.a Retinanet).
3 # See Lin et al, https://arxiv.org/abs/1708.01102
4 # Trained on COCO, initialized from Imagenet
5 # Train on TPU-8
6 #
7 # Achieves 34.3 mAP on COCO17 Val
8
9 model {
10   ssd {
11     inplace_batchnorm_update: true
12     freeze_batchnorm: false
13     num_classes: 90
14     box_coder {
15       faster_rcnn_box_coder {
16         y_scale: 10.0
17         x_scale: 10.0
18         height_scale: 5.0
19         width_scale: 5.0
20     }
```

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

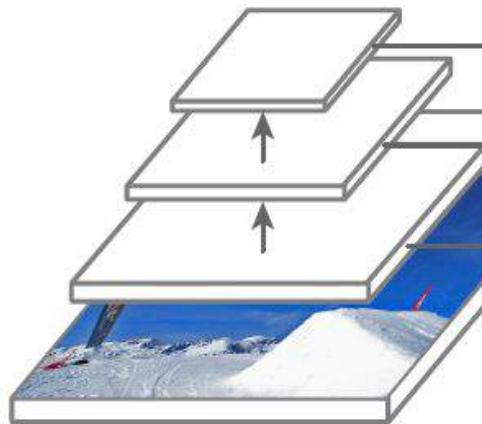
Model configuration

```
configs = config_util.get_configs_from_pipeline_file(pipeline_config)

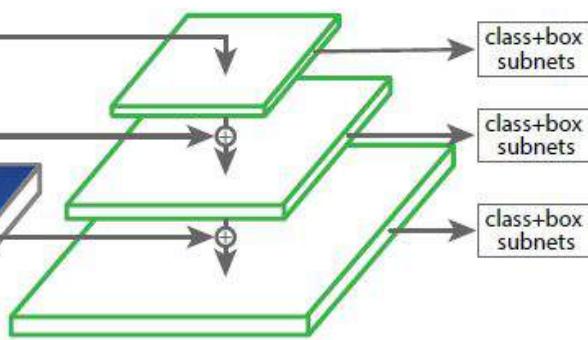
model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
    model_config=model_config, is_training=True)
```

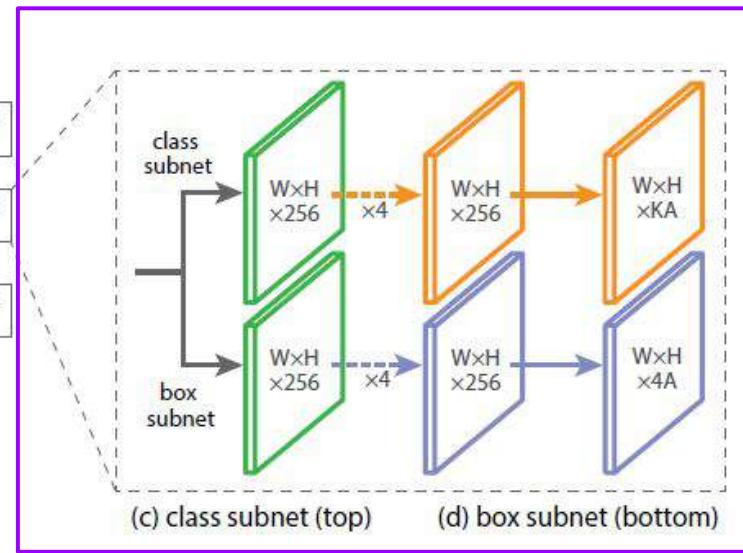
Model configuration



(a) ResNet

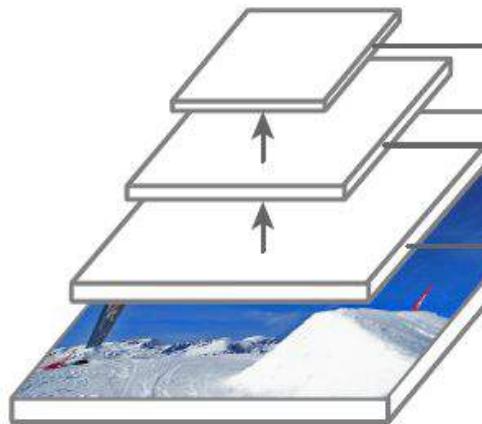


(b) feature pyramid net

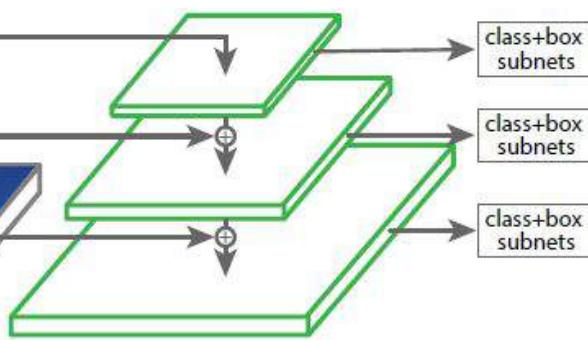


(c) class subnet (top)

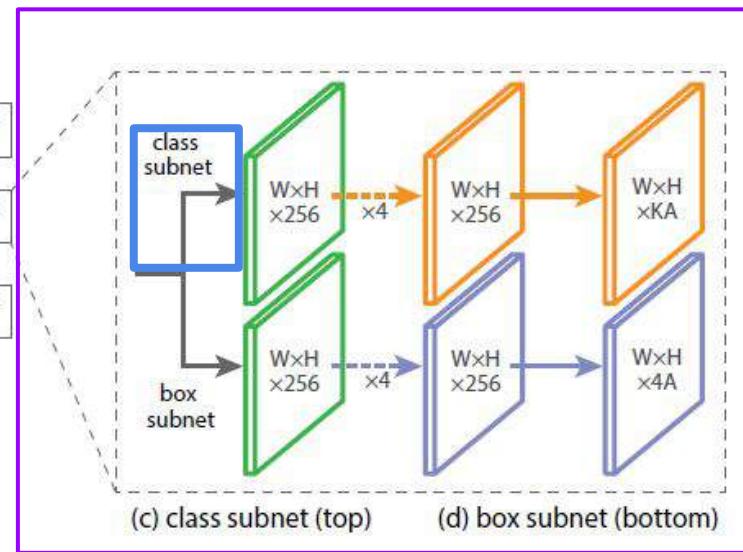
(d) box subnet (bottom)



(a) ResNet

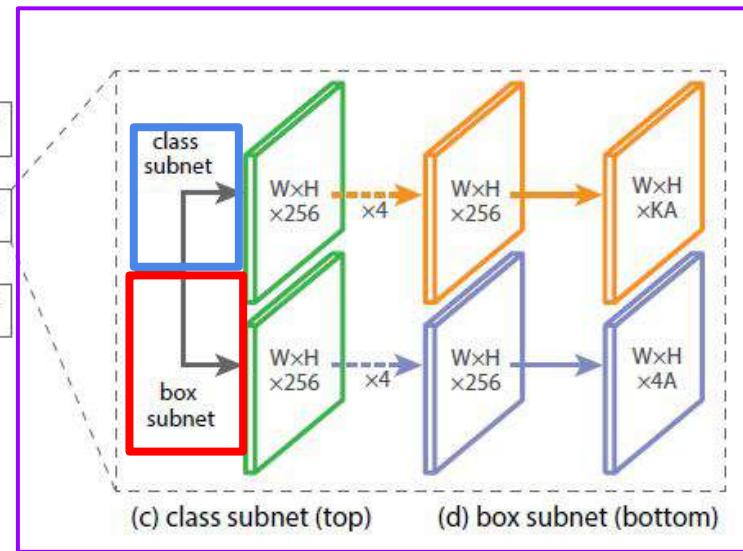
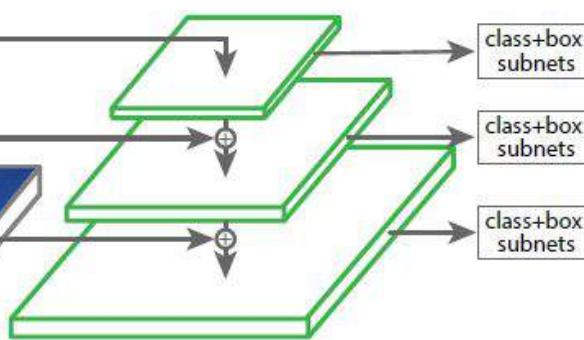
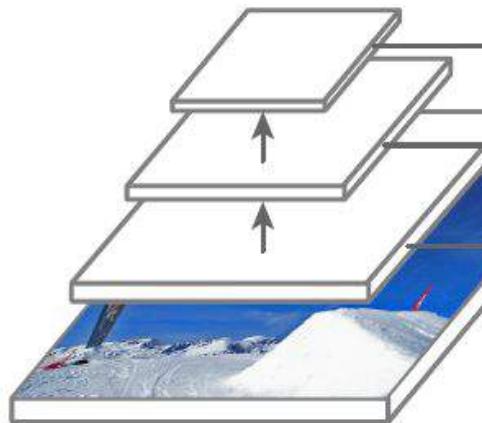


(b) feature pyramid net



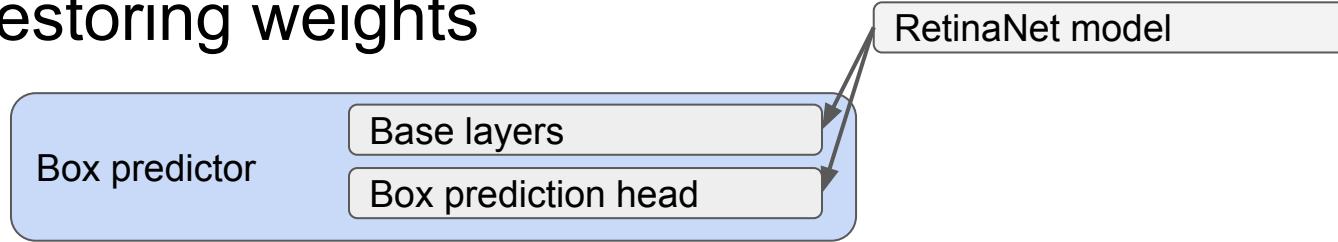
(c) class subnet (top)

(d) box subnet (bottom)

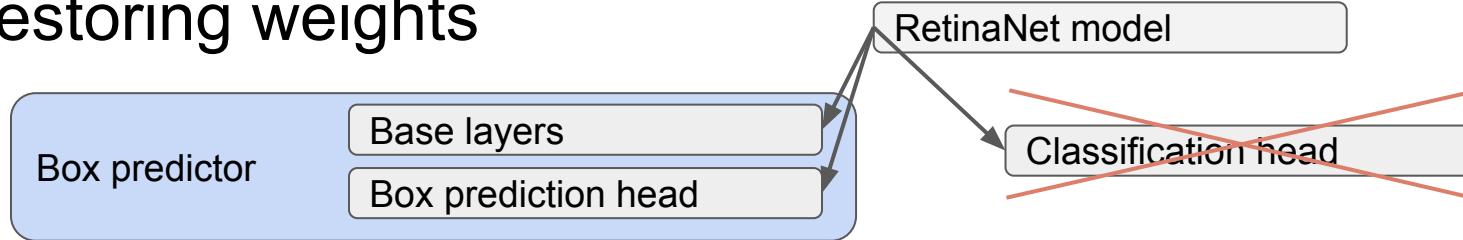


# Restoring weights

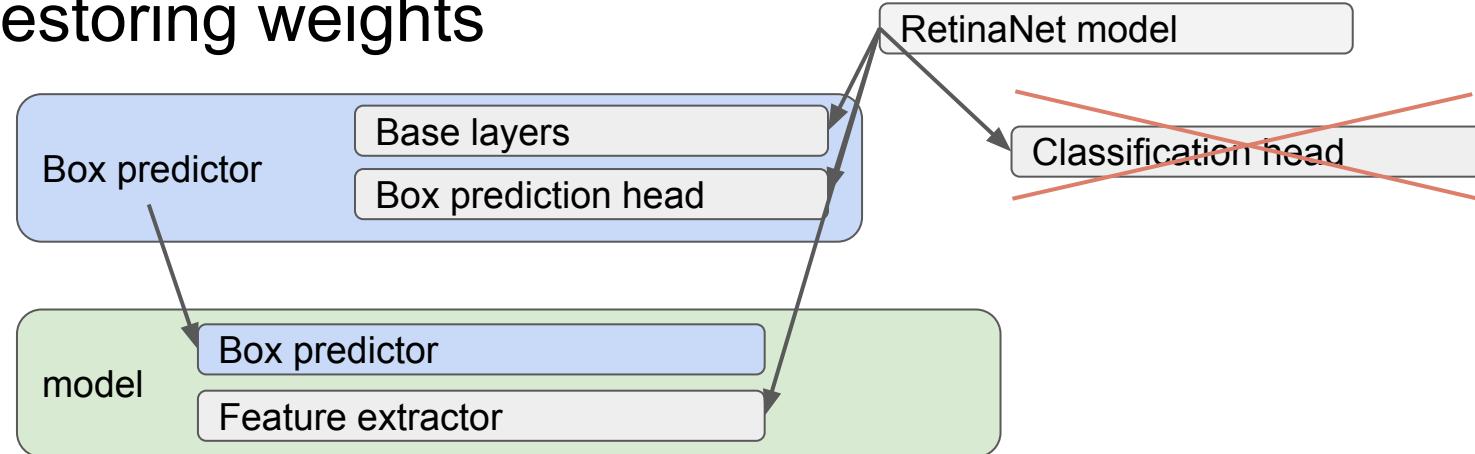
# Restoring weights



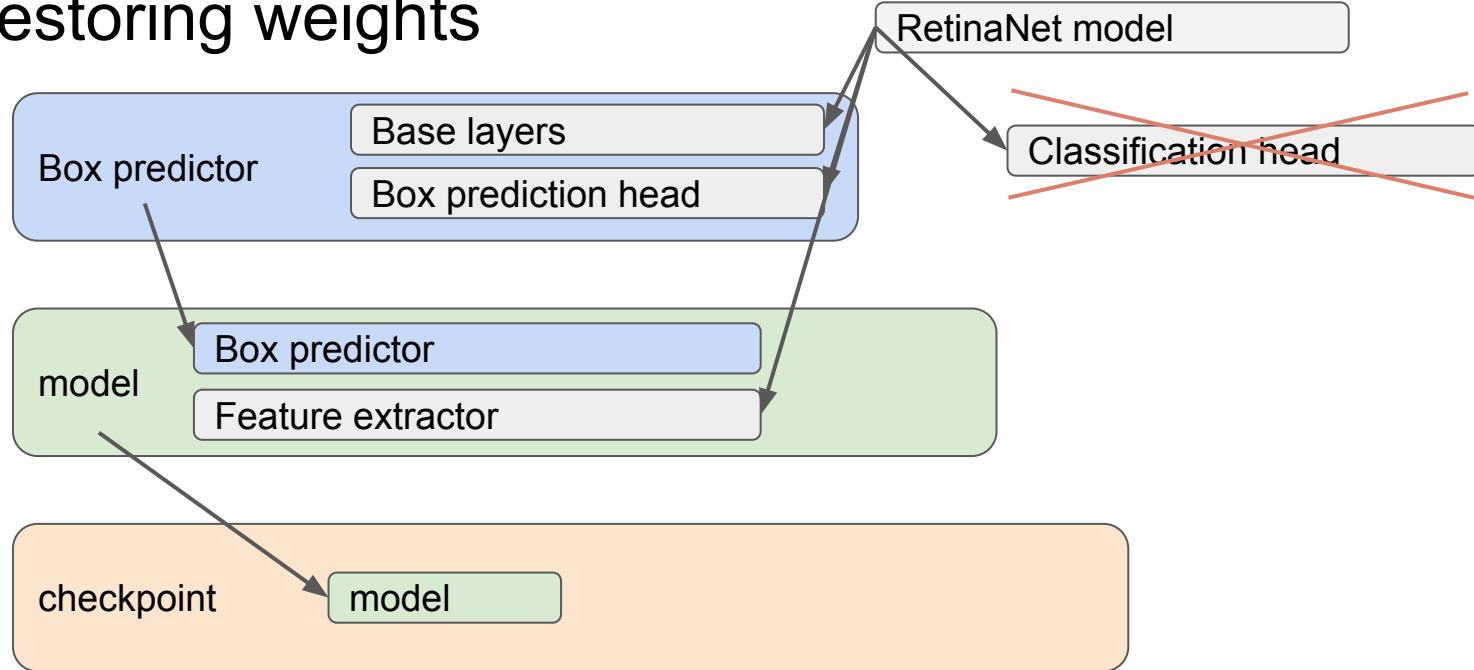
# Restoring weights



# Restoring weights



# Restoring weights



```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)  
  
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)  
  
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```

```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)  
  
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)  
  
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```

Box predictor

```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)  
  
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)  
  
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```

Box predictor

RetinaNet model

Classification head



```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)
```

```
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)
```

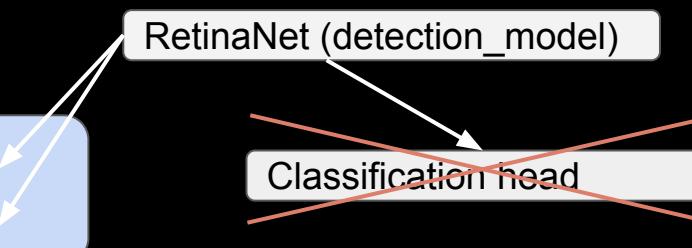
```
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```

Box predictor

Base layers  
Box prediction head

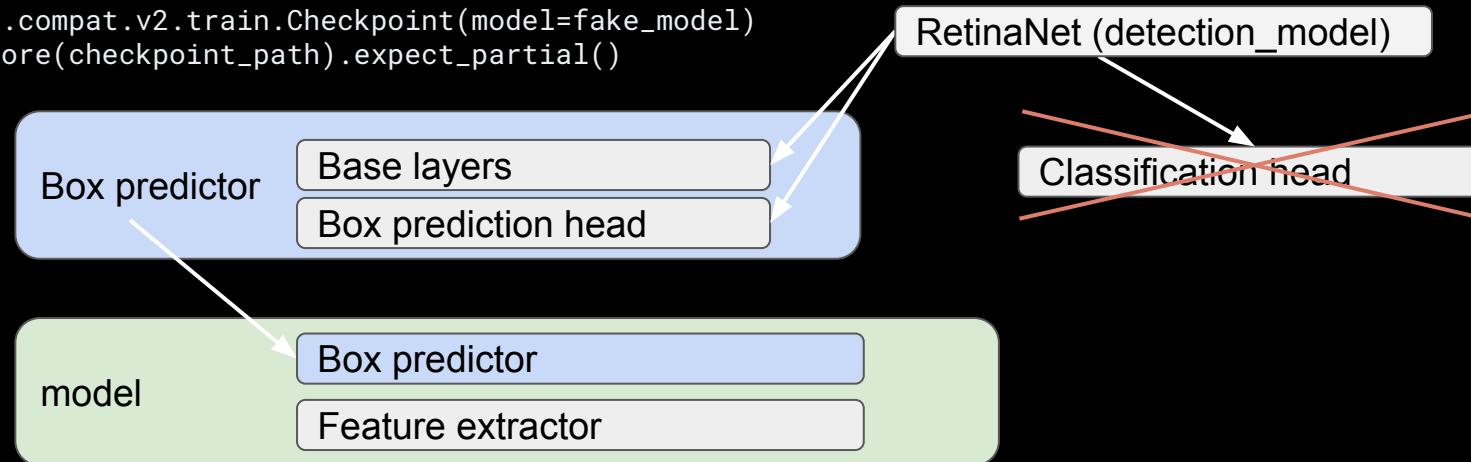
RetinaNet (detection\_model)

Classification head



```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)  
  
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)
```

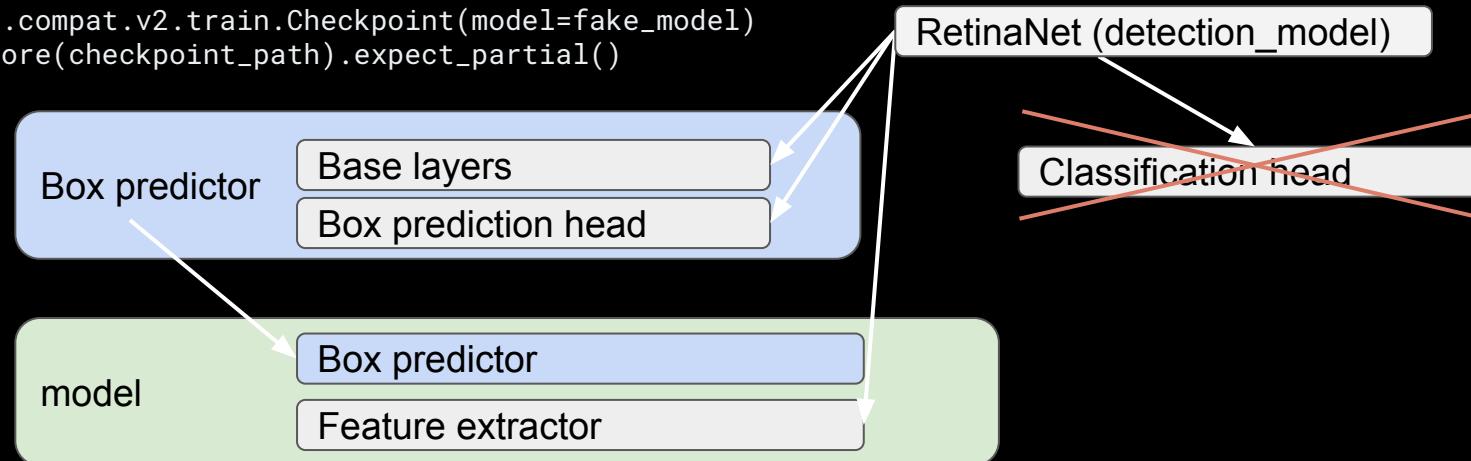
```
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```



```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)
```

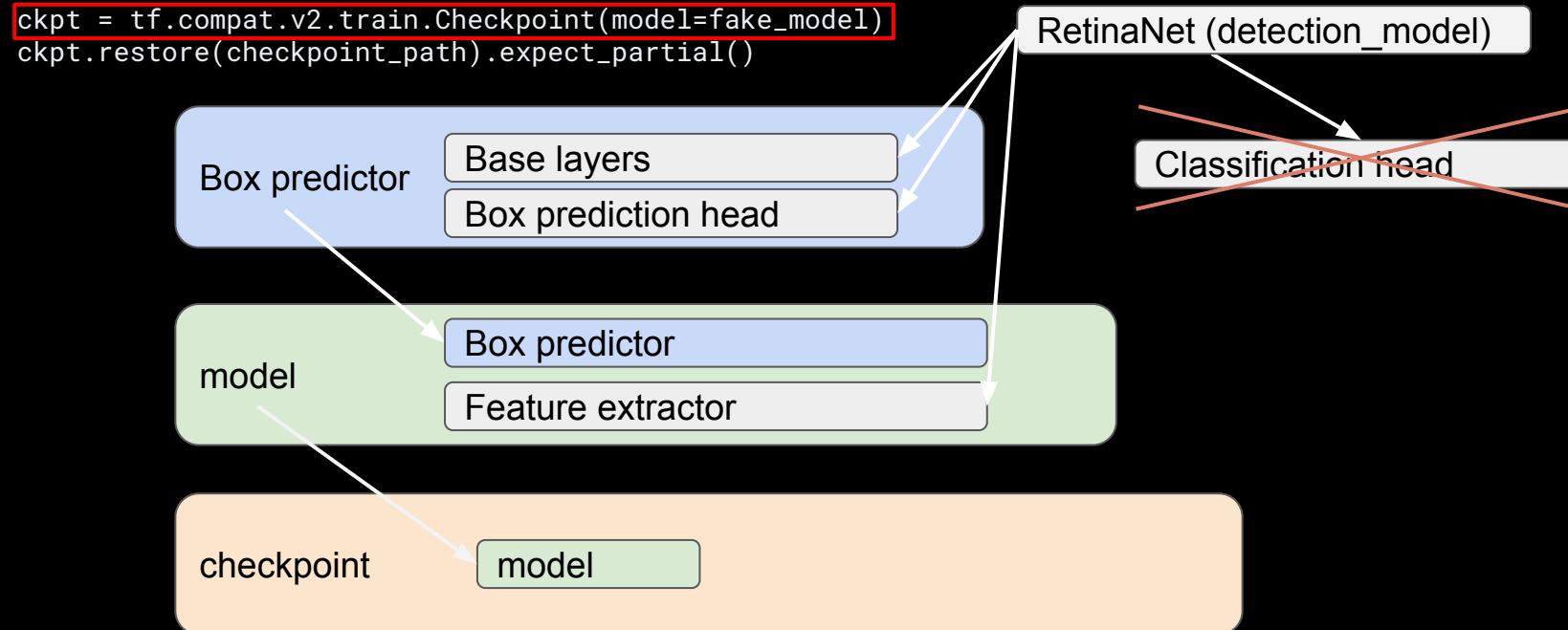
```
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)
```

```
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```



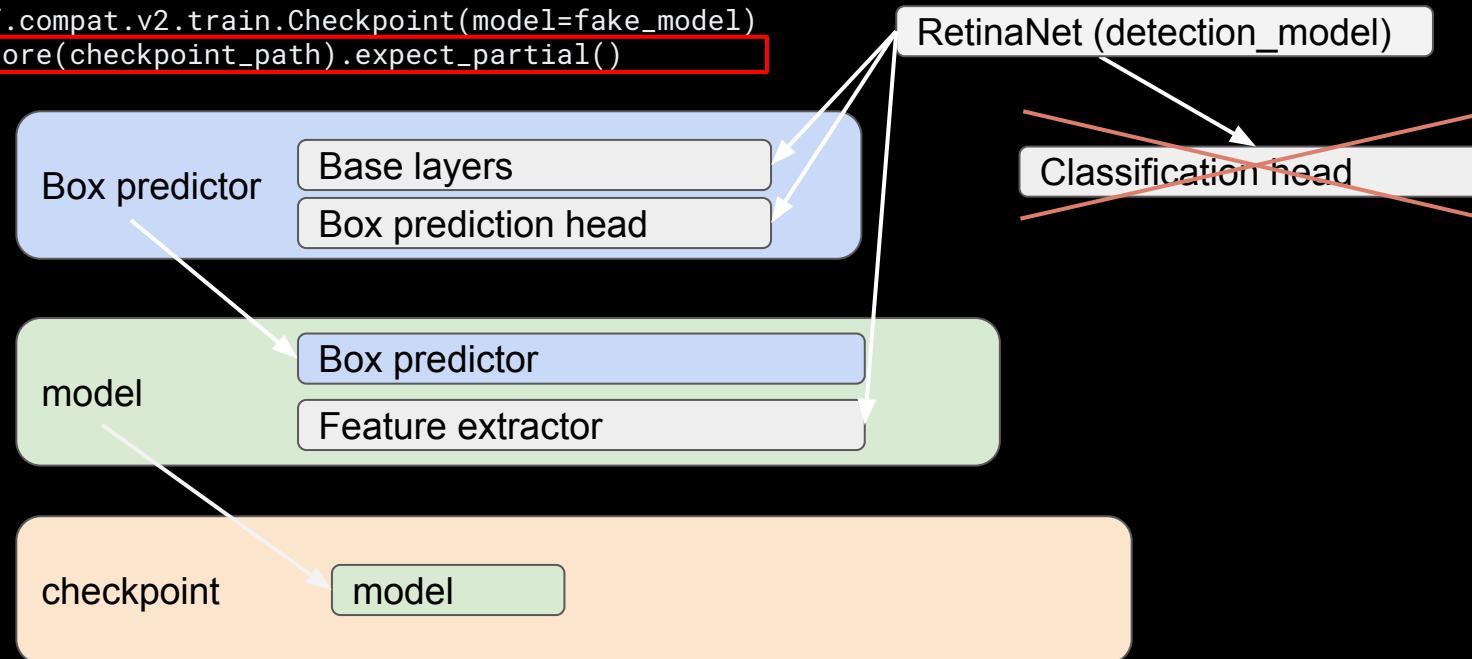
```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)  
  
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)
```

```
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```



```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)  
  
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)
```

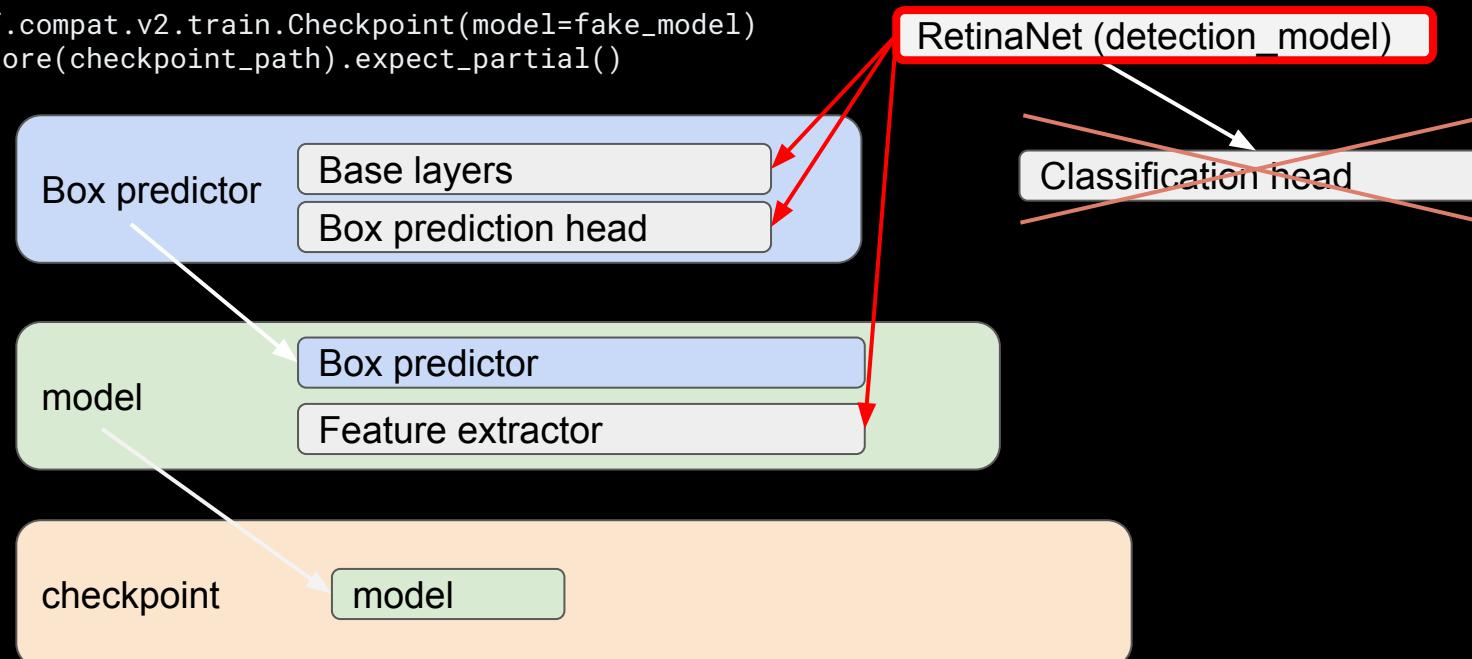
```
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```



```
fake_box_predictor = tf.compat.v2.train.Checkpoint(  
    _base_tower_layers_for_heads=detection_model._box_predictor._base_tower_layers_for_heads,  
    _box_prediction_head=detection_model._box_predictor._box_prediction_head,  
)
```

```
fake_model = tf.compat.v2.train.Checkpoint(  
    _feature_extractor=detection_model._feature_extractor,  
    _box_predictor=fake_box_predictor)
```

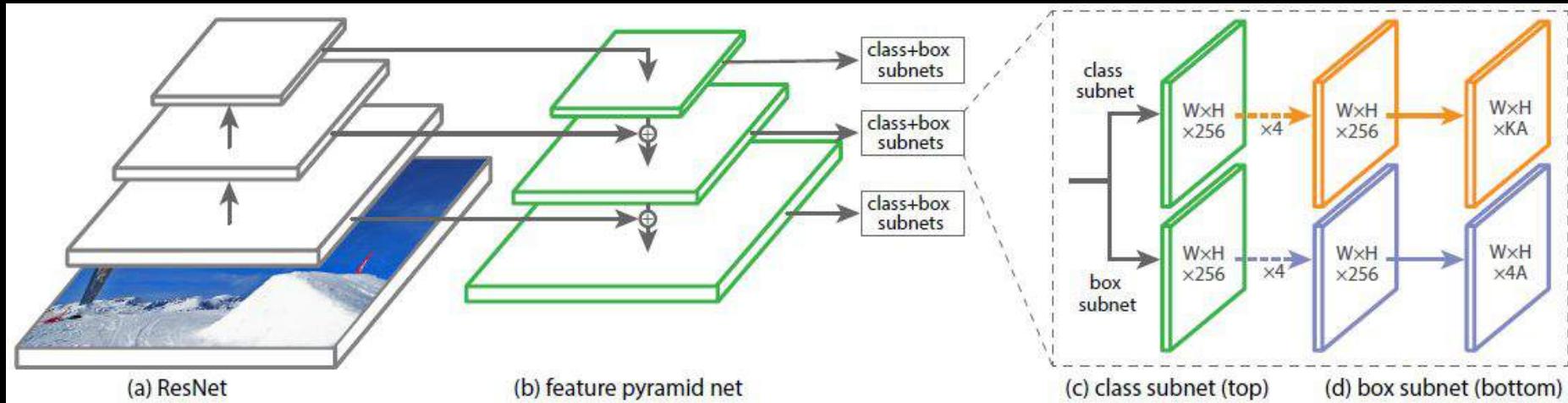
```
ckpt = tf.compat.v2.train.Checkpoint(model=fake_model)  
ckpt.restore(checkpoint_path).expect_partial()
```



```

# Run dummy image through the model so that variables are created
image, shapes = detection_model.preprocess(tf.zeros([1, 640, 640, 3]))
prediction_dict = detection_model.predict(image, shapes)
_ = detection_model.postprocess(prediction_dict, shapes)
print('Weights restored!')

```



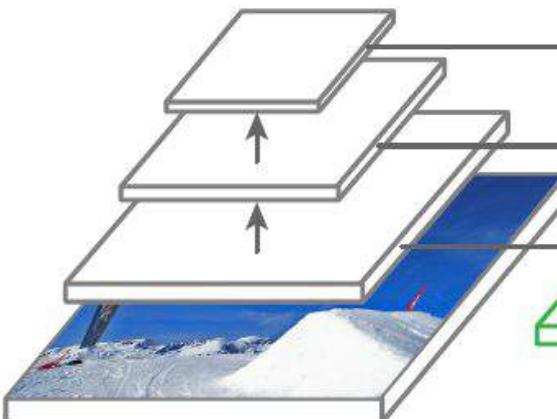
```
# Run model through a dummy image so that variables are created
```

```
image, shapes = detection_model.preprocess(tf.zeros([1, 640, 640, 3]))
```

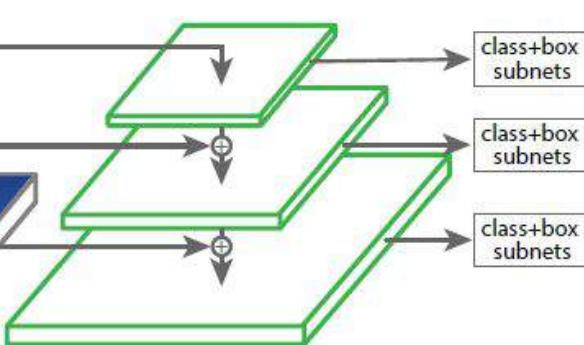
```
prediction_dict = detection_model.predict(image, shapes)
```

```
_ = detection_model.postprocess(prediction_dict, shapes)
```

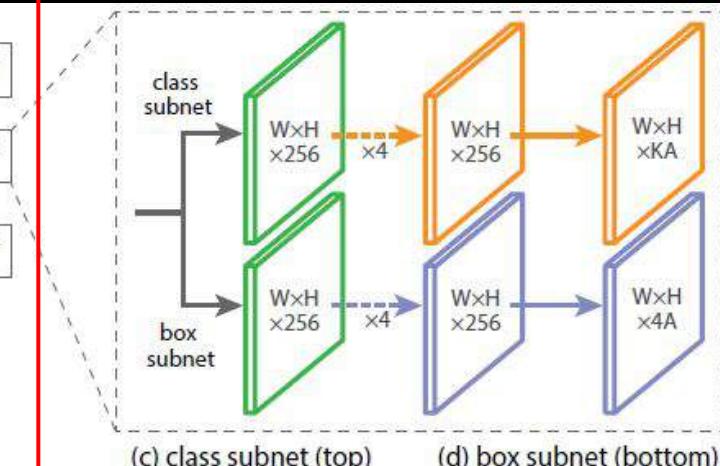
```
print('Weights restored!')
```



(a) ResNet



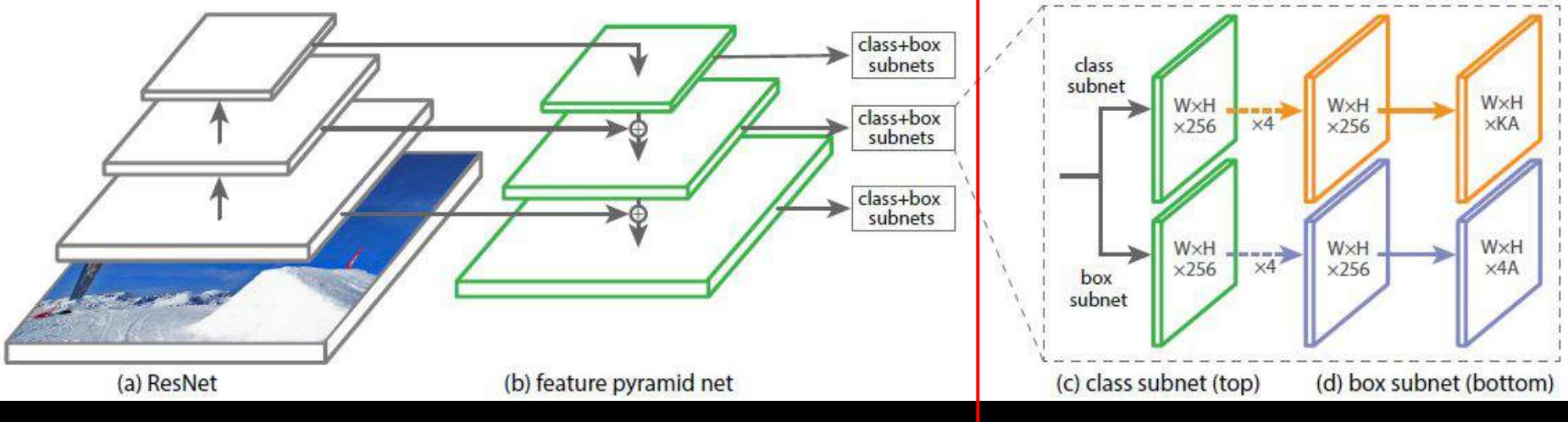
(b) feature pyramid net



(c) class subnet (top)

(d) box subnet (bottom)

```
# Run model through a dummy image so that variables are created
image, shapes = detection_model.preprocess(tf.zeros([1, 640, 640, 3]))
prediction_dict = detection_model.predict(image, shapes)
_ = detection_model.postprocess(prediction_dict, shapes)
print('Weights restored!')
```

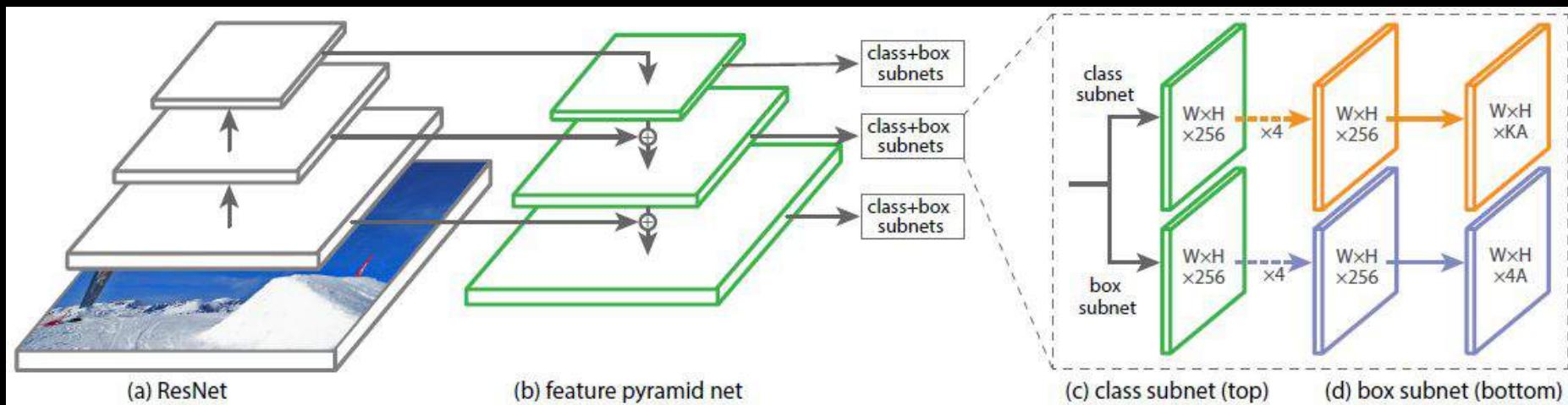


```

# Run model through a dummy image so that variables are created
image, shapes = detection_model.preprocess(tf.zeros([1, 640, 640, 3]))
prediction_dict = detection_model.predict(image, shapes)
_ = detection_model.postprocess(prediction_dict, shapes)

print('Weights restored!')

```



```
gt_boxes = []
colab_utils.annotate(train_images_np, box_storage_pointer=gt_boxes)
```

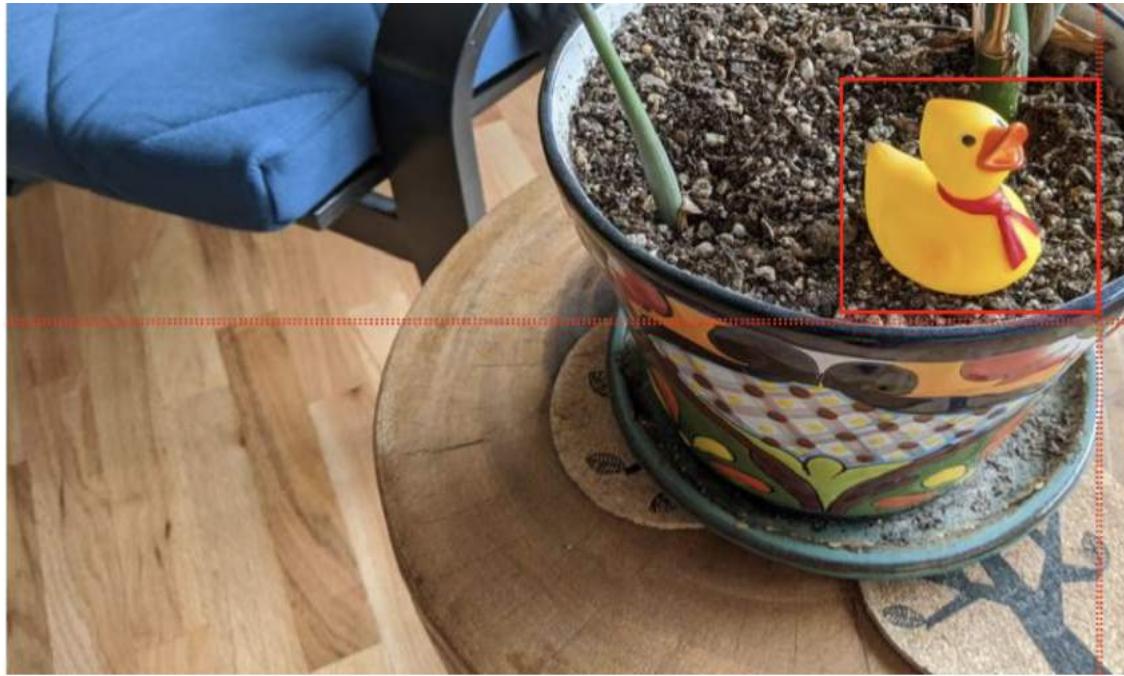
```
gt_boxes = []
colab_utils.annotate(train_images_np, box_storage_pointer=gt_boxes)
```



[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

[submit](#)

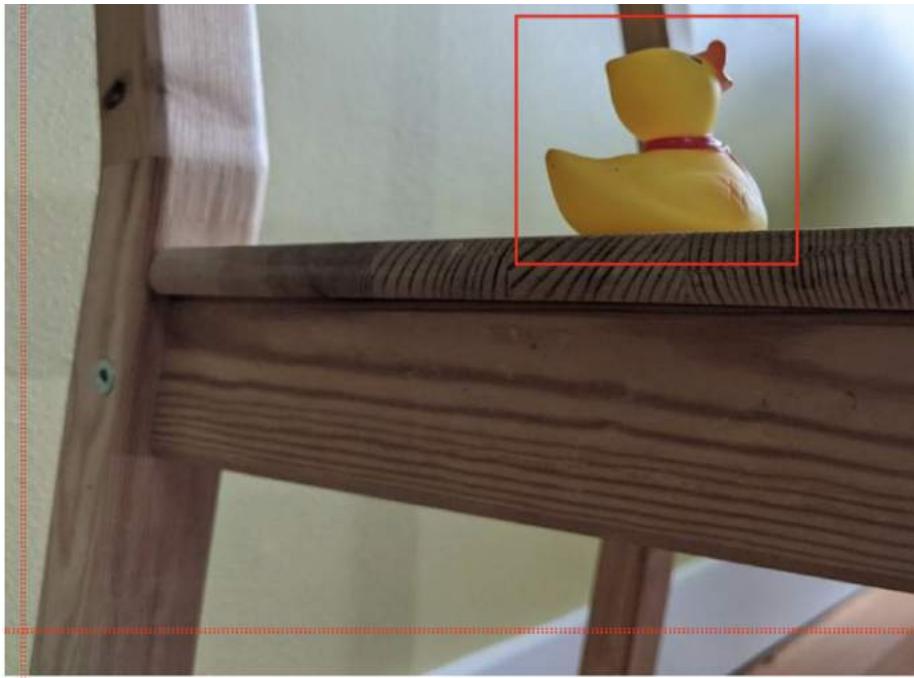
```
gt_boxes = []
colab_utils.annotate(train_images_np, box_storage_pointer=gt_boxes)
```



[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

[submit](#)

```
gt_boxes = []
colab_utils.annotate(train_images_np, box_storage_pointer=gt_boxes)
```

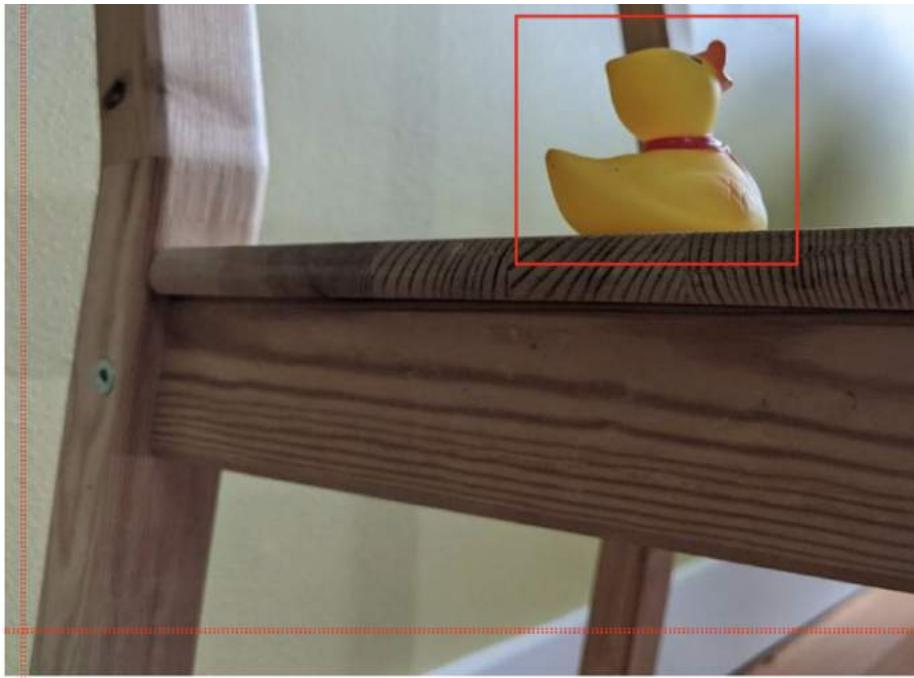


All images completed!!

[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

[submit](#)

```
gt_boxes = []
colab_utils.annotate(train_images_np, box_storage_pointer=gt_boxes)
```

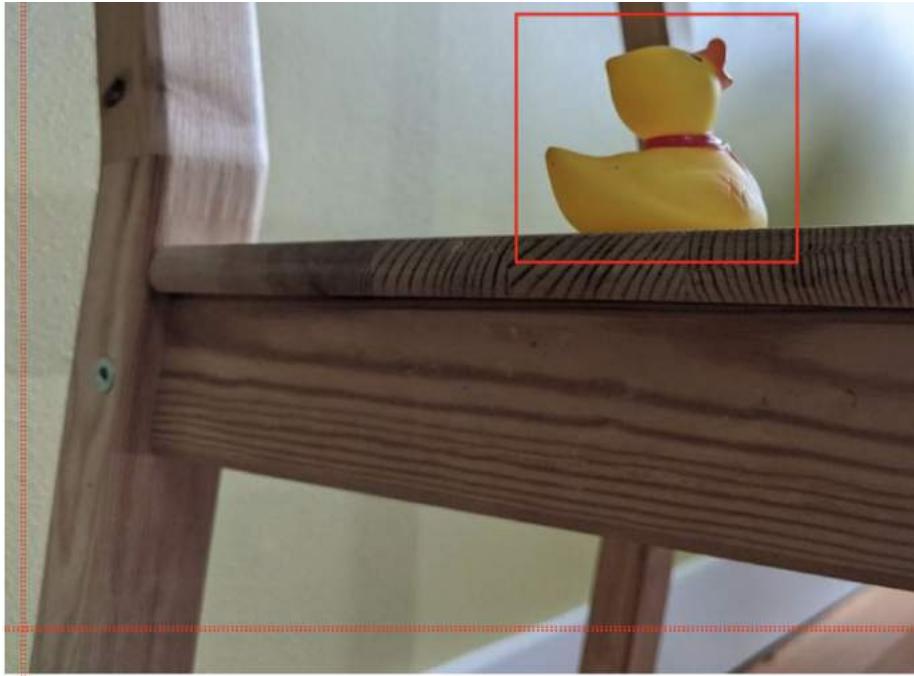


All images completed!!

[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

[submit](#)

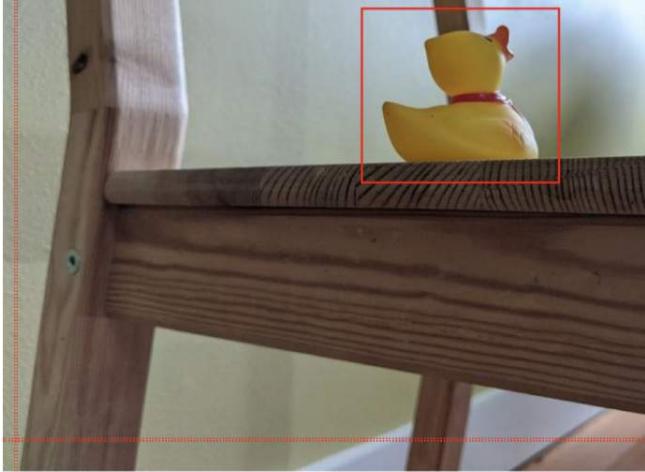
```
gt_boxes = []
colab_utils.annotate(train_images_np, box_storage_pointer=gt_boxes)
```



All images completed!!

[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

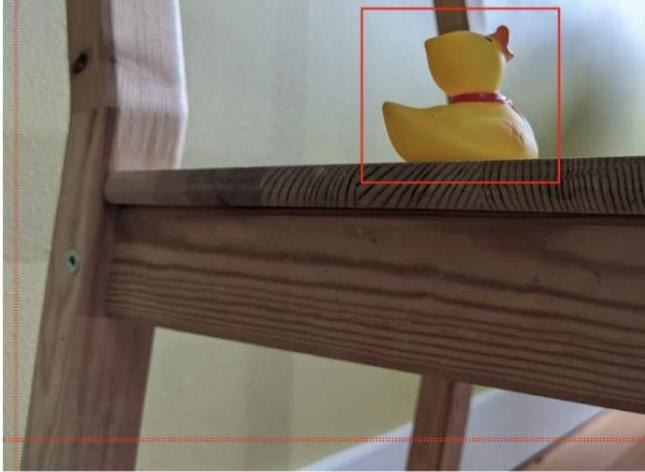
[submit](#)



All images completed!!

[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

```
gt_boxes = [
    np.array([[0.436, 0.591, 0.629, 0.712]], dtype=np.float32),
    np.array([[0.539, 0.583, 0.73, 0.71]], dtype=np.float32),
    np.array([[0.464, 0.414, 0.626, 0.548]], dtype=np.float32),
    np.array([[0.313, 0.308, 0.648, 0.526]], dtype=np.float32),
    np.array([[0.256, 0.444, 0.484, 0.629]], dtype=np.float32)
]
```



All images completed!!

[prev image](#) [next image](#) [undo bbox](#) [delete all](#)

```
gt_boxes = [  
    np.array([[0.436, 0.591, 0.629, 0.712]], dtype=np.float32),  
    np.array([[0.539, 0.583, 0.73, 0.71]], dtype=np.float32),  
    np.array([[0.464, 0.414, 0.626, 0.548]], dtype=np.float32),  
    np.array([[0.313, 0.308, 0.648, 0.526]], dtype=np.float32),  
    np.array([[0.256, 0.444, 0.484, 0.629]], dtype=np.float32)  
]
```

# Prepare data for training

Classes are said to be class associations [not associations], because in a single class in this codebase, through it would be strongly forward-referenced by another class or multiple classes]. The class increased everything by the former that the training frequency increased (e.g., everything concerned by formerly unassociated words had a higher frequency than others).

Let's just visualize the rubber duckies as a sanity check.

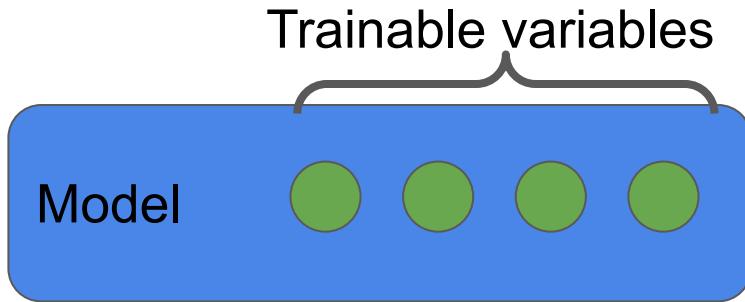


## • See the colab later!

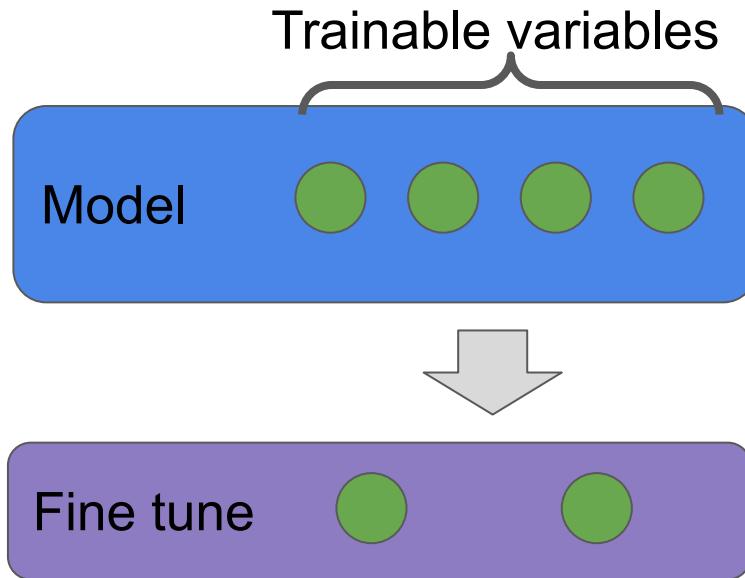


# Custom Training Loop

# Custom Training Loop

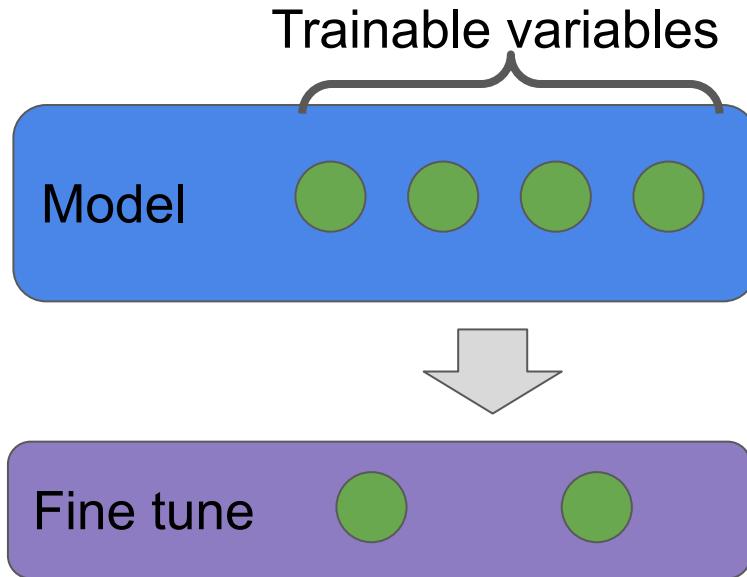


# Custom Training Loop

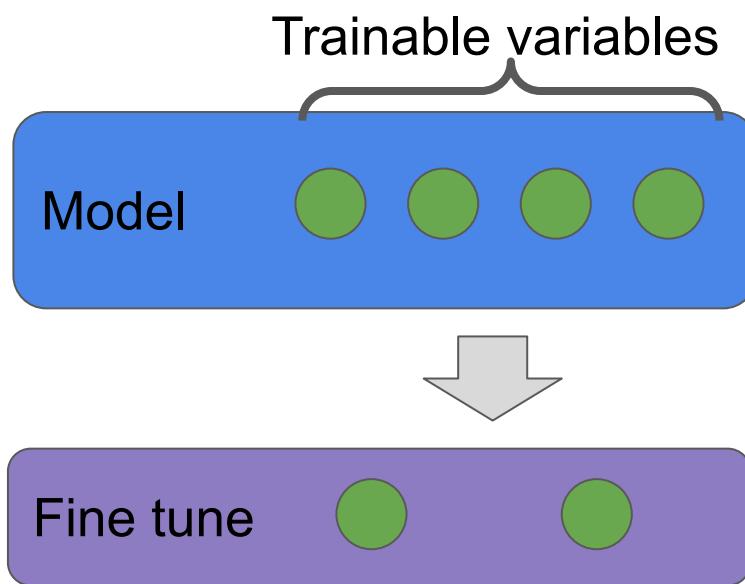


# Custom Training Loop

For all training images:



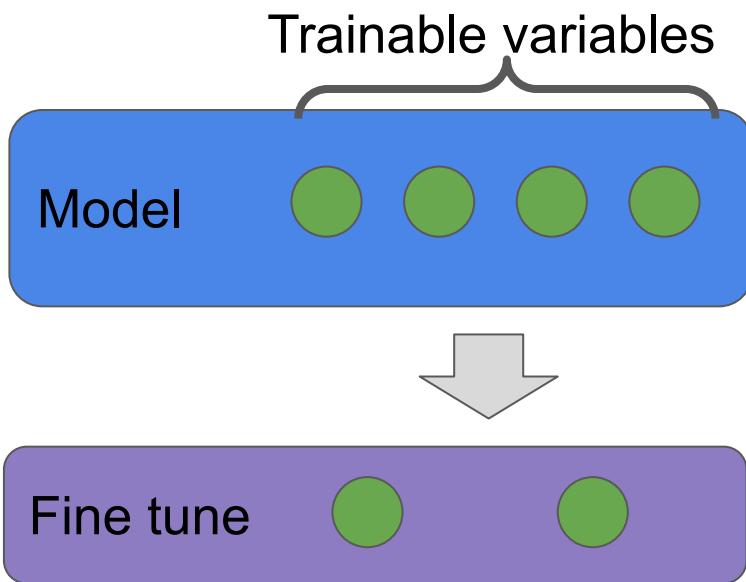
# Custom Training Loop



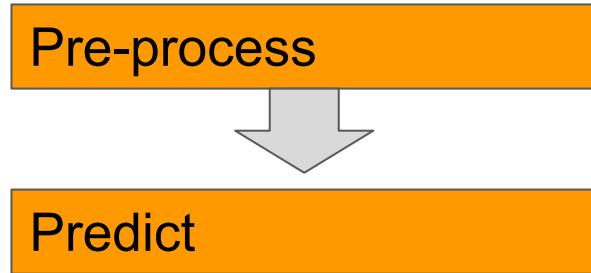
For all training images:

Pre-process

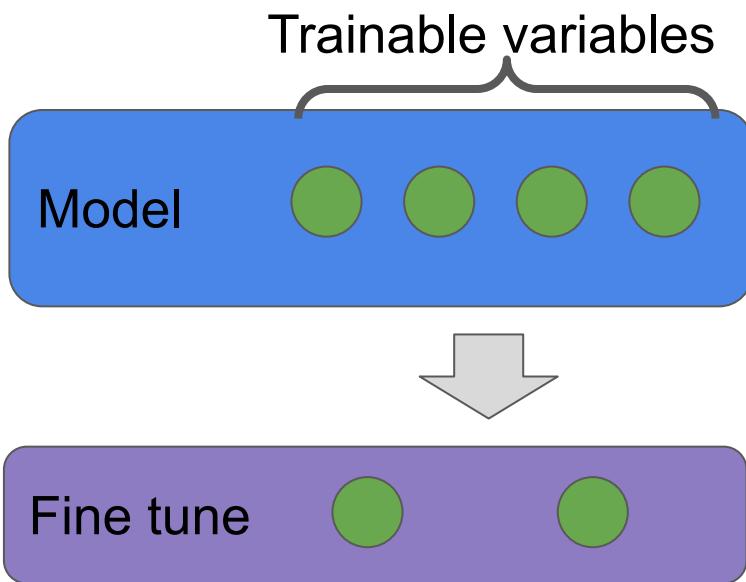
# Custom Training Loop



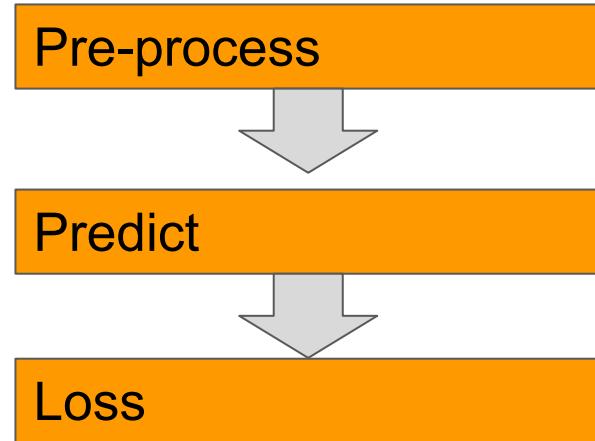
For all training images:



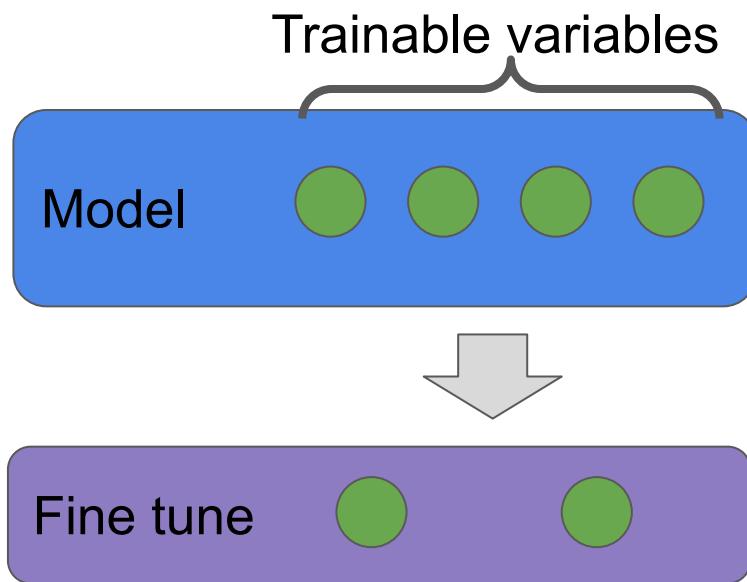
# Custom Training Loop



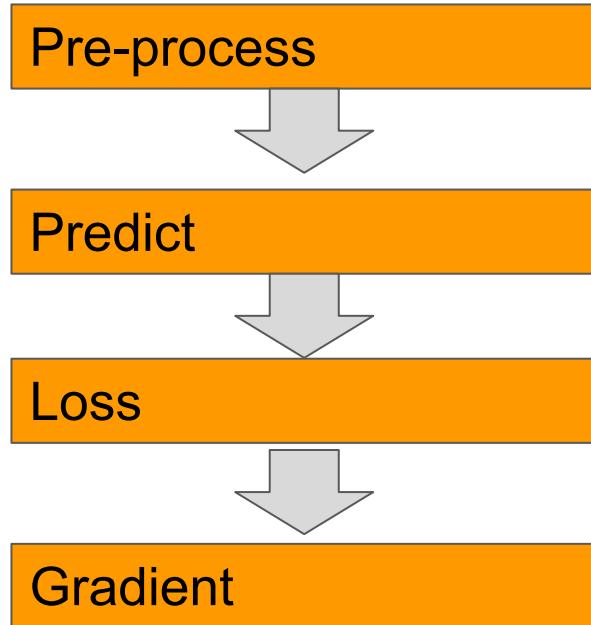
For all training images:



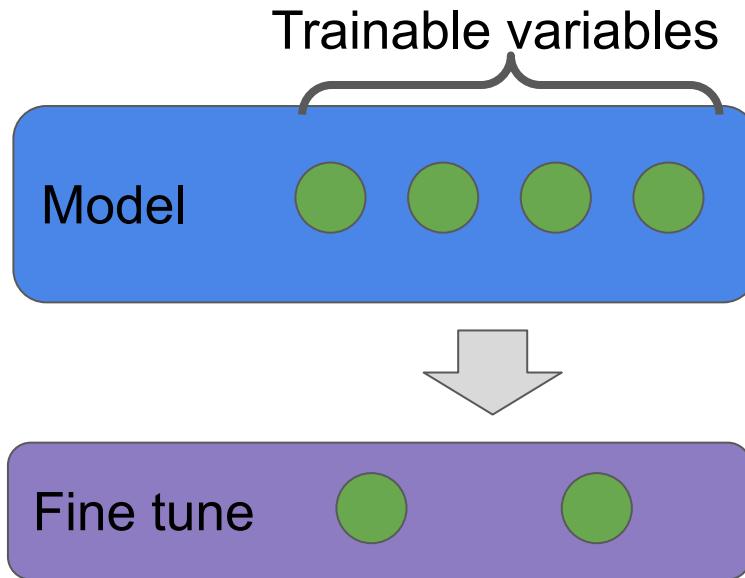
# Custom Training Loop



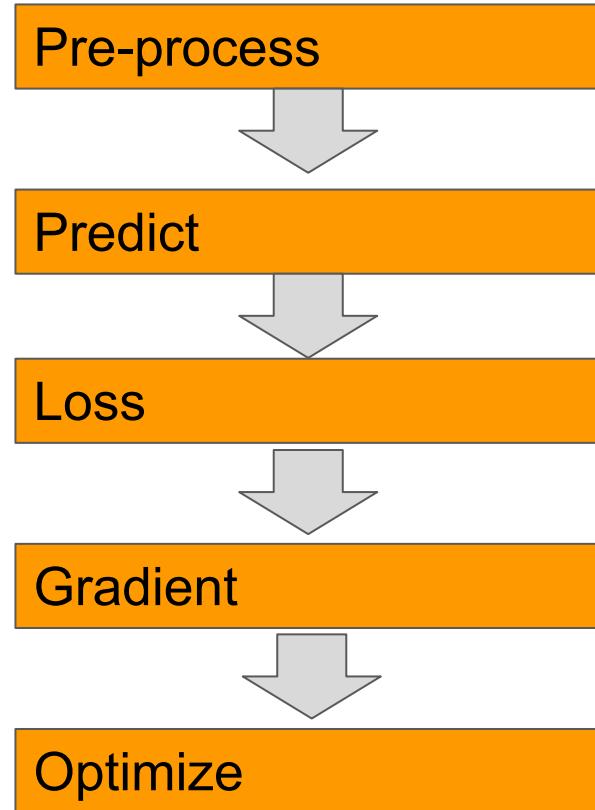
For all training images:



# Custom Training Loop



For all training images:



```
for var in detection_model.trainable_variables:  
    print(var.name)
```

```
for var in detection_model.trainable_variables:  
    print(var.name)
```

```
for var in detection_model.trainable_variables:  
    print(var.name)
```

```
WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalBoxHead/BoxPredictor/kernel:0  
WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalBoxHead/BoxPredictor/bias:0  
WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalClassHead/ClassPredictor/kernel:0  
WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalClassHead/ClassPredictor/bias:0  
WeightSharedConvolutionalBoxPredictor/BoxPredictionTower/conv2d_0/kernel:0  
WeightSharedConvolutionalBoxPredictor/BoxPredictionTower/conv2d_0/BatchNorm/feature_0/gamma:0  
WeightSharedConvolutionalBoxPredictor/BoxPredictionTower/conv2d_0/BatchNorm/feature_0/beta:0  
WeightSharedConvolutionalBoxPredictor/BoxPredictionTower/conv2d_1/kernel:0
```

```
prefixes_to_train = [  
    'WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalBoxHead',  
    'WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalClassHead' ]  
  
for var in trainable_variables:  
    if any([var.name.startswith(prefix) for prefix in prefixes_to_train]):  
        to_fine_tune.append(var)
```

```
prefixes_to_train = [  
    'WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalBoxHead',  
    'WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalClassHead' ]
```

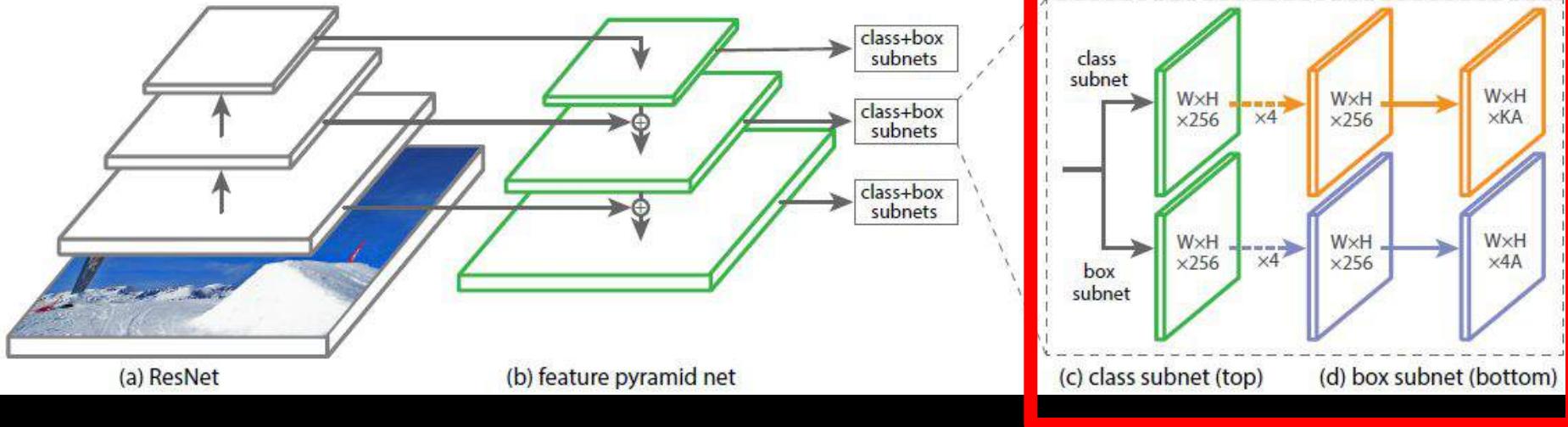
```
for var in trainable_variables:  
    if any([var.name.startswith(prefix) for prefix in prefixes_to_train]):  
        to_fine_tune.append(var)
```

```

prefixes_to_train = [
    'WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalBoxHead',
    'WeightSharedConvolutionalBoxPredictor/WeightSharedConvolutionalClassHead']

for var in trainable_variables:
    if any([var.name.startswith(prefix) for prefix in prefixes_to_train]):
        to_fine_tune.append(var)

```



```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
    prediction_dict = model.predict(preprocessed_images, shapes)  
    losses_dict = model.loss(prediction_dict, shapes)  
    total_loss = losses_dict['Loss/localization_loss'] +  
                 losses_dict['Loss/classification_loss']  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
return total_loss
```

```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
  
    prediction_dict = model.predict(preprocessed_images, shapes)  
    losses_dict = model.loss(prediction_dict, shapes)  
    total_loss = losses_dict['Loss/localization_loss'] +  
                 losses_dict['Loss/classification_loss']  
  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
  
return total_loss
```

```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
  
    prediction_dict = model.predict(preprocessed_images, shapes)  
  
    losses_dict = model.loss(prediction_dict, shapes)  
    total_loss = losses_dict['Loss/localization_loss'] +  
                 losses_dict['Loss/classification_loss']  
  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
  
return total_loss
```

```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
    prediction_dict = model.predict(preprocessed_images, shapes)  
    losses_dict = model.loss(prediction_dict, shapes)  
    total_loss = losses_dict['Loss/localization_loss'] +  
                 losses_dict['Loss/classification_loss']  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
return total_loss
```

```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
    prediction_dict = model.predict(preprocessed_images, shapes)  
    losses_dict = model.loss(prediction_dict, shapes)  
    total_loss = losses_dict['Loss/localization_loss'] +  
                losses_dict['Loss/classification_loss']  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
    return total_loss
```

```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
  
    prediction_dict = model.predict(preprocessed_images, shapes)  
    losses_dict = model.loss(prediction_dict, shapes)  
    total_loss = losses_dict['Loss/localization_loss'] +  
                losses_dict['Loss/classification_loss']  
  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
  
return total_loss
```

```
with tf.GradientTape() as tape:  
    preprocessed_images = tf.concat(  
        [detection_model.preprocess(image_tensor)[0]  
         for image_tensor in image_tensors], axis=0)  
  
    prediction_dict = model.predict(preprocessed_images, shapes)  
    losses_dict = model.loss(prediction_dict, shapes)  
  
    total_loss = losses_dict['Loss/localization_loss'] +  
                losses_dict['Loss/classification_loss']  
  
    gradients = tape.gradient(total_loss, vars_to_fine_tune)  
    optimizer.apply_gradients(zip(gradients, vars_to_fine_tune))  
  
return total_loss
```

Start fine-tuning!

epoch 0 of 100, loss=1.1637363

epoch 10 of 100, loss=1.2570567

epoch 20 of 100, loss=0.054274812

epoch 30 of 100, loss=0.015053727

epoch 40 of 100, loss=0.0052895746

epoch 50 of 100, loss=0.014283805

epoch 60 of 100, loss=0.003811138

epoch 70 of 100, loss=0.0035868755

epoch 80 of 100, loss=0.003034658

epoch 90 of 100, loss=0.002482821

Done fine-tuning!

```
input_tensor = tf.convert_to_tensor(test_images_np[i], dtype=tf.float32)  
...  
preprocessed_image, shapes = detection_model.preprocess(input_tensor)  
prediction_dict = detection_model.predict(preprocessed_image, shapes)  
return detection_model.postprocess(prediction_dict, shapes)
```

```
input_tensor = tf.convert_to_tensor(test_images_np[i], dtype=tf.float32)
...
preprocessed_image, shapes = detection_model.preprocess(input_tensor)
prediction_dict = detection_model.predict(preprocessed_image, shapes)
return detection_model.postprocess(prediction_dict, shapes)
```

```
input_tensor = tf.convert_to_tensor(test_images_np[i], dtype=tf.float32)  
...  
preprocessed_image, shapes = detection_model.preprocess(input_tensor)  
prediction_dict = detection_model.predict(preprocessed_image, shapes)  
return detection_model.postprocess(prediction_dict, shapes)
```

```
input_tensor = tf.convert_to_tensor(test_images_np[i], dtype=tf.float32)  
...  
preprocessed_image, shapes = detection_model.preprocess(input_tensor)  
prediction_dict = detection_model.predict(preprocessed_image, shapes)  
return detection_model.postprocess(prediction_dict, shapes)
```

```
input_tensor = tf.convert_to_tensor(test_images_np[i], dtype=tf.float32)
...
preprocessed_image, shapes = detection_model.preprocess(input_tensor)
prediction_dict = detection_model.predict(preprocessed_image, shapes)
return detection_model.postprocess(prediction_dict, shapes)
```



rubber\_ducky: 100.0%