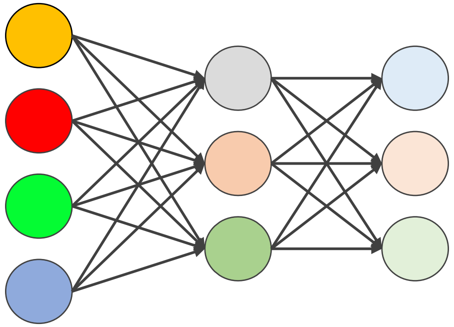
Image Classification Example

→



→

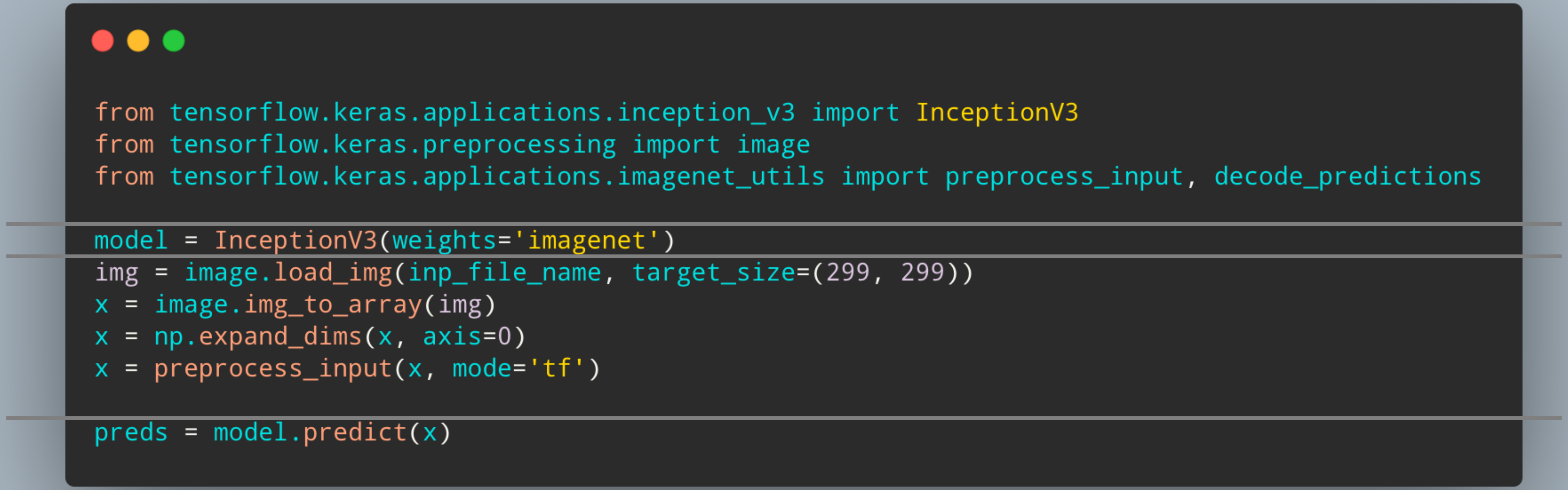
Golden Retriever

Input Image

Pre-trained Model

Prediction Output

Image Classification Tensorflow Code



```python
from tensorflow.keras.applications.inception_v3 import InceptionV3
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.imagenet_utils import preprocess_input, decode_predictions

model = InceptionV3(weights='imagenet')
img = image.load_img(inp_file_name, target_size=(299, 299))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x, mode='tf')

preds = model.predict(x)
```

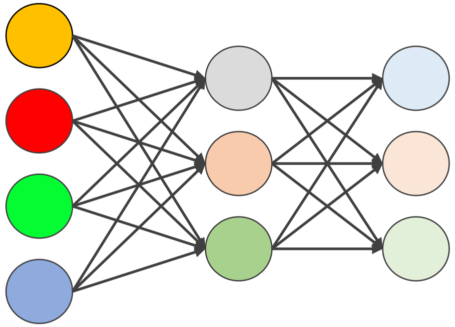
Pre-trained model

Pre-processing

Inference

Pre-processing

[299,299]



filtering

```
golden_retriever: 85.93 %
Labrador_retriever: 5.18 %
kuvasz: 0.94 %
```

Input Image

[dog.jpg]

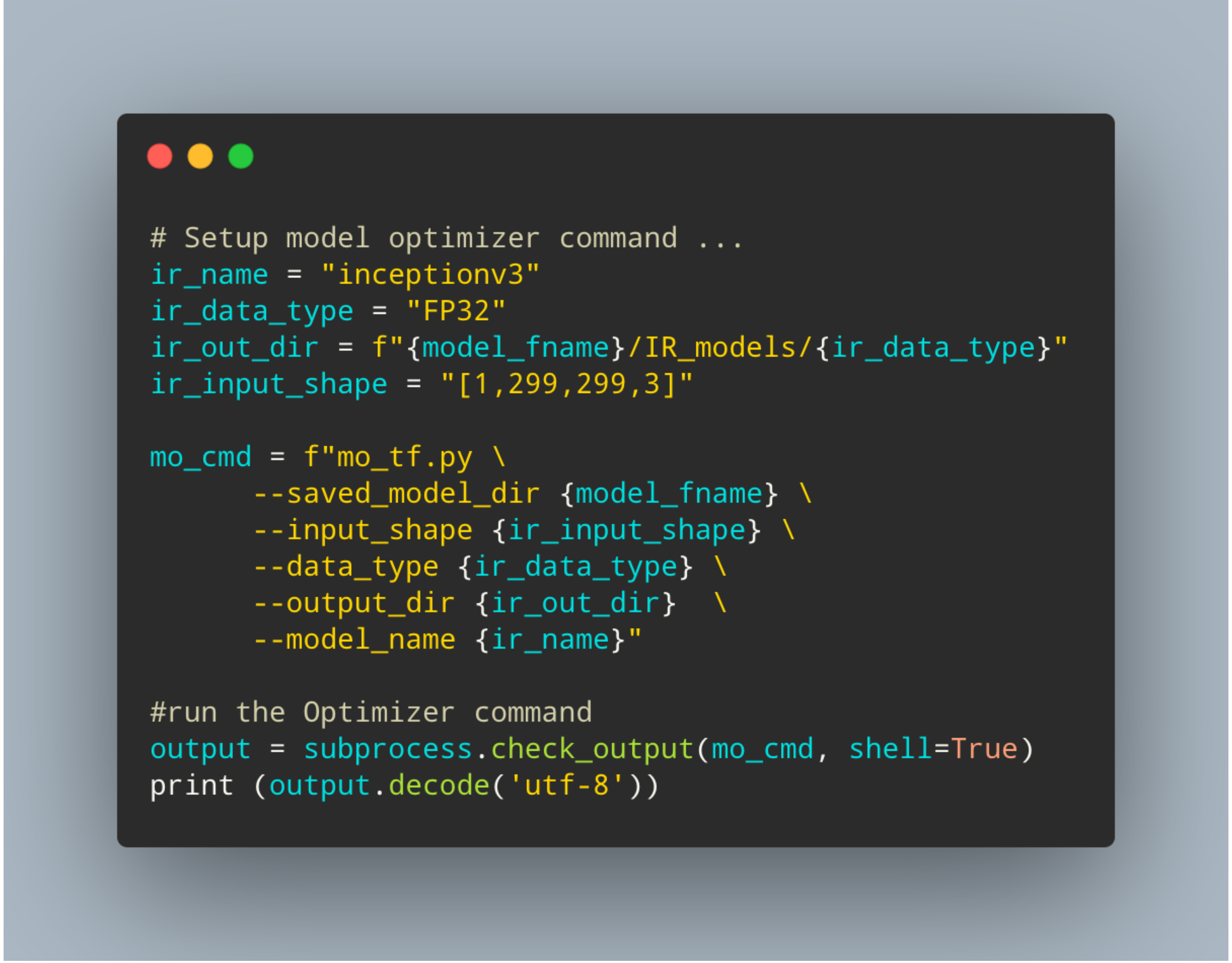
InceptionV3
imagenet

[frozen.pb]

Predictions

[Array]

OpenVINO Model Optimizer

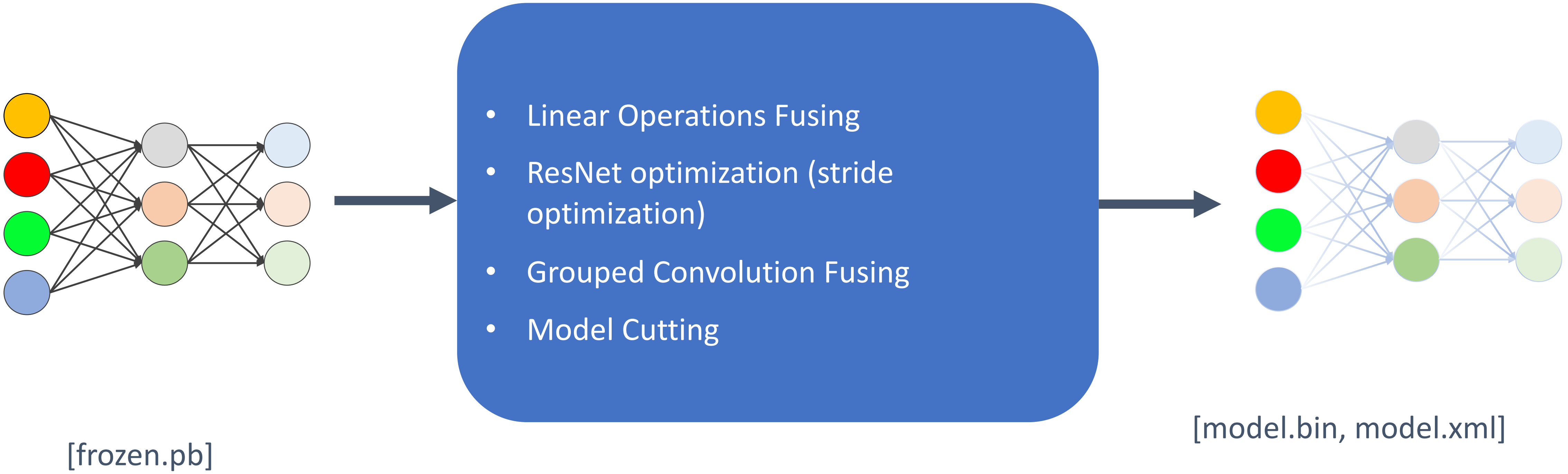


```
# Setup model optimizer command ...
ir_name = "inceptionv3"
ir_data_type = "FP32"
ir_out_dir = f"{model_fname}/IR_models/{ir_data_type}"
ir_input_shape = "[1,299,299,3]"

mo_cmd = f"mo_tf.py \
      --saved_model_dir {model_fname} \
      --input_shape {ir_input_shape} \
      --data_type {ir_data_type} \
      --output_dir {ir_out_dir}  \
      --model_name {ir_name}"

#run the Optimizer command
output = subprocess.check_output(mo_cmd, shell=True)
print (output.decode('utf-8'))
```

OpenVINO
Model Optimizer



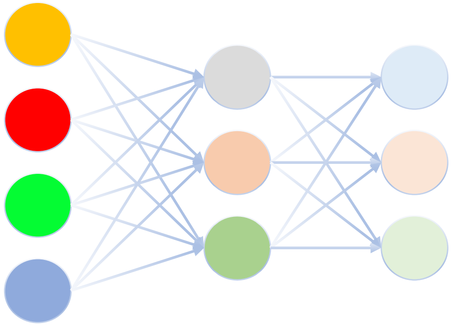
- Linear Operations Fusing
- ResNet optimization (stride optimization)
- Grouped Convolution Fusing
- Model Cutting

[frozen.pb]

[model.bin, model.xml]

Pre-processing →

[299,299]



filtering →

```
golden_retriever: 85.93 %
Labrador_retriever: 5.18 %
kuvasz: 0.94 %
```

Input Image

[dog.jpg]

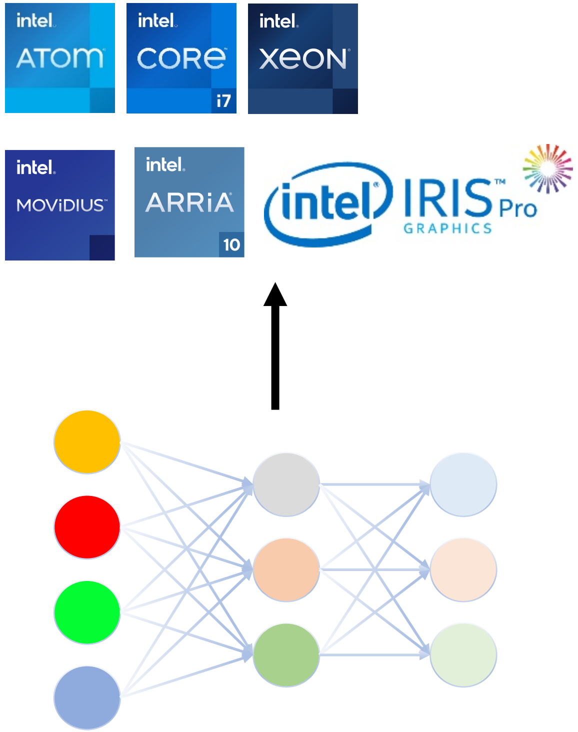
OpenVINO Optimized Model

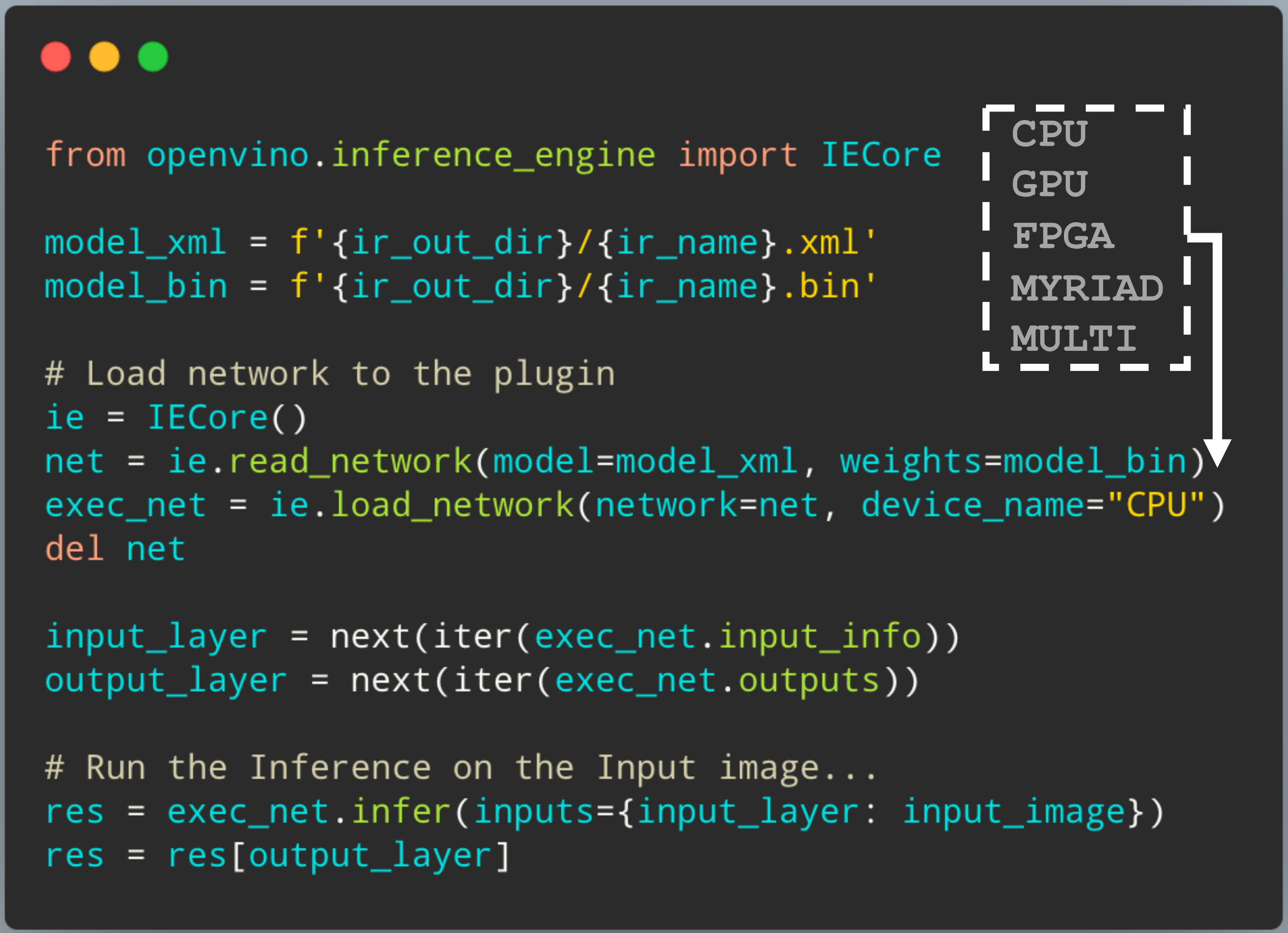
[model.bin, model.xml]

Predictions

[Array]

OpenVINO Inference





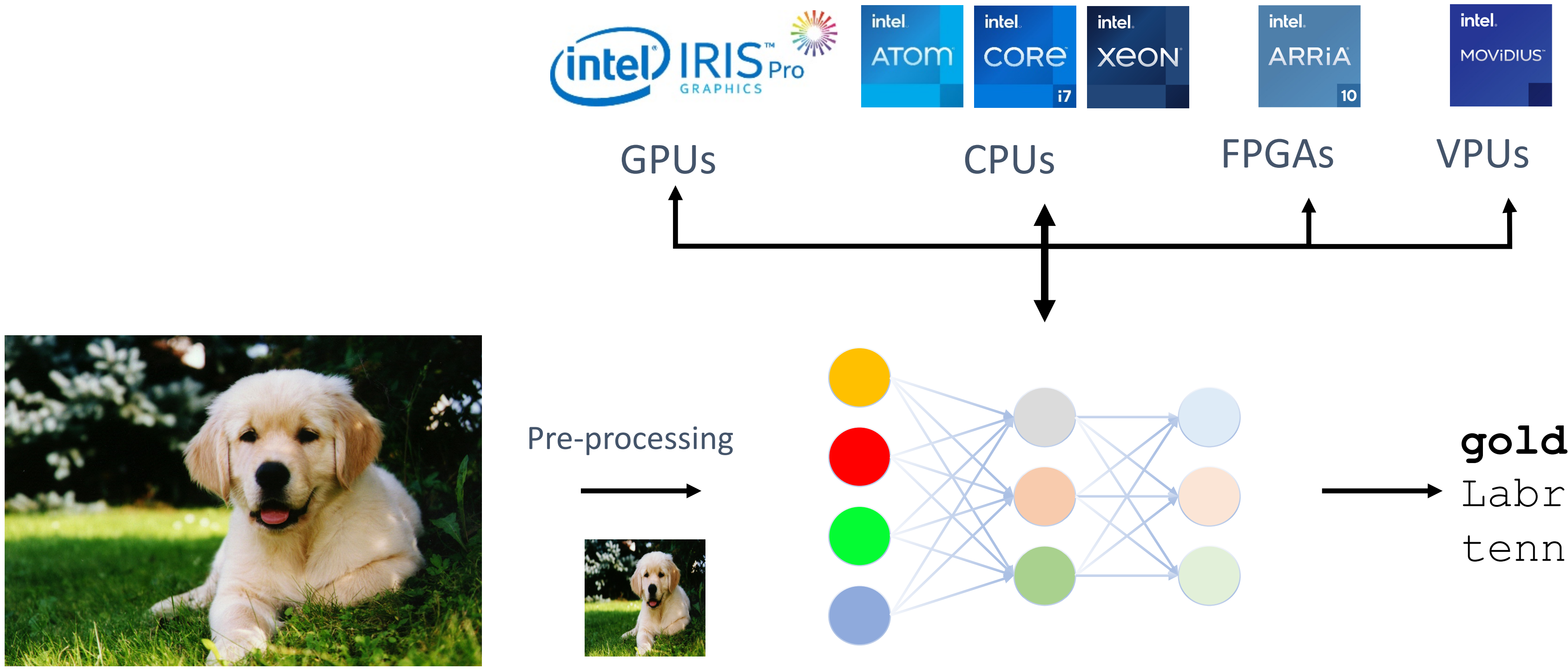
```python
from openvino.inference_engine import IECore

model_xml = f'{ir_out_dir}/{ir_name}.xml'
model_bin = f'{ir_out_dir}/{ir_name}.bin'

# Load network to the plugin
ie = IECore()
net = ie.read_network(model=model_xml, weights=model_bin)
exec_net = ie.load_network(network=net, device_name="CPU")
del net

input_layer = next(iter(exec_net.input_info))
output_layer = next(iter(exec_net.outputs))

# Run the Inference on the Input image...
res = exec_net.infer(inputs={input_layer: input_image})
res = res[output_layer]
```



```
golden_retriever: 89.87 %
Labrador_retriever: 1.84 %
tennis_ball: 0.79 %
```

Input Image

OpenVINO Optimized Pre-trained Model

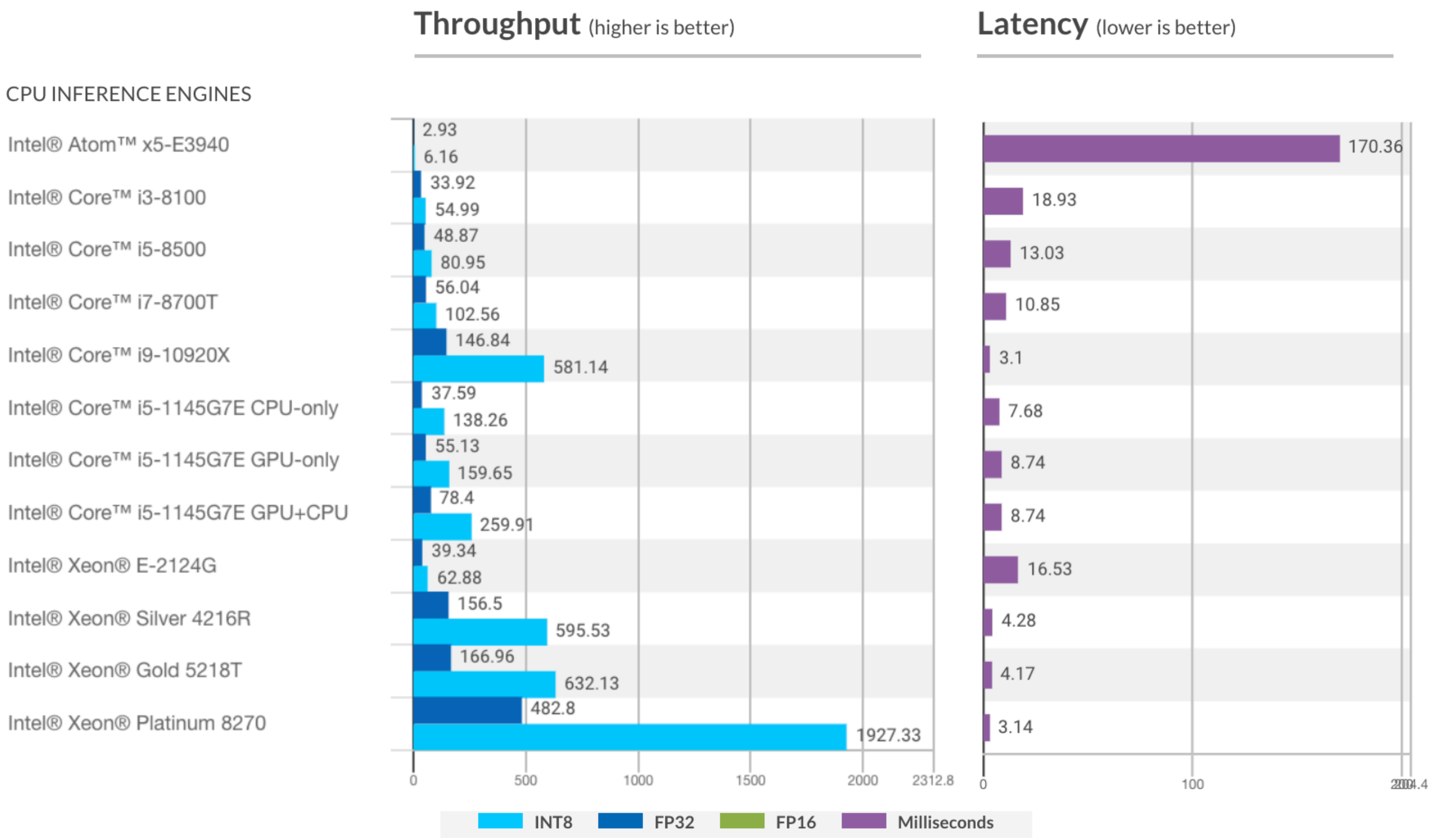
Prediction Outcomes

[dog.jpg]

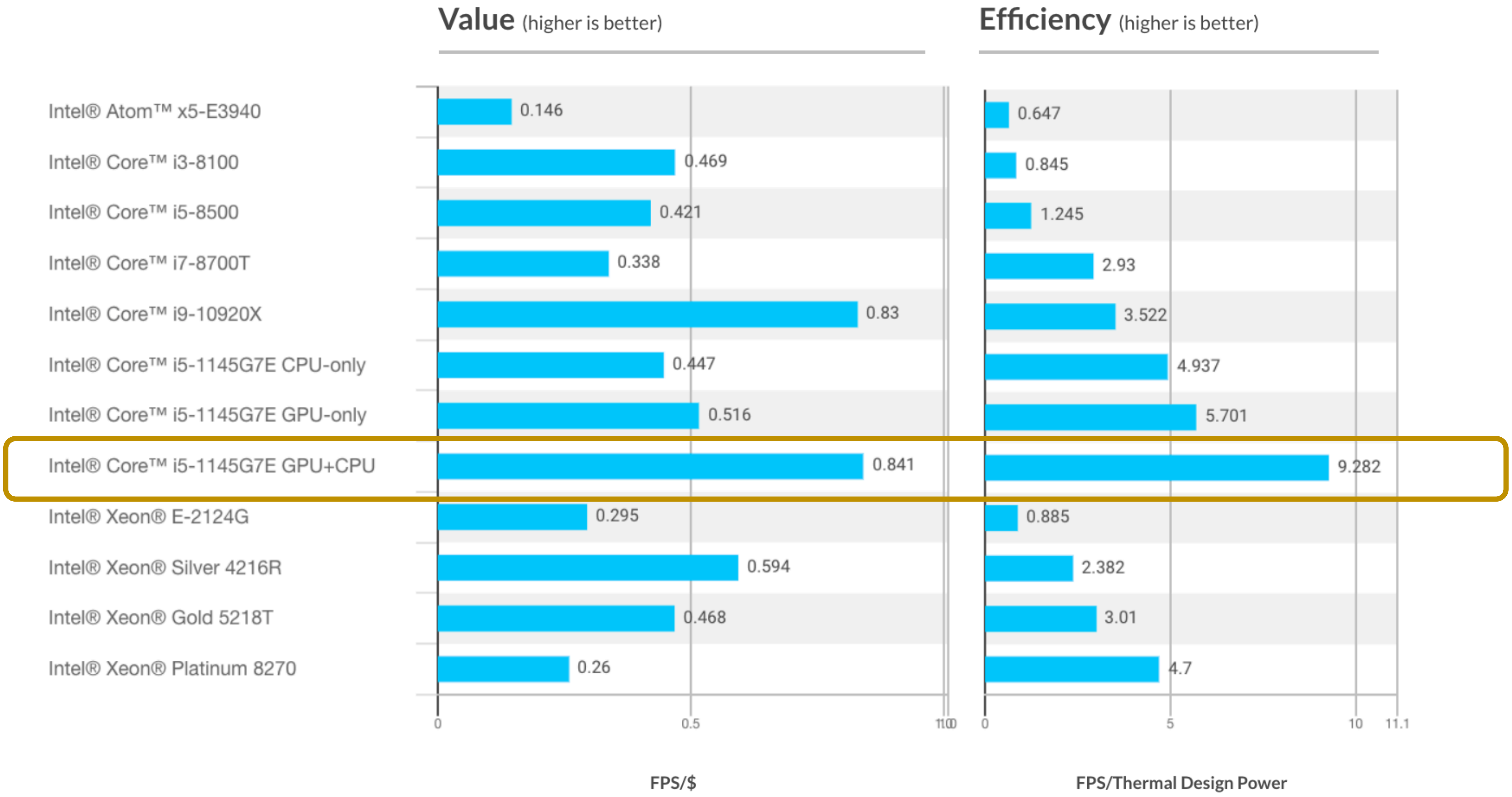
[model.bin, model.xml]

[Array]

inception-v3-TF



Source: https://docs.openvinotoolkit.org/latest/openvino_docs_performance_benchmarks.html



Value (higher is better)

Efficiency (higher is better)