

# DL Training Example

With TF Keras API

<https://medium.com/tensorflow/training-and-serving-ml-models-with-tf-keras-fd975cc0fa27>

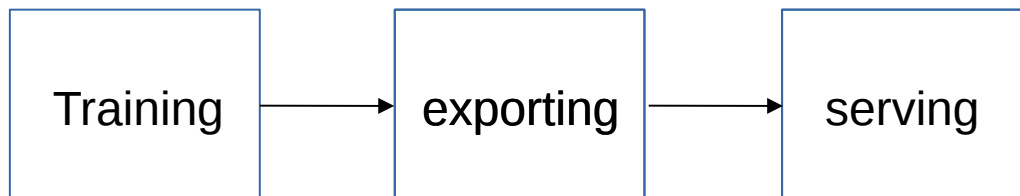
Keras introduced in 2017 is a API for neural networks that runs on multiple backends such as Tensorflow 1.9, as tf.keras. Although tf.keras and Keras have separate code bases, they are tightly coupled.

Prerequisite: (1) TensorFlow, 1.11.0; (2) Keras

To find TF version, `$python -c 'import tensorflow as tf; print(tf.__version__)'`

To find keras version, `$python -c 'import keras; print(keras.__version__)'`

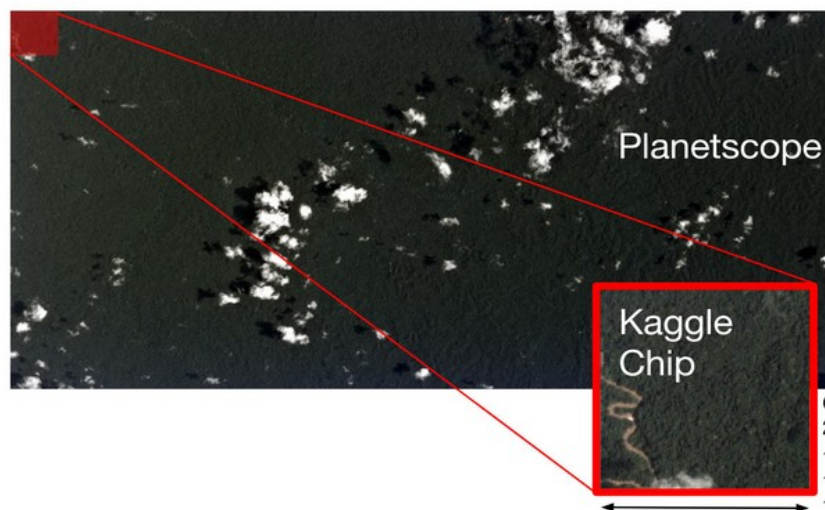
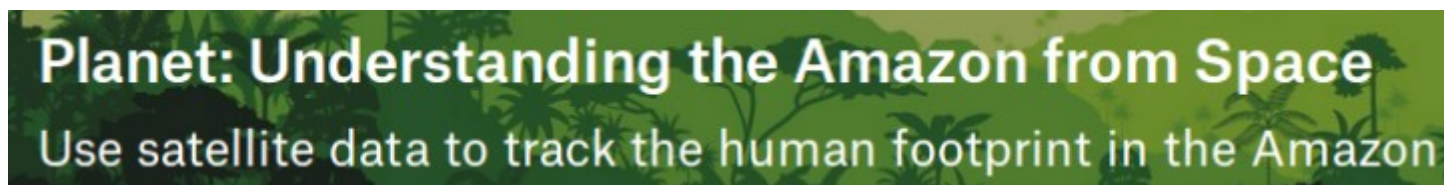
```
python -c 'import keras; print(keras.__version__)'  
Using TensorFlow backend.  
2.2.4
```



# Images From 4-band Planet Sensing Satellite

## With TF Keras API

<https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data>



**Chip Size**

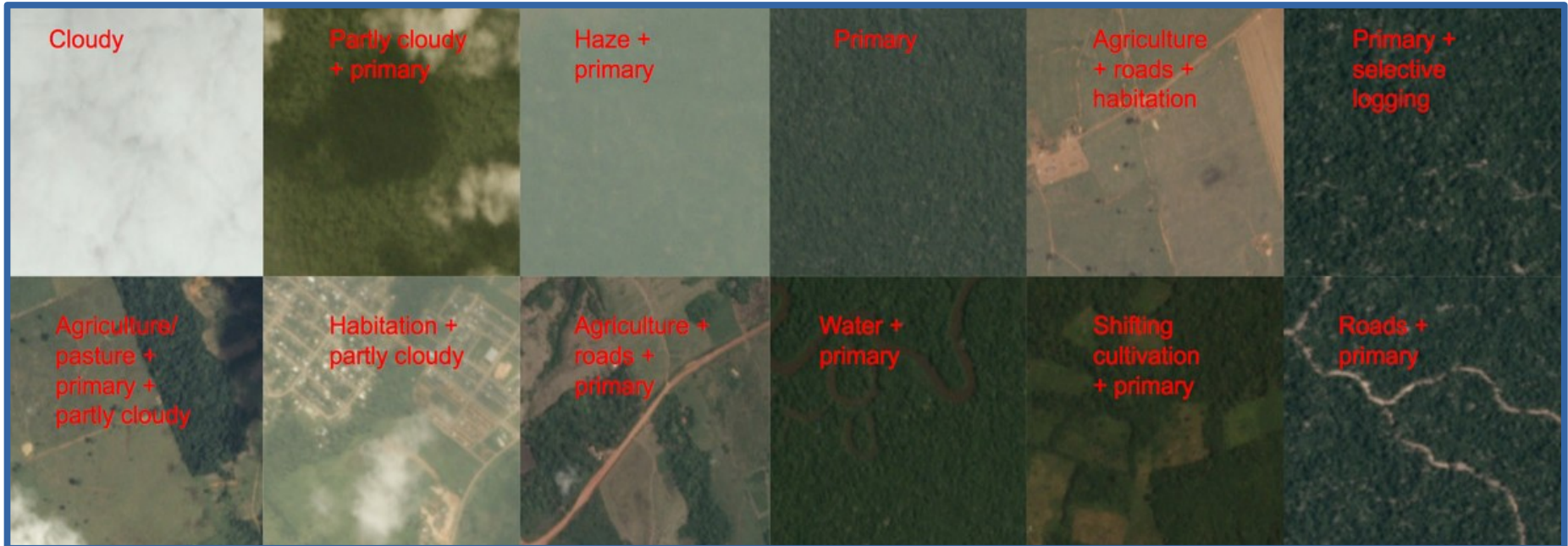
256x256 pixels @ ~3.7m GSD



**Chip (Image) Data Format:** The chips were derived from Planet's full-frame analytic scene products using 4-band satellites in sun-synchronous orbit (SSO) and International Space Station (ISS) orbit. The set of chips use the GeoTiff format and each contain four bands of data: red, green, blue, and near infrared. The specific spectral response of the satellites can be found in the Planet documentation. Each channel is in 16-bit digital number format, and meets the specification of the Planet four band analytic ortho scene product.

# Image Labels Example

<https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data>



The image is available for download on Kaggle. The training data consists of approximately 40000 labeled images of the Amazon rain forest. Each image is associated with multiple labels:

Exactly one 'weather' label: clear, haze, cloudy or partly cloudy

One or more 'ground' labels: agriculture, bare ground, habitation, road, water...

# A Pandas DataFrame For Image Labels

<https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data>

```
image_name,weather_labels,ground_labels
train_0,"[0, 0, 1, 0]", "[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]"
train_1,"[1, 0, 0, 0]", "[1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1]"
train_2,"[1, 0, 0, 0]", "[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]"
train_3,"[1, 0, 0, 0]", "[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]"
train_4,"[1, 0, 0, 0]", "[1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0]"
train_5,"[0, 0, 1, 0]", "[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1]"
train_6,"[1, 0, 0, 0]", "[1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1]"
```

pandas.DataFrame

```
In [1]: import pandas as pd
```

```
In [2]: import numpy as np
```

```
In [3]: import matplotlib.pyplot as plt
```

```
In [9]: df
```

```
Out[9]:
```

	A	B	C	D
2013-01-01	0.469112	-0.282863	-1.509059	-1.135632
2013-01-02	1.212112	-0.173215	0.119209	-1.044236
2013-01-03	-0.861849	-2.104569	-0.494929	1.071804
2013-01-04	0.721555	-0.706771	-1.039575	0.271860
2013-01-05	-0.424972	0.567020	0.276232	-1.087401
2013-01-06	-0.673690	0.113648	-1.478427	0.524988

<https://pandas.pydata.org/pandas-docs/stable/10min.html>

# Build DL Model

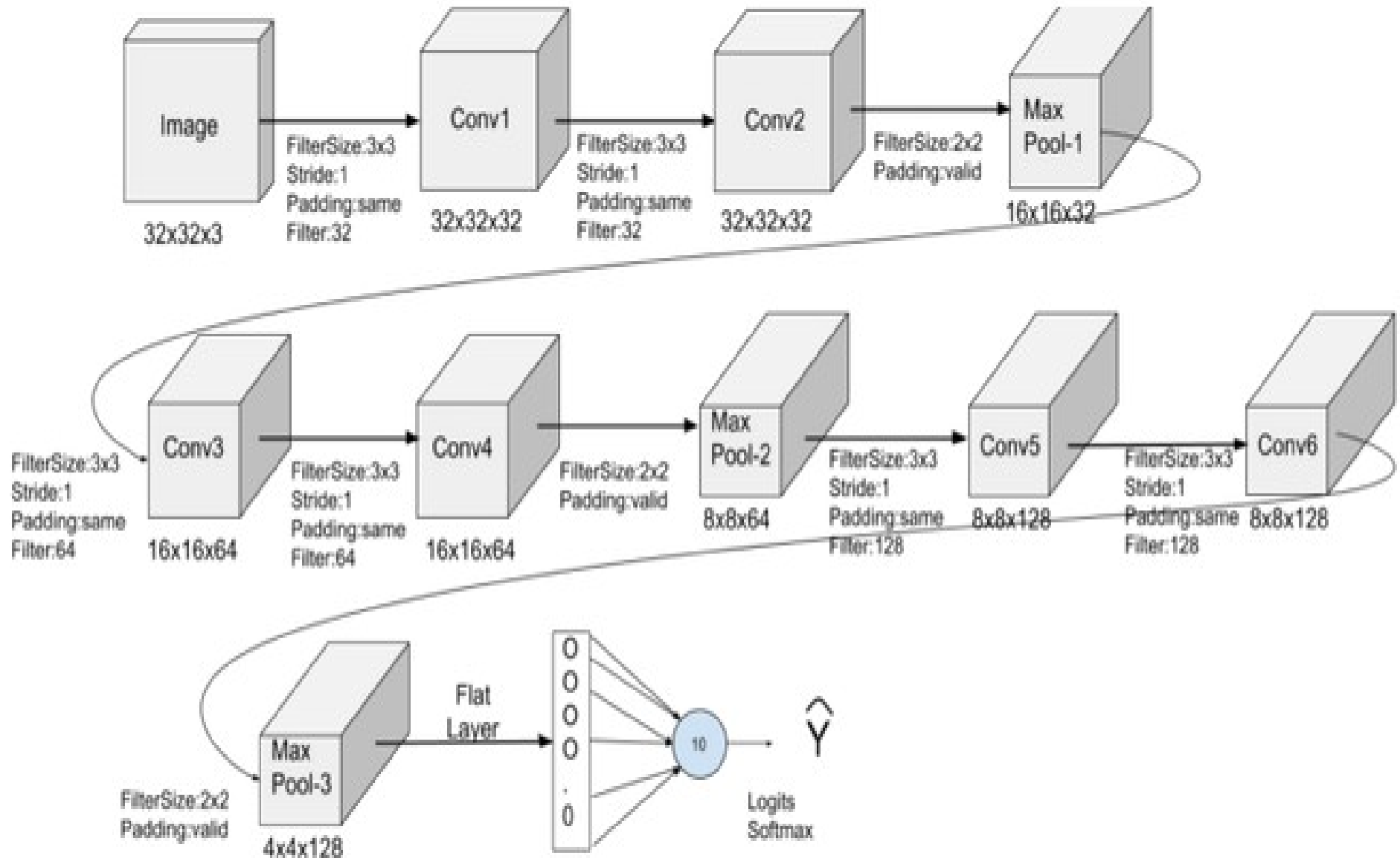
```
import tensorflow as tf
IM_SIZE = 128
image_input = tf.keras.Input(shape=(IM_SIZE, IM_SIZE, 3), name='input_layer')
# Some convolutional layers
conv_1 = tf.keras.layers.Conv2D(32, kernel_size=(3, 3),
                                padding='same',
                                activation='relu')(image_input)
conv_1 = tf.keras.layers.MaxPooling2D(padding='same')(conv_1)
conv_2 = tf.keras.layers.Conv2D(32, kernel_size=(3, 3),
                                padding='same',
                                activation='relu')(conv_1)
conv_2 = tf.keras.layers.MaxPooling2D(padding='same')(conv_2)
# Flatten the output of the convolutional layers
conv_flat = tf.keras.layers.Flatten()(conv_2)
# Some dense layers with two separate outputs
fc_1 = tf.keras.layers.Dense(128, activation='relu')(conv_flat)
fc_1 = tf.keras.layers.Dropout(0.2)(fc_1)
fc_2 = tf.keras.layers.Dense(128, activation='relu')(fc_1)
fc_2 = tf.keras.layers.Dropout(0.2)(fc_2)
# Output layers: separate outputs 4 weather & ground labels
weather_output = tf.keras.layers.Dense(4,
                                       activation='softmax', name='weather')(fc_2)
ground_output = tf.keras.layers.Dense(13, activation='sigmoid',
                                       name='ground')(fc_2)
# Wrap in a Model
model = tf.keras.Model(inputs=image_input, outputs=[weather_output, ground_output])
```

(1) 2 output layers, so they are passed as a list of outputs

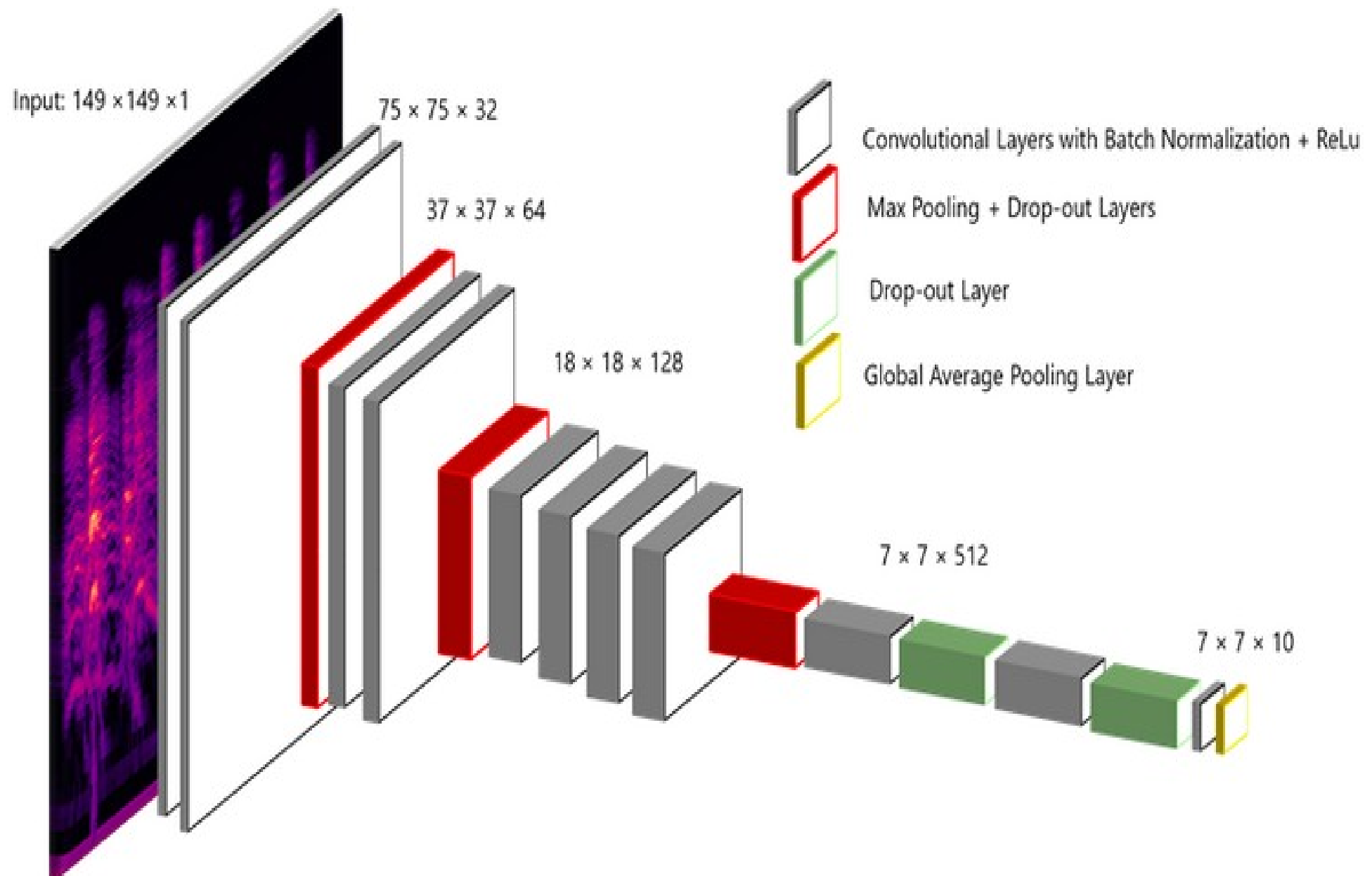
(1) 2 output layers, so they are passed as a list of outputs



# Convolutional Neural Network Architecture Example



# tf.keras Example of ANN Architecture

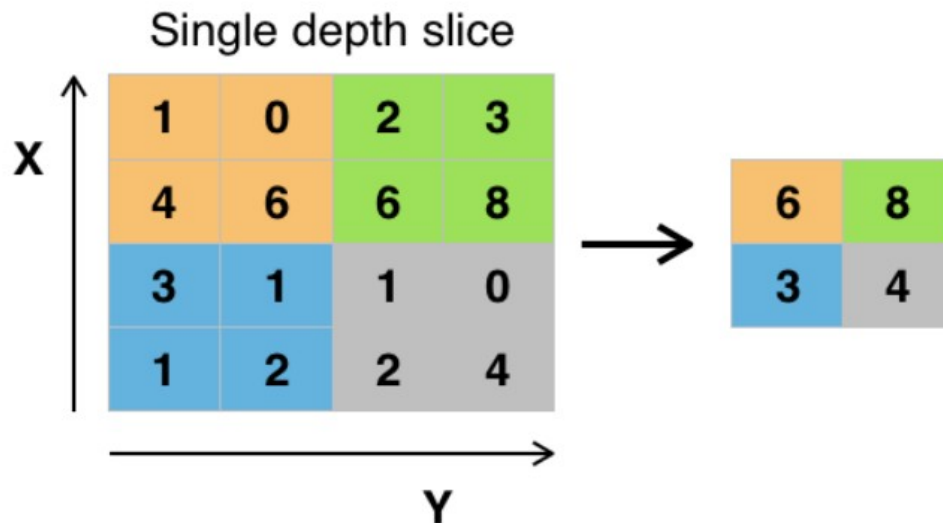


# tf.keras.Model etc

```
import tensorflow as tf
tf.keras.Input( )

tf.keras.layers.Conv2D()
tf.keras.layers.MaxPooling2D()
tf.keras.layers.Flatten()(conv_2)
tf.keras.layers.Dense()(conv_flat)
tf.keras.layers.Dropout( )(fc_1)
tf.keras.Model()
```

[https://www.tensorflow.org/api\\_docs/python/tf/keras/layers/MaxPool2D](https://www.tensorflow.org/api_docs/python/tf/keras/layers/MaxPool2D)



`tf.keras.layers.MaxPooling2D()`



# tf.keras.layers.Dense( ) Activation Functions

