CodeDemo

March 23, 2021

1 Using Machine Learning for Object Detection of StarCraft Units

2 Code Demo

2.0.1 Author: Aeon Williams

2.0.2 December 2020

Updated March 2021 for portfolio

[1]:

```
# Jupyter Q.O.L tools by Aeon Williams
from bae0n_utils import *
FitCellsToWindow()
```

```
<IPython.core.display.HTML object>
```

[2]:

```
import xml.etree.ElementTree as ET # parse, getroot, iter
from sklearn.datasets import load_files
import cv2 # imread, imwrite, selectiveSearchSegmentation, cvtColor, flip,
           # setBaseImage, switchToSelectiveSearchFast, proces
import os # listdir
import keras
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd
import tensorflow as tf
import PIL.Image as Image
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Conv2D,MaxPool2D,Dropout,Flatten,Dense
from keras.optimizers import Adam
from keras.models import load_model
from sklearn.preprocessing import OneHotEncoder
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
from sklearn.model_selection import StratifiedKFold
from keras.utils.vis_utils import plot_model
```

```
Using TensorFlow backend.
```

3 RCNN

Object detection in photos with variable number of classes and objects per photo.

[3]:

```
class Config:
    """
    Configuration settings for model training.

    Attributes:
    -----------
    epochs : int = 100
        Number of epochs to run the model during training
    batchsize : int = 64
        Number of samples per segment of model training
    k_folds : int = 10
        Number of folds for k-fold cross validation
    test_split : float (0-1) = 0.3
        Percentage size of test data
    val_split : float (0-1) = 0.15
        Percentage size of validation data
    image_size : float = 128
        Width & Height of images for the model - must match size used in
        preprocessing
    labels : list of strings = []
        If not empty, these are the only class labels that will be preprocessed
        to use to train the model
    lr : float = 0.0001
        Learning rate of the model
    filename : string = 'model.h5'
        Name of the model to create - must end in .h5
    datapath : string = ''
        Directory to find training data
    patience : int = 10
        Number of epochs to tolerate 0 improvement before early stopping
    es : Bool = True
        Toggle early stopping
    crossvalidate : Bool = False
        Toggle cross validation (if True, runs does not matter)
    runs : int = 1
        Number of times to evaluate model results (does not matter if
        crossvalidate = True)
    savemodel : Bool = False
        Toggle saving the model into a file. filename is only used if this
        is true
    """
    def __init__(self, epochs=100, batchsize=64, k_folds=10, test_split=0.3,
                 val_split=0.15, image_size=128, labels=[], lr=0.0001,
                 filename='model.h5', datapath='', patience=10, es=True,
                 crossvalidate=False, runs=1, savemodel=False):
        """
        Constructs all attributes for the config data.

        Parameters:
            self - Implied "this" parameter

            See class attributes above.
        """
        self.epochs = epochs
        self.batchsize = batchsize
        self.k_folds = k_folds
        self.test_split = test_split
        self.val_split = val_split
        self.image_size = image_size
        self.labels = labels
        self.lr = lr
        self.filename = filename
        self.datapath = datapath
        self.patience = patience
        self.es = es
        self.crossvalidate = crossvalidate
        self.runs = runs
        self.savemodel = savemodel
```

3.1 File I/O

[4]:

```
def parse_annotations(xml_file: str, classes=[]):
    """
    Reads an XML file of image information.

    Parameters:
        xml_file - The name of the xml file to parse

    Returns (in order):
        - The name of the image
        - list of lists of boxes [xmin, ymin, xmax, ymax]
        - list of label names that correspond with each box list
        If xml_file is not found, None is returned for all 3 values.
    """
    try:
        # open the xml file and find the start of the root
        tree = ET.parse(xml_file)
        root = tree.getroot()
        list_with_all_boxes = []
        labels = []
        # each bounding box for an image is stored in "row" chunks
        for rows in root.iter('row'):
            # store relevant information
            filename = rows.find('image').text
            name = rows.find('label').text
            if len(classes) > 0:
                if str(name) not in classes:
                    continue
            ymin = int(float(rows.find('ymin').text))
            ymax = int(float(rows.find('ymax').text))
            xmin = int(float(rows.find('xmin').text))
            xmax = int(float(rows.find('xmax').text))
            labels.append(name)
            list_with_all_boxes.append([xmin, ymin, xmax, ymax])
        return filename, list_with_all_boxes, labels
    # return None if the xml file is not found
    except FileNotFoundError:
        return None, None, None
```

[5]:

```
def load_data(path=''):
    """
    Load image and class label data into lists for model training.

    Parameters:
        path - The directory of images to load. Expects:
               path
               |--positive
               |  |-- images...
               |--negative
               |  |-- images...

    Returns (in order):
        - Array of image data
        - List of corresponding class labels
        - Dictionary {class label: integer encoded version of label}
    """
    # variables for traversing through image files and recording information
    data = load_files(path)
    filename = data['filenames']
    targets = data['target']
    target_names = data['target_names']
    x, y = [], []
    negative_count = 0
    class_count = 1
    classes_dct = {'negative':0}
    # go through every image in the directories (positive & negative)
    for name in filename:
        # split the filename into data path ex: positive/img_1.png and
        # class label ex: 'Zergling'
        path, label, _ = name.strip().split('_')
        # if we haven't come across this class label yet, create a dictionary
        # item for it
        if label not in classes_dct:
            classes_dct[label] = class_count
            class_count += 1
        # the image does not have a labeled class object in it
        if 'negative' in name:
            negative_count += 1
            if negative_count < 2300: # cap for memory/storage reasons
                img = cv2.imread(name)
                img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                x.append(img)
                y.append(0)
            else:
                pass
        # the image has a labeled class object in it
        else:
            image = cv2.imread(name)
            # load in specifically as RGB
            image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            x.append(image)
            # create an augmented copy of the image to boost training data size
            x.append(augment_image(image, name))
            y.append(classes_dct[label])
            y.append(classes_dct[label])

    return np.array(x), y, classes_dct
```

3.2 Image Preprocessing

[6]:

```
def augment_image(image, filename):
    """
    Create an augmented copy of an image and save it in the dataset directory.
    Augment(s): horizontal flip

    Parameters:
        image   - Image data to augment
        filname - Filename of the image data to augment

    Returns:
        - The augmented image data
    """
    # Other augmentations were evaluated, but probably not realistic for the
    # dataset of this specific project, as SC units only have 1 orientation.
    _, path, ext = filename.split('.')
    name = path + 'aug.' + ext
    img = cv2.flip(image, 1) # horizontal flip
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    cv2.imwrite(name, img)
    return img
```

[7]:

```
def compute_iou(box1, box2):
    """
    Computes the Intersection Over Union (iou) of two bounding boxes.
    1 = they are the same box, 0 = they are very far apart.
    iou = Area of Overlap / Area of Union

    https://www.pyimagesearch.com/2016/11/07/
 ↪intersection-over-union-iou-for-object-detection/

    Paramters:
        box1 - One of the bounding box's data. [xmin, ymin, xmax, ymax]
        box2 - One of the bounding box's data. [xmin, ymin, xmax, ymax]

    Returns:
        - iou value (float between 0 & 1)
    """
    # coordinates of the intersection rectangle
    int_x1 = max(box1[0], box2[0])
    int_x2 = min(box1[2], box2[2])
    int_y1 = max(box1[1], box2[1])
    int_y2 = min(box1[3], box2[3])

    # area of overlap rectangle
    int_area = max(0, int_x2 - int_x1 + 1) * max(0, int_y2 - int_y1 + 1)

    # area of prediction and ground truth boxes - for area of union calculation
    box1_area = (box1[2] - box1[0] + 1) * (box1[3] - box1[1] + 1)
    box2_area = (box2[2] - box2[0] + 1) * (box2[3] - box2[1] + 1)

    # intersection over union
    return int_area / float(box1_area + box2_area - int_area)
```

[8]:

```
def preprocess(data_path='',img_type='.jpg', classes=[], dev=-1):
    """
    Separates raw training images into segments of class labeled objects,
    and stores them in corresponding folders. Positive contains class labeled
    objects, negative contains no class labeled objects.
    Parameters:
        data_path - Directory path of the dataset. Expects:
                    data_path
                    |-- image files...
                    |-- xml files with corresponding names...
        img_type - File extension type. Requires preceding dot.

    Results in:
        data_path
        |-- positive
        |   |-- image files named: positive_classlabel_integer.img_type
        |-- negative
        |   |-- image files named: negative_classlabel_integer.img_type
    """
    positive_save_path = data_path+'positive/'
    negative_save_path = data_path+'negative/'
    total_positive = total_negative = 0
    i = 0
    # for each image file in the directory
    for file in os.listdir(data_path):
        # dev tools
        if dev != -1 and i > dev:
            break
        i += 1
        # make sure it's the right kind of file
        if str(img_type) in file:
            # open xml that corresponds with current image file and
            # splits file into filename, box list [xmin, ymin, xmax, ymax],
            # label list that corresponds with box list
            name, box_list, labels = parse_annotations(
                data_path +file.split('.')[0]+'.xml', classes=classes)
            # xml file wasn't found
            if name == None or box_list == None or labels == None:
                continue

            if len(labels) < 1 or str(labels[0]) not in classes:
                continue
            # read in the image data
            pic = cv2.imread(data_path+file)
            pic = cv2.cvtColor(pic, cv2.COLOR_BGR2RGB)
            pic_temp = pic.copy()
            # segment the image with Selective Search
            ss = cv2.ximgproc.segmentation.createSelectiveSearchSegmentation()
            ss.setBaseImage(pic)
            ss.switchToSelectiveSearchFast()
            results = ss.process()
            positive_count = negative_count = total_count = 0
            # evaluate each proposed segment
            for found_box in results:
                found_box_use = [found_box[0],found_box[1],found_box[0]+
                                 found_box[2],found_box[1]+found_box[3]]
                image_roi = pic_temp[found_box[1]:found_box[3]+found_box[1],
                                     found_box[0]:found_box[0]+found_box[2]]
                iou = compute_iou(found_box_use, box_list[0])
                # if the iou of the proposed ssegment and the actual bounding
                # box of the image object is within reasonable threshold,
                # create a positive image segment of size 128x128
                if iou > 0.7:
                    if positive_count < 16:
                        image_roi_use = Image.fromarray(
                            cv2.resize(image_roi,(128,128))).save(
                            positive_save_path+'positive_'+str(labels[0])+
                            '_'+str(total_positive)+'.png')
                        total_positive += 1
                        positive_count += 1
                # if the iou is too small, the proposed segment becomes a
                # negative image of size 128x128
                elif iou < 0.3:
                    if negative_count < 6:
                        image_roi_use = Image.fromarray(
                            cv2.resize(image_roi,(128,128))).save(
                            negative_save_path+'negative_'+str(labels[0])+
                            '_'+str(total_positive)+'.png')
                        total_negative += 1
                        negative_count += 1

                total_count += 1
            print('finished parsing %s' % name)
```

3.3 Model Creation & Training

[9]:

```
def get_model(input_shape, n_classes):
    """
    Creates a sequential model to predict a variable number of class
    labels in an image.

    Parameters:
        input_shape - List of width, height, and channel count of the images
        n_classes   - Number of class labels to predict

    Returns:
        - Sequential model to compile and fit.
    """
    model = Sequential()
    # layered convolution layers and maxpool layers, activated with
    # Rectified Linear Unit so negative values aren't passed to the next layer
    model.add(Conv2D(filters=32,kernel_size=(3,3),input_shape=input_shape,
                     activation='relu'))
    model.add(MaxPool2D(pool_size=(2,2)))
    model.add(Conv2D(filters=64,kernel_size=(3,3),activation='relu'))
    model.add(MaxPool2D(pool_size=(2,2)))
    model.add(Conv2D(filters=128,kernel_size=(3,3),activation='relu'))
    model.add(Conv2D(filters=128,kernel_size=(3,3),activation='relu'))
    model.add(MaxPool2D(pool_size=(2,2)))
    model.add(Conv2D(filters=256,kernel_size=(3,3),activation='relu'))
    model.add(Conv2D(filters=256,kernel_size=(3,3),activation='relu'))
    model.add(MaxPool2D(pool_size=(2,2)))
    # flatten vector from convolutions
    model.add(Flatten())
    # dense & dropout layers
    model.add(Dense(128,activation='relu'))
    model.add(Dense(64,activation='relu'))
    model.add(Dropout(rate=0.35))
    # final layer with size equal to number of class labels to predict
    # softmax because multiclass logistic regression
    model.add(Dense(n_classes,activation='softmax'))

    return model
```

[10]:

```
def compile_fit(X_train, y_train, model, config):
    """
    Compiles and fits the given model.

    Parameters:
        X_train - Train split of dataset
        y_train - Train split of dataset labels
        model   - Model to compile and fit
        config  - Config settings

    Returns (in order):
        - Compiled & fit (trained) model
        - History information about the model training
    """
    # compile the model with Keras
    # lr = 0.0005 1e-05
    model.compile(optimizer=Adam(lr=config.lr), loss='categorical_crossentropy'
                  , metrics=['accuracy'])
    # fit the model to the dataset
    #batch_size: 1=stochastic gradient descent, len(X_train)=gradient descent,
    # 32=minibatch gradient descent
    mc = ModelCheckpoint(config.filename, monitor='val_acc', mode='max',
                         verbose=1, save_best_only=True)
    # check for early stopping
    if config.es == True:
        es = EarlyStopping(monitor='val_loss', mode='min', verbose=1,
                           patience=config.patience)
        calls=[es,mc]
    else:
        calls=[mc]
    history = model.fit(X_train, y_train, validation_split=config.val_split,
                        epochs=config.epochs, batch_size=config.batchsize
                        , callbacks=calls)

    return model, history
```

[11]:

```
def train_model(config):
    """
    Train a multiclass sequential model with image data to predict class
    labels of objets in images.

    Parameters:
        data_path - Directory of images to load to train the model with
        name      - Name of the model file to create

    Returns:
        - Dictionary of {class label: integer encoded label}
        - List of train accuracies
        - List of test accuracies
        - List of model histories
    """
    filename = config.filename
    # load the training data and class label dictionary
    X, y, labels = load_data(config.datapath)
    values = np.array(y)
    # onehotencode the integer representation of the class labels
    values = values.reshape(len(values), 1)
    onehot_encoder = OneHotEncoder(sparse=False)
    Y = onehot_encoder.fit_transform(values)
    n_classes = len(labels)

    # create the model CNN
    model = get_model(input_shape=(int(config.image_size),
                                   int(config.image_size),3)
                      , n_classes=n_classes)

    train_scores = []
    test_scores = []
    histories = []
```

```
# cross validation model evaluation
if config.crossvalidate == True:
    kf = StratifiedKFold(n_splits=config.k_folds, shuffle=True)
    for train_index, test_index in kf.split(X,y):
        values = np.array(y)
        values = values.reshape(len(values), 1)
        Y = onehot_encoder.fit_transform(values)
        fold_Xtrain = X[train_index]
        fold_Ytrain = Y[train_index]
        fold_Xtest = X[test_index]
        fold_Ytest = Y[test_index]

        model, history = compile_fit(fold_Xtrain, fold_Ytrain, model,
                                     config)

        _, train = model.evaluate(fold_Xtrain, fold_Ytrain, verbose=0)
        _, test = model.evaluate(fold_Xtest, fold_Ytest, verbose=0)
        train_scores.append(train)
        test_scores.append(test)
        histories.append(history)
        print("K: %d\tTrain: %.4f\tTest: %.4f" % (len(train_scores)
                                                  , train, test))
# grand mean model evaluation
else:
    for i in range(config.runs):
        # split data into test/train for model creation
        X_train,X_test,y_train,y_test = train_test_split(
            X,Y,test_size=config.test_split, shuffle=True)
        model, history = compile_fit(X_train, y_train, model,
                                         config)
        _, train = model.evaluate(X_train, y_train, verbose=0)
        _, test = model.evaluate(X_test, y_test, verbose=0)
        print("Train: %.4f\tTest: %.4f" % (train, test))
        train_scores.append(train)
        test_scores.append(test)
        histories.append(history)

# last minute dirty fix
for file in os.listdir('datasets/starcraft/labeled/positive/'):
  if 'aug' in file:
    os.remove('datasets/starcraft/labeled/positive/'+file)

for file in os.listdir('datasets/starcraft/labeled/negative/'):
  if 'aug' in file:
    os.remove('datasets/starcraft/labeled/negative/'+file)

if config.savemodel == True:
        model.save(filename)
    #plot_model(model, to_file='model_plot.png', show_shapes=True,␣
↪show_layer_names=True)

    return labels, train_scores, test_scores, histories
```

3.4 Predicting

[12]:

```
def non_max_suppression(boxes, overlapThresh, probs):
    """
    Filter candidate bounding boxes down to the one most relevant, which
    becomes the final predicted box.

    Parameters:
        boxes         - Array of boxes [xmin, ymin, xmax, ymax] to filter
        overlapThresh - Threshold for clustering based on iou
        probs         - Array of probabilities corresponding to boxes

    Returns:
        - List of filtered boxes
    """
    # No boxes
    if len(boxes) == 0:
        return []

    # coordinates of bounding boxes
    x1, x2, y1, y2 = boxes[:,0], boxes[:,2], boxes[:,1], boxes[:,3]
    # area of bounding boxes
    area = (x2 - x1 + 1) * (y2 - y1 + 1)
    # sort bounding boxes by probability
    index = np.argsort(probs)

    final_boxes, pick = [], []
    while len(index) > 0:
        # grab last index in index list and add it to
        # list of picked indices
        last = len(index)-1
        i = index[last]
        pick.append(i)

        # find best coordinates for bounding box
        xx1 = np.maximum(x1[i], x1[index[:last]])
        yy1 = np.maximum(y1[i], y1[index[:last]])
        xx2 = np.minimum(x2[i], x2[index[:last]])
        yy2 = np.minimum(y2[i], y2[index[:last]])
        # width and height of bounding box
        w = np.maximum(0, xx2 - xx1 + 1)
        h = np.maximum(0, yy2 - yy1 + 1)

        overlap = (w*h) / area[index[:last]]
        # remove indices that are over the threshold
        index = np.delete(index, np.concatenate(([last], np.where(
            overlap > overlapThresh)[0])))
        # only return boxes that were picked
        final_boxes.append(boxes[pick])

    return boxes[pick]
```

[47]:

```
def rcnn(image_name, base_model_name, colors=None, labels=None):
    """
    Use a trained rcnn model to predict object locations and class values
    in the given image.

    Parameters:
        image_name      - Name of image file to predict objects in
        base_model_name - Name of trained model file. Must be .h5
    """
    # load in the image
    image = cv2.imread(image_name)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    model = load_model(base_model_name)
    # segment the image with Selective Search to decide object location
    # of objects to attempt to predict the class of
    ss = cv2.ximgproc.segmentation.createSelectiveSearchSegmentation()
    ss.setBaseImage(image)
    ss.switchToSelectiveSearchFast()
    results = ss.process()
    temp1, temp2 = image.copy(), image.copy()
    preds = {}
    # for each found object in the image
    for box in results:
        # prep the segment for predicting
        x1, x2, y1, y2 = box[0], box[0]+box[2], box[1], box[1]+box[3]
        roi = image.copy()[y1:y2,x1:x2]
        roi = cv2.resize(roi,(128,128))
        roi_use = roi.reshape((1,128,128,3))
        # predict the class label for the object
        pred = model.predict_classes(roi_use)[0]
        if pred not in preds:
            # frequency, [probabilities], [positive boxes]
            preds[pred] = [0, [], []]
        preds[pred][0] += 1
        # if there was actually an object found in the segment
        if pred != 0:
            # calculate probabilities that the predicted class is correct
            prob = model.predict(roi_use)[0]
            max_prob = 0
            max_prob_indx = 0
            for i in range(len(prob)):
                if prob[i] > max_prob:
                    max_prob = prob[i]
                    max_prob_indx = i
            # store data
            if max_prob > 0.98:
                preds[pred][2].append([x1,y1,x2,y2])
                preds[pred][1].append(max_prob)
                cv2.rectangle(temp2,(x1,y1),(x2,y2),(255,0,0),5)
    if colors == None:
        colors = [(np.random.randint(100,256),np.random.randint(100,256),
                     np.random.randint(100,256)) for i in range(len(preds))]
    total_boxes = 0
    for key, ls in preds.items():
        color = colors[key]
        if key != 0:
            probs = ls[1]
            positive_boxes = ls[2]
            # filter the proposed boxes down to the one most likely
            cleaned_boxes = non_max_suppression(np.array(positive_boxes),
                                                0.1,probs)
            # display boxes
            for box in cleaned_boxes:
                x1, x2, y1, y2 = box[0], box[2], box[1], box[3]
                total_boxes += 1
                cv2.rectangle(temp1,(x1,y1),(x2,y2),
                              color=color, thickness=3)
        if key != 0:
            cv2.putText(temp1, str(labels[key]), (x1,y1-5), cv2.
↪FONT_HERSHEY_SIMPLEX, .5, (color[0],color[1],color[2]), 1)
    plt.imshow(temp1)
    plt.show()
    print("Total object count: %d" % total_boxes)
```

4 Example

[14]:

```
config = Config(filename='demo_model.h5',
                datapath='./datasets/starcraft/labeled/',
               savemodel=True)
```

4.1 Preprocessing

[15]:
```
%%time
# ClearDir('datasets/starcraft/labeled/positive', safe_del=False)
# ClearDir('datasets/cartoonedzergdata/negative', safe_del=False)
# preprocess(data_path=config.datapath,img_type='.PNG',
#             classes=['Drone', 'Zergling'])
```

```
Wall time: 0 ns
```

4.2 Model Creation

[16]:
```
%%time
labels, train_scores, test_scores, histories = train_model(config)
```

```
WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:74: The name tf.get_default_graph
is deprecated. Please use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:517: The name tf.placeholder is
deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:4138: The name tf.random_uniform is
deprecated. Please use tf.random.uniform instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:3976: The name tf.nn.max_pool is
deprecated. Please use tf.nn.max_pool2d instead.


C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\sklearn\preprocessing\_encoders.py:415: FutureWarning: The handling of
integer data will change in version 0.22. Currently, the categories are
determined based on the range [0, max(values)], while in the future they will be
determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the
categories to integers, then you can now use the OneHotEncoder directly.
  warnings.warn(msg, FutureWarning)

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:133: The name
tf.placeholder_with_default is deprecated. Please use
tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
```

```
packages\keras\backend\tensorflow_backend.py:3445: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed
in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 -
keep_prob`.
WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\optimizers.py:790: The name tf.train.Optimizer is deprecated.
Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:3295: The name tf.log is
deprecated. Please use tf.math.log instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\math_grad.py:1424: where (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:986: The name tf.assign_add is
deprecated. Please use tf.compat.v1.assign_add instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:973: The name tf.assign is
deprecated. Please use tf.compat.v1.assign instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:2741: The name tf.Session is
deprecated. Please use tf.compat.v1.Session instead.

Train on 113 samples, validate on 20 samples
Epoch 1/100
WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:174: The name
tf.get_default_session is deprecated. Please use
tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:181: The name tf.ConfigProto is
deprecated. Please use tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:190: The name tf.global_variables
is deprecated. Please use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
```

```
packages\keras\backend\tensorflow_backend.py:199: The name
tf.is_variable_initialized is deprecated. Please use
tf.compat.v1.is_variable_initialized instead.

WARNING:tensorflow:From C:\Users\Aeon Williams\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:206: The name
tf.variables_initializer is deprecated. Please use
tf.compat.v1.variables_initializer instead.

113/113 [==============================] - 10s 90ms/step - loss: 4.4393 - acc:
0.3628 - val_loss: 2.1342 - val_acc: 0.5500

Epoch 00001: val_acc improved from -inf to 0.55000, saving model to
demo_model.h5
Epoch 2/100
113/113 [==============================] - 8s 68ms/step - loss: 2.3070 - acc:
0.5310 - val_loss: 0.9178 - val_acc: 0.6500

Epoch 00002: val_acc improved from 0.55000 to 0.65000, saving model to
demo_model.h5
Epoch 3/100
113/113 [==============================] - 8s 69ms/step - loss: 1.4129 - acc:
0.4956 - val_loss: 0.8607 - val_acc: 0.5500

Epoch 00003: val_acc did not improve from 0.65000
Epoch 4/100
113/113 [==============================] - 8s 69ms/step - loss: 1.2922 - acc:
0.5841 - val_loss: 0.7817 - val_acc: 0.6000

Epoch 00004: val_acc did not improve from 0.65000
Epoch 5/100
113/113 [==============================] - 8s 71ms/step - loss: 1.2258 - acc:
0.5929 - val_loss: 0.6920 - val_acc: 0.7500

Epoch 00005: val_acc improved from 0.65000 to 0.75000, saving model to
demo_model.h5
Epoch 6/100
113/113 [==============================] - 8s 70ms/step - loss: 0.9739 - acc:
0.6283 - val_loss: 0.5160 - val_acc: 0.8500

Epoch 00006: val_acc improved from 0.75000 to 0.85000, saving model to
demo_model.h5
Epoch 7/100
113/113 [==============================] - 8s 71ms/step - loss: 0.6565 - acc:
0.6991 - val_loss: 0.5816 - val_acc: 0.7000

Epoch 00007: val_acc did not improve from 0.85000
Epoch 8/100
```

```
113/113 [==============================] - 8s 69ms/step - loss: 0.7204 - acc:
0.7080 - val_loss: 0.5284 - val_acc: 0.6500

Epoch 00008: val_acc did not improve from 0.85000
Epoch 9/100
113/113 [==============================] - 8s 70ms/step - loss: 0.5111 - acc:
0.7965 - val_loss: 0.4645 - val_acc: 0.7500

Epoch 00009: val_acc did not improve from 0.85000
Epoch 10/100
113/113 [==============================] - 8s 70ms/step - loss: 0.4372 - acc:
0.8407 - val_loss: 0.4483 - val_acc: 0.8500

Epoch 00010: val_acc did not improve from 0.85000
Epoch 11/100
113/113 [==============================] - 8s 73ms/step - loss: 0.4579 - acc:
0.8053 - val_loss: 0.4197 - val_acc: 0.8000

Epoch 00011: val_acc did not improve from 0.85000
Epoch 12/100
113/113 [==============================] - 8s 73ms/step - loss: 0.4040 - acc:
0.8407 - val_loss: 0.4019 - val_acc: 0.7500

Epoch 00012: val_acc did not improve from 0.85000
Epoch 13/100
113/113 [==============================] - 8s 67ms/step - loss: 0.3025 - acc:
0.8850 - val_loss: 0.4031 - val_acc: 0.8000

Epoch 00013: val_acc did not improve from 0.85000
Epoch 14/100
113/113 [==============================] - 7s 66ms/step - loss: 0.2973 - acc:
0.8673 - val_loss: 0.3394 - val_acc: 0.8500

Epoch 00014: val_acc did not improve from 0.85000
Epoch 15/100
113/113 [==============================] - 7s 66ms/step - loss: 0.2487 - acc:
0.9292 - val_loss: 0.3069 - val_acc: 0.8000

Epoch 00015: val_acc did not improve from 0.85000
Epoch 16/100
113/113 [==============================] - 8s 68ms/step - loss: 0.2673 - acc:
0.9027 - val_loss: 0.3016 - val_acc: 0.8000

Epoch 00016: val_acc did not improve from 0.85000
Epoch 17/100
113/113 [==============================] - 8s 69ms/step - loss: 0.1971 - acc:
0.9469 - val_loss: 0.2997 - val_acc: 0.8500
```

```
Epoch 00017: val_acc did not improve from 0.85000
Epoch 18/100
113/113 [==============================] - 8s 70ms/step - loss: 0.2113 - acc:
0.9381 - val_loss: 0.2779 - val_acc: 0.8500

Epoch 00018: val_acc did not improve from 0.85000
Epoch 19/100
113/113 [==============================] - 8s 71ms/step - loss: 0.1461 - acc:
0.9912 - val_loss: 0.2851 - val_acc: 0.8500

Epoch 00019: val_acc did not improve from 0.85000
Epoch 20/100
113/113 [==============================] - 8s 67ms/step - loss: 0.1729 - acc:
0.9292 - val_loss: 0.2764 - val_acc: 0.8500

Epoch 00020: val_acc did not improve from 0.85000
Epoch 21/100
113/113 [==============================] - 8s 69ms/step - loss: 0.1103 - acc:
0.9558 - val_loss: 0.2725 - val_acc: 0.8500

Epoch 00021: val_acc did not improve from 0.85000
Epoch 22/100
113/113 [==============================] - 7s 66ms/step - loss: 0.0918 - acc:
0.9823 - val_loss: 0.2526 - val_acc: 0.8500

Epoch 00022: val_acc did not improve from 0.85000
Epoch 23/100
113/113 [==============================] - 8s 67ms/step - loss: 0.0984 - acc:
0.9912 - val_loss: 0.2479 - val_acc: 0.8500

Epoch 00023: val_acc did not improve from 0.85000
Epoch 24/100
113/113 [==============================] - 8s 68ms/step - loss: 0.0906 - acc:
0.9912 - val_loss: 0.2486 - val_acc: 0.8500

Epoch 00024: val_acc did not improve from 0.85000
Epoch 25/100
113/113 [==============================] - 7s 66ms/step - loss: 0.0783 - acc:
0.9912 - val_loss: 0.2702 - val_acc: 0.9000

Epoch 00025: val_acc improved from 0.85000 to 0.90000, saving model to
demo_model.h5
Epoch 26/100
113/113 [==============================] - 8s 68ms/step - loss: 0.0747 - acc:
0.9823 - val_loss: 0.2562 - val_acc: 0.9000

Epoch 00026: val_acc did not improve from 0.90000
Epoch 27/100
```

```
113/113 [==============================] - 8s 69ms/step - loss: 0.0583 - acc:
1.0000 - val_loss: 0.2462 - val_acc: 0.9000

Epoch 00027: val_acc did not improve from 0.90000
Epoch 28/100
113/113 [==============================] - 8s 67ms/step - loss: 0.0631 - acc:
0.9912 - val_loss: 0.2514 - val_acc: 0.9000

Epoch 00028: val_acc did not improve from 0.90000
Epoch 29/100
113/113 [==============================] - 8s 68ms/step - loss: 0.0412 - acc:
1.0000 - val_loss: 0.2425 - val_acc: 0.9000

Epoch 00029: val_acc did not improve from 0.90000
Epoch 30/100
113/113 [==============================] - 8s 70ms/step - loss: 0.0370 - acc:
1.0000 - val_loss: 0.2362 - val_acc: 0.9000

Epoch 00030: val_acc did not improve from 0.90000
Epoch 31/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0354 - acc:
1.0000 - val_loss: 0.2423 - val_acc: 0.9000

Epoch 00031: val_acc did not improve from 0.90000
Epoch 32/100
113/113 [==============================] - 8s 69ms/step - loss: 0.0333 - acc:
1.0000 - val_loss: 0.2399 - val_acc: 0.9000

Epoch 00032: val_acc did not improve from 0.90000
Epoch 33/100
113/113 [==============================] - 9s 76ms/step - loss: 0.0317 - acc:
1.0000 - val_loss: 0.2233 - val_acc: 0.9000

Epoch 00033: val_acc did not improve from 0.90000
Epoch 34/100
113/113 [==============================] - 8s 69ms/step - loss: 0.0467 - acc:
0.9823 - val_loss: 0.1977 - val_acc: 0.9000

Epoch 00034: val_acc did not improve from 0.90000
Epoch 35/100
113/113 [==============================] - 8s 71ms/step - loss: 0.0361 - acc:
1.0000 - val_loss: 0.1850 - val_acc: 0.9500

Epoch 00035: val_acc improved from 0.90000 to 0.95000, saving model to
demo_model.h5
Epoch 36/100
113/113 [==============================] - 8s 70ms/step - loss: 0.0258 - acc:
1.0000 - val_loss: 0.1807 - val_acc: 0.9500
```

```
Epoch 00036: val_acc did not improve from 0.95000
Epoch 37/100
113/113 [==============================] - 8s 68ms/step - loss: 0.0387 - acc:
0.9823 - val_loss: 0.1969 - val_acc: 0.9000

Epoch 00037: val_acc did not improve from 0.95000
Epoch 38/100
113/113 [==============================] - 8s 70ms/step - loss: 0.0351 - acc:
0.9912 - val_loss: 0.1536 - val_acc: 0.9000

Epoch 00038: val_acc did not improve from 0.95000
Epoch 39/100
113/113 [==============================] - 8s 70ms/step - loss: 0.0225 - acc:
1.0000 - val_loss: 0.1506 - val_acc: 0.9000

Epoch 00039: val_acc did not improve from 0.95000
Epoch 40/100
113/113 [==============================] - 8s 71ms/step - loss: 0.0298 - acc:
1.0000 - val_loss: 0.1797 - val_acc: 0.9000

Epoch 00040: val_acc did not improve from 0.95000
Epoch 41/100
113/113 [==============================] - 8s 67ms/step - loss: 0.0262 - acc:
1.0000 - val_loss: 0.2440 - val_acc: 0.9000

Epoch 00041: val_acc did not improve from 0.95000
Epoch 42/100
113/113 [==============================] - 7s 66ms/step - loss: 0.0235 - acc:
1.0000 - val_loss: 0.1965 - val_acc: 0.9000

Epoch 00042: val_acc did not improve from 0.95000
Epoch 43/100
113/113 [==============================] - 7s 62ms/step - loss: 0.0161 - acc:
1.0000 - val_loss: 0.1712 - val_acc: 0.9500

Epoch 00043: val_acc did not improve from 0.95000
Epoch 44/100
113/113 [==============================] - 7s 66ms/step - loss: 0.0202 - acc:
1.0000 - val_loss: 0.1702 - val_acc: 0.9500

Epoch 00044: val_acc did not improve from 0.95000
Epoch 45/100
113/113 [==============================] - 8s 71ms/step - loss: 0.0215 - acc:
1.0000 - val_loss: 0.1773 - val_acc: 0.9500

Epoch 00045: val_acc did not improve from 0.95000
Epoch 46/100
```

```
113/113 [==============================] - 7s 65ms/step - loss: 0.0248 - acc:
0.9912 - val_loss: 0.1619 - val_acc: 0.9000

Epoch 00046: val_acc did not improve from 0.95000
Epoch 47/100
113/113 [==============================] - 8s 71ms/step - loss: 0.0162 - acc:
1.0000 - val_loss: 0.1690 - val_acc: 0.9000

Epoch 00047: val_acc did not improve from 0.95000
Epoch 48/100
113/113 [==============================] - 8s 75ms/step - loss: 0.0187 - acc:
0.9912 - val_loss: 0.1507 - val_acc: 0.9000

Epoch 00048: val_acc did not improve from 0.95000
Epoch 49/100
113/113 [==============================] - 8s 71ms/step - loss: 0.0180 - acc:
0.9912 - val_loss: 0.1265 - val_acc: 0.9000

Epoch 00049: val_acc did not improve from 0.95000
Epoch 50/100
113/113 [==============================] - 8s 73ms/step - loss: 0.0153 - acc:
1.0000 - val_loss: 0.1229 - val_acc: 0.9000

Epoch 00050: val_acc did not improve from 0.95000
Epoch 51/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0114 - acc:
1.0000 - val_loss: 0.1267 - val_acc: 0.9500

Epoch 00051: val_acc did not improve from 0.95000
Epoch 52/100
113/113 [==============================] - 9s 76ms/step - loss: 0.0091 - acc:
1.0000 - val_loss: 0.1465 - val_acc: 0.9500

Epoch 00052: val_acc did not improve from 0.95000
Epoch 53/100
113/113 [==============================] - 8s 71ms/step - loss: 0.0090 - acc:
1.0000 - val_loss: 0.1870 - val_acc: 0.9500

Epoch 00053: val_acc did not improve from 0.95000
Epoch 54/100
113/113 [==============================] - 9s 76ms/step - loss: 0.0195 - acc:
0.9912 - val_loss: 0.1413 - val_acc: 0.9500

Epoch 00054: val_acc did not improve from 0.95000
Epoch 55/100
113/113 [==============================] - 8s 75ms/step - loss: 0.0071 - acc:
1.0000 - val_loss: 0.0737 - val_acc: 1.0000
```

```
Epoch 00055: val_acc improved from 0.95000 to 1.00000, saving model to
demo_model.h5
Epoch 56/100
113/113 [==============================] - 8s 75ms/step - loss: 0.0179 - acc:
0.9912 - val_loss: 0.0684 - val_acc: 1.0000

Epoch 00056: val_acc did not improve from 1.00000
Epoch 57/100
113/113 [==============================] - 8s 72ms/step - loss: 0.0107 - acc:
1.0000 - val_loss: 0.0865 - val_acc: 0.9500

Epoch 00057: val_acc did not improve from 1.00000
Epoch 58/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0099 - acc:
1.0000 - val_loss: 0.1271 - val_acc: 0.9000

Epoch 00058: val_acc did not improve from 1.00000
Epoch 59/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0071 - acc:
1.0000 - val_loss: 0.1845 - val_acc: 0.9000

Epoch 00059: val_acc did not improve from 1.00000
Epoch 60/100
113/113 [==============================] - 8s 74ms/step - loss: 0.0148 - acc:
1.0000 - val_loss: 0.2585 - val_acc: 0.9000

Epoch 00060: val_acc did not improve from 1.00000
Epoch 61/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0120 - acc:
1.0000 - val_loss: 0.2294 - val_acc: 0.9000

Epoch 00061: val_acc did not improve from 1.00000
Epoch 62/100
113/113 [==============================] - 10s 86ms/step - loss: 0.0066 - acc:
1.0000 - val_loss: 0.1494 - val_acc: 0.9000

Epoch 00062: val_acc did not improve from 1.00000
Epoch 63/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0054 - acc:
1.0000 - val_loss: 0.1072 - val_acc: 0.9000

Epoch 00063: val_acc did not improve from 1.00000
Epoch 64/100
113/113 [==============================] - 9s 84ms/step - loss: 0.0066 - acc:
1.0000 - val_loss: 0.0884 - val_acc: 0.9500

Epoch 00064: val_acc did not improve from 1.00000
Epoch 65/100
```

```
113/113 [==============================] - 9s 76ms/step - loss: 0.0057 - acc:
1.0000 - val_loss: 0.0672 - val_acc: 1.0000

Epoch 00065: val_acc did not improve from 1.00000
Epoch 66/100
113/113 [==============================] - 9s 76ms/step - loss: 0.0086 - acc:
1.0000 - val_loss: 0.0638 - val_acc: 1.0000

Epoch 00066: val_acc did not improve from 1.00000
Epoch 67/100
113/113 [==============================] - 9s 78ms/step - loss: 0.0055 - acc:
1.0000 - val_loss: 0.0842 - val_acc: 0.9500

Epoch 00067: val_acc did not improve from 1.00000
Epoch 68/100
113/113 [==============================] - 9s 81ms/step - loss: 0.0209 - acc:
1.0000 - val_loss: 0.1283 - val_acc: 0.9000

Epoch 00068: val_acc did not improve from 1.00000
Epoch 69/100
113/113 [==============================] - 8s 74ms/step - loss: 0.0070 - acc:
1.0000 - val_loss: 0.1514 - val_acc: 0.9000

Epoch 00069: val_acc did not improve from 1.00000
Epoch 70/100
113/113 [==============================] - 8s 72ms/step - loss: 0.0056 - acc:
1.0000 - val_loss: 0.1430 - val_acc: 0.9000

Epoch 00070: val_acc did not improve from 1.00000
Epoch 71/100
113/113 [==============================] - 8s 73ms/step - loss: 0.0058 - acc:
1.0000 - val_loss: 0.1164 - val_acc: 0.9000

Epoch 00071: val_acc did not improve from 1.00000
Epoch 72/100
113/113 [==============================] - 8s 75ms/step - loss: 0.0079 - acc:
1.0000 - val_loss: 0.1009 - val_acc: 0.9000

Epoch 00072: val_acc did not improve from 1.00000
Epoch 73/100
113/113 [==============================] - 8s 75ms/step - loss: 0.0068 - acc:
1.0000 - val_loss: 0.1192 - val_acc: 0.9000

Epoch 00073: val_acc did not improve from 1.00000
Epoch 74/100
113/113 [==============================] - 8s 73ms/step - loss: 0.0055 - acc:
1.0000 - val_loss: 0.1498 - val_acc: 0.9000
```

```
Epoch 00074: val_acc did not improve from 1.00000
Epoch 75/100
113/113 [==============================] - 8s 74ms/step - loss: 0.0166 - acc:
0.9912 - val_loss: 0.1102 - val_acc: 0.9500

Epoch 00075: val_acc did not improve from 1.00000
Epoch 76/100
113/113 [==============================] - 8s 75ms/step - loss: 0.0048 - acc:
1.0000 - val_loss: 0.1149 - val_acc: 0.9500

Epoch 00076: val_acc did not improve from 1.00000
Epoch 00076: early stopping
Train: 0.9925   Test: 0.9655
Wall time: 10min 25s
```

4.3 Label Predicting

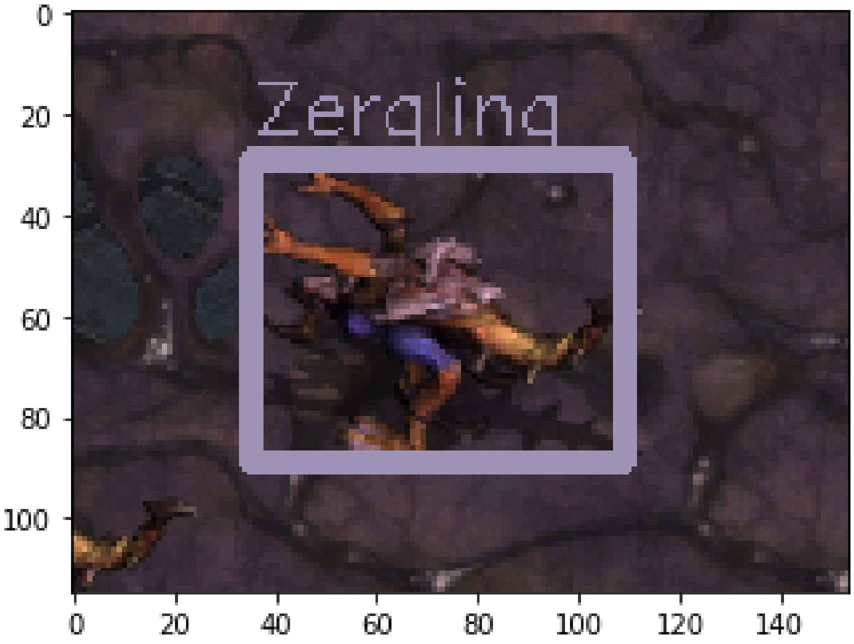
[17]:

```
display(labels)
colors = [(np.random.randint(100,256),np.random.randint(100,256),
             np.random.randint(100,256)) for i in range(len(labels))]
```

```
{'negative': 0, 'Drone': 1, 'Zergling': 2}
```

[48]:

```
%%time
rcnn('./datasets/starcraft/labeled/Capture20.PNG', config.filename,␣
 ↪colors=colors, labels=list(labels))
```



```
Total object count: 1
Wall time: 14.9 s
```