Homework11

Steven Black

4/28/2019

IS 677: Introduction to Data Science Spring 2019

Homework Assignment 11 (Due: May 5, 2019, midnight EST)

1. Load Keras, clear the previous session, and load the full stack VGG16 model. (15 points)

```
library(keras)
k_clear_session()

conv_base <- application_vgg16(
  weights = "imagenet",
  include_top = FALSE
)

conv_base
```

```
## Model
## ___________________________________________________________________________
## Layer (type)                     Output Shape                  Param #
## ===========================================================================
## input_1 (InputLayer)             (None, None, None, 3)         0
## ___________________________________________________________________________
## block1_conv1 (Conv2D)            (None, None, None, 64)        1792
## ___________________________________________________________________________
## block1_conv2 (Conv2D)            (None, None, None, 64)        36928
## ___________________________________________________________________________
## block1_pool (MaxPooling2D)       (None, None, None, 64)        0
## ___________________________________________________________________________
## block2_conv1 (Conv2D)            (None, None, None, 128)       73856
## ___________________________________________________________________________
## block2_conv2 (Conv2D)            (None, None, None, 128)       147584
## ___________________________________________________________________________
## block2_pool (MaxPooling2D)       (None, None, None, 128)       0
## ___________________________________________________________________________
## block3_conv1 (Conv2D)            (None, None, None, 256)       295168
## ___________________________________________________________________________
## block3_conv2 (Conv2D)            (None, None, None, 256)       590080
## ___________________________________________________________________________
## block3_conv3 (Conv2D)            (None, None, None, 256)       590080
## ___________________________________________________________________________
## block3_pool (MaxPooling2D)       (None, None, None, 256)       0
## ___________________________________________________________________________
## block4_conv1 (Conv2D)            (None, None, None, 512)       1180160
## ___________________________________________________________________________
## block4_conv2 (Conv2D)            (None, None, None, 512)       2359808
## ___________________________________________________________________________
## block4_conv3 (Conv2D)            (None, None, None, 512)       2359808
## ___________________________________________________________________________
## block4_pool (MaxPooling2D)       (None, None, None, 512)       0
## ___________________________________________________________________________
## block5_conv1 (Conv2D)            (None, None, None, 512)       2359808
## ___________________________________________________________________________
## block5_conv2 (Conv2D)            (None, None, None, 512)       2359808
## ___________________________________________________________________________
## block5_conv3 (Conv2D)            (None, None, None, 512)       2359808
## ___________________________________________________________________________
## block5_pool (MaxPooling2D)       (None, None, None, 512)       0
## ===========================================================================
## Total params: 14,714,688
## Trainable params: 14,714,688
## Non-trainable params: 0
## ___________________________________________________________________________
```

2. Load and pre-process the image of the zebra. (15 points).

```r
img_path <- "./zebra.jpg"
img <- image_load(img_path, target_size = c(224, 224))

img_tensor <- img %>%
  image_to_array() %>%
  array_reshape(dim = c(1, 224, 224, 3)) %>%
  `/`(255)

plot(as.raster(img_tensor[1,,,]))
```

3. Create a keras model to extract the output feature maps of the first eight layers, conv and maxpooling combined. (20 points)

```r
layer_outputs <- lapply(conv_base$layers[2:8], function(layer) layer$output)
activation_model <- keras_model(inputs = conv_base$input, outputs = layer_outputs)
```

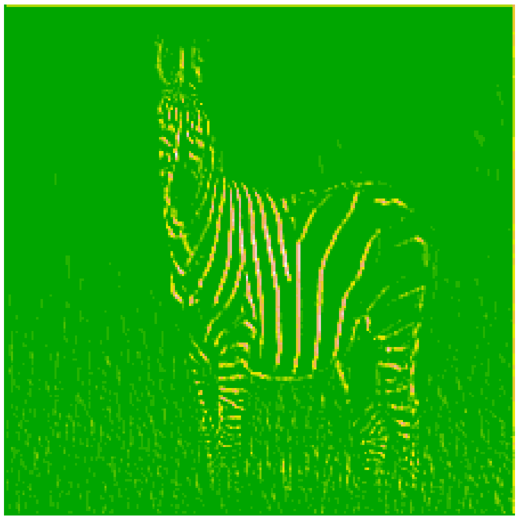
4. Run the keras model you created for the last question. (20 points)

```r
activations <- activation_model %>% predict(img_tensor)
```

5. Create a function to plot the channels (filters). (20 points)

```r
plot_channel <- function(channel) {
  rotate <- function(x) t(apply(x, 2, rev))
  image(rotate(channel), axes = FALSE, asp = 1,
        col = terrain.colors(12))
}

first_layer_activation <- activations[[1]]
plot_channel(first_layer_activation[1,,,7])
```



6. create the activations for all channels for all eight layers and save them in a directory, and display the images all together. (10 points)

```r
image_size <- 58
images_per_row <- 16
file_names <- c()

for (i in 1:7) {

  layer_activation <- activations[[i]]
  layer_name <- conv_base$layers[[i]]$name

  n_features <- dim(layer_activation)[[4]]
  n_cols <- n_features %/% images_per_row

  file_name <- paste0("zebra_activations_", i, "_", layer_name, ".png")
  file_names <- append(file_names, c(file_name))

  png(file_name,
      width = image_size * images_per_row,
      height = image_size * n_cols)
  op <- par(mfrow = c(n_cols, images_per_row), mai = rep_len(0.02, 4))

  for (col in 0:(n_cols-1)) {
    for (row in 0:(images_per_row-1)) {
      channel_image <- layer_activation[1,,,(col*images_per_row) + row + 1]
      plot_channel(channel_image)
    }
  }

  par(op)
  dev.off()
}
```

```
for(img_path in file_names) {
  paste(img_path)
  img <- image_load(img_path) %>%
    image_to_array() %>%
    `/`(255)
  paste("Image of ", img_path)
  plot(as.raster(img))
}
```