Assignment_4 Solution

1. Convolution layers

a. $\mathrm{R}:\begin{bmatrix}9 & 9\\ 9 & 9\end{bmatrix}$ $\mathrm{G}:\begin{bmatrix}18 & 18\\ 18 & 18\end{bmatrix}$ $\mathrm{B}:\begin{bmatrix}18 & 18\\ 27 & 27\end{bmatrix}$ $\xrightarrow{R+G+B}$ $\mathrm{Final}:\begin{bmatrix}45 & 45\\ 54 & 54\end{bmatrix}$

b.

$$\mathrm{R}:\begin{bmatrix}4 & 6 & 6 & 4\\ 6 & 9 & 9 & 6\\ 6 & 9 & 9 & 6\\ 4 & 6 & 6 & 4\end{bmatrix} \quad \mathrm{G}:\begin{bmatrix}8 & 12 & 12 & 8\\ 12 & 18 & 18 & 12\\ 12 & 18 & 18 & 12\\ 8 & 12 & 12 & 8\end{bmatrix} \quad \mathrm{B}:\begin{bmatrix}6 & 9 & 9 & 6\\ 12 & 18 & 18 & 12\\ 18 & 27 & 27 & 18\\ 14 & 21 & 21 & 14\end{bmatrix}$$

$$\xrightarrow{R+G+B} \mathrm{Final}:\begin{bmatrix}18 & 27 & 27 & 18\\ 30 & 45 & 45 & 30\\ 36 & 54 & 54 & 36\\ 26 & 39 & 39 & 26\end{bmatrix}$$

c.

$$\mathrm{R}:\begin{bmatrix}4 & 4 & 4 & 4\\ 4 & 4 & 4 & 4\\ 4 & 4 & 4 & 4\\ 4 & 4 & 4 & 4\end{bmatrix} \quad \mathrm{G}:\begin{bmatrix}8 & 8 & 8 & 8\\ 8 & 8 & 8 & 8\\ 8 & 8 & 8 & 8\\ 8 & 8 & 8 & 8\end{bmatrix} \quad \mathrm{B}:\begin{bmatrix}8 & 8 & 8 & 8\\ 12 & 12 & 12 & 12\\ 8 & 8 & 8 & 8\\ 12 & 12 & 12 & 12\end{bmatrix}$$

$$\xrightarrow{R+B+G} \mathrm{Final}:\begin{bmatrix}20 & 20 & 20 & 20\\ 24 & 24 & 24 & 24\\ 20 & 20 & 20 & 20\\ 24 & 24 & 24 & 24\end{bmatrix}$$

Also accept: $\mathrm{R}:\begin{bmatrix}4 & 4\\ 4 & 4\end{bmatrix}$ $\mathrm{G}:\begin{bmatrix}8 & 8\\ 8 & 8\end{bmatrix}$ $\mathrm{B}:\begin{bmatrix}12 & 12\\ 8 & 8\end{bmatrix}$ $\xrightarrow{R+B+G}$ $\mathrm{Final}:\begin{bmatrix}24 & 24\\ 20 & 20\end{bmatrix}$

d. When applying convolution, we do the dot product between the filter and the image. When the filter resembles the image, we expect a high response. The network is trying to find matches in the image.

e. When pooling between layers (or using convolution with a stride greater than 1), the spatial dimensions are sampled and so we get a pyramid with different spatial resolutions at the different layers. In this way, a fixed-size convolution filter covers a larger spatial region in deeper layers.

f.

- Increase the number of filters layers by layers.
- We decrease the spatial dimensions and increase the depth to keep the same number of coefficients. Also, it helps in learning more levels of global abstract structures and shrinking the feature space for input to the dense (fully connected) networks.

g. For output width, $W_{out} = \lfloor (W_{in} + 2P - F)/S + 1 \rfloor = 126$. Same for height, the final output shape is $126 \times 126 \times 16$

h. For output width, $W_{out} = \lfloor (W_{in} + 2P - F)/S + 1 \rfloor = 63$. Same for height, the final output shape is $63 \times 63 \times 16$

g. & h. Complete equation: Input $H_{in} \times W_{in} \times C_{in}$, convolved with C_{out} filters of size $\text{kernel\_size}\,[0] \times \text{kernel\_size}\,[1] \times C_{in}$, the output is $H_{out} \times W_{out} \times C_{out}$, where

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2 \times \text{ padding}\,[0] - \text{dilation}[0] \times (\text{ kernel\_size}\,[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor$$

$$W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2 \times \text{ padding}\,[1] - \text{dilation}[1] \times (\text{ kernel\_size}\,[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor$$

i. Given an input feature with m channels, n one-by-one filters can be applied to reduce the number of channels from m to n ($m > n$) without changing spatial dimension.

j.

- The convolutional layers are used to extract image patterns or templates.
- Difference between early and deeper convolutional layers:
  - Early layers: extract simple patterns such as edge.
  - Deeper layers: extract complex patterns.

k. $R: \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad G: \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \quad B: \begin{bmatrix} 2 & 2 \\ 4 & 4 \end{bmatrix}$

l. Downsampling the spatial dimension

m.

- The purpose of data augmentation is for better generalization and addresses the problem of limited data. It adds examples with perturbations (e.g. rotation, flip, contrast change) increasing variability in the training data which results in better generalization and prevents overfitting.
- It is most useful when the dataset is small.

---

2. CNNs

a.

- The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the related source task.
- It is most useful when the target data is similar to the source data and there is significantly more data in the source dataset.

b. Avoid weights destruction by the gradient from untrained fully connected layers.

c. The coefficients of a pre-trained network can be fine-tuned by:

1. Add customized layers on top of the pre-trained layer.
2. Freeze the pre-trained layer.
3. Train the customized layer.
4. Unfreeze the pre-trained layer.
5. Train the entire network.

d.

- The inception blocks having filters with multiple receptive fields operate on the same level which can detect different size variations in the location of the information.
- GoogleNet uses auxiliary classifiers for addressing vanishing gradients.

e.

- Advantages of residual blocks:
  - With zero weights, the network computes the identity and can learn to zero blocks to eliminate unneeded layers.
  - Identity connections provide useful feedback throughout the network.
  - While increasing network depth, it avoids negative outcomes.
- Skip connections resolve the vanishing gradient problem. When the gradient passes through layers, it will become smaller. With residual blocks, the gradient can be passed directly through the skip connection and won't be smaller.

f.

- DenseNet uses direct connections (concatenation) from any layer to all subsequent layers, allowing for feature reuse and reducing the number of parameters.
- The complexity is controlled by
  - The growth rate k (the number of channels) so that each layer only produces k output feature maps.
  - Bottleneck layers are used to reduce the number of feature maps before each 3×3 convolutional layer in dense blocks, further reducing the number of parameters.
  - Transition layers are used between each dense block to further compact the model by reducing the number of feature maps and spatial dimensions.
  - Global Average Pooling (GAP) is used to replace the fully connected layer at the end of the network, reducing the number of parameters and improving regularization.

g. Given a 4×4 image with three channels where the first has all 1s, the second has all 2s and the third has all 3s, standard convolution with a 3×3 filter having all 1s in its first layer, all 2s in its second layer and all 3s in its third layer, depth-wise convolution with a 3×3 filter having all 1s in its first layer, all 2s in its second layer and all 3s in its third layer, and point-wise convolution with a filter with all 1s in it.

- For standard convolution, the output is $\begin{bmatrix} 126 & 126 \\ 126 & 126 \end{bmatrix}$.
- For depth-wise separable convolution:
  - The output of depth-wise convolution has three channels: the first is $\begin{bmatrix} 9 & 9 \\ 9 & 9 \end{bmatrix}$, the second is $\begin{bmatrix} 36 & 36 \\ 36 & 36 \end{bmatrix}$, and the third is $\begin{bmatrix} 81 & 81 \\ 81 & 81 \end{bmatrix}$.
  - Then, the output is convolved with the point-wise convolution (standard 1×1 convolution), and the result is $\begin{bmatrix} 126 & 126 \\ 126 & 126 \end{bmatrix}$.

h. MobileNets make computation faster by:

- Depth-wise separable convolution: the depth-wise convolution reduces the number of computations required in comparison to traditional convolution layers by applying a single filter to each input channel. Then, the point-wise convolution combines the intermediate feature maps with a 1×1 filter, which greatly reduces the number of parameters.
- Width multiplier (fewer channels): reduce the number of input and output channels to each layer by a factor of $\alpha \in [0, 1]$.
- Resolution multiplier (smaller resolution): reduce the resolution by a factor $\rho \in [0, 1]$.

---

3. Object detection

a. The goal of object detection is to detect all instances of the predefined classes and provide its coarse localization in the image by axis-aligned boxes.

- Classification: classify the type of object in each bounding box. (classification problem)
- Localization: find the bounding box of different objects. (regression problem)

b.

- $\text{IoU} = \frac{|\text{pred} \cap \text{truth}|}{|\text{pred} \cup \text{truth}|} = \frac{16}{25+25-16} = \frac{16}{34}$
- $\text{Jaccard distance} = 1 - \text{IoU} = 1 - \frac{16}{34} = \frac{18}{34}$

c. r_i is recall at threshold t_i and p_i is precision at threshold t_i.

$$
\begin{aligned}
AP_{0.5} &= \sum_{\text{confidence threshold } t_i} (r_{i+1} - r_i) p_{i+1} \\
&= (0.2 - 0) \times 1 + (0.4 - 0.2) \times 0.6 + (0.6 - 0.4) \times 0.6 + (0.8 - 0) \times 0 + (1 - 0.8) \times 0 \\
&= 0.2 + 0.12 + 0.12 + 0 + 0 \\
&= 0.44
\end{aligned}
$$

d. Because we process the image at different scales, absolute coordinates cannot be used in so we normalize the box coordinates in $[0, 1]$.

e. Given the prediction $\hat{y} = [\hat{p}, \hat{x}_c, \hat{y}_c, \hat{w}, \hat{h}, \hat{c}_1, \ldots, \hat{c}_k] = [\ \hat{p}, \hat{\text{box}}, \hat{\text{class}}\ ]$ and the groundtruth label $y = [p, x_c, y_c, w, h, c_1, \ldots, c_k] = [\ p, \text{box}, \text{class}\ ]$, the general loss for object detection is $L = L_{reg} + L_{cls}$, where L_{reg} denotes the regression loss, e.g. L_2 loss, used to learn the bounding box and L_{cls} denotes the classification loss, e.g. cross-entropy, used to learn object type.

- If there is an object ($p = 1$), the loss $L = (p - \hat{p})^2 + L_{reg}(\text{box}, \hat{\text{box}}) + L_{cls}(\text{class}, \hat{\text{class}})$.
- If there is no object ($p = 0$), the loss $L = (p - \hat{p})^2$.

f. At each cell location, the output tensor shape is $10 \times (4 + 1 + K)$ for the 10 detection boxes, 4 bounding box offsets, 1 objectness prediction, and K class predictions.

g.

- Single-shot detectors perform object detection in a single network that looks at the image once.
- Two-shot detectors look at the image twice: one to propose object regions (region proposals) and a second to refine and classify regions. Two-shot detectors have a higher computational cost but may be more accurate.

h. The loss function of the YOLOv1:

$$
\begin{aligned}
L_{\text{YOLOv1}} = & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2 \right] && (1) \\
& + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2 \right] && (2) \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i\right)^2 && (3) \\
& + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i\right)^2 && (4) \\
& + \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{ classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2, && (5^*)
\end{aligned}
$$

where $\mathbb{1}_i^{\text{obj}}$ denotes if an object appears in cell i and $\mathbb{1}_{ij}^{\text{obj}}$ denotes that the jth bounding box predictor in cell i is responsible for that prediction. λ_{coord} increases the loss from bounding box coordinate predictions and λ_{noobj} decreases the loss from confidence predictions for boxes that don't contain objects.

(1) Center regression, $\mathbb{1}_{ij}^{\text{obj}} = 1$ if the jth bounding box predictor in cell i is responsible for that prediction.

(2) Height and width regression, $\mathbb{1}_{ij}^{\text{obj}} = 1$ if the jth bounding box predictor in cell i is responsible for that prediction.

(3) Object classification (Objectness loss), $\mathbb{1}_{ij}^{\text{obj}} = 1$ if the jth bounding box predictor in cell i is responsible for that prediction. The C denotes the confidence/objectness score.

(4) Non-object classification (Objectness loss), $\mathbb{1}_{ij}^{\text{noobj}} = 1$ if the jth bounding box predictor in cell i is not responsible for that prediction.

(5^*) Class prediction, $\mathbb{1}_{i}^{\text{obj}} = 1$ if an object appears in grid cell i. This term is for YOLOv1, which is different from later YOLO versions. In YOLOv1, the bounding box predictors are not responsible for class classification. Each bounding box consists of only 5 predictions: x, y, w, h, and confidence (C). The class prediction is made at the grid cell level. From YOLO9000 (YOLOv2), the class prediction is moved from the grid cell level to the bounding box level which means each bounding box consists of $5 + K$ predictions where K is the number of classes. And, this term is written as $\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \sum_{c \in \text{ classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2$.

Same as **3.e.**, if there is an object, the loss is the sum of objectness, regression, and class prediction loss, otherwise it only counts the objectness loss.

i.

- Region of Interest (RoI) pooling can be done by adjusting the window that is used to scan the input. It uses max pooling to convert the RoI to a fixed size to the Fully Connected (FC) classifier. RoI max-pooling works by dividing the $h \times w$ RoI window into an $H \times W$ grid of sub-windows of approximate size $h/H \times w/W$ and then max-pooling the values in each sub-window into the corresponding output grid cell.
- The FC classifier accepts fixed-size input, but the size of RoI may vary.

j.

- Select a detection in the grid cell with the highest detection score (probability) and delete detections that have an IoU (with the selected detection) that is greater than a threshold.
- We may detect the same object multiple times. Non-maximal suppression can be used to fix these multiple detections.

k. The general loss used in Mask R-CNN: $L = L_{cls} + L_{reg} + L_{seg}$

- Loss for classification task: $L_{cls} = L_{cls}^{RPN} + L_{cls}^{obj}$, where
  - L_{cls}^{RPN} is the (binary cross-entropy) loss for Region Proposal Network (RPN).
  - L_{cls}^{obj} is the (cross-entropy) loss for the object class prediction.
- Loss for regression task: $L_{reg} = L_{reg}^{RPN} + L_{reg}^{obj}$, where
  - L_{reg}^{RPN} is the (smooth L_1) loss for RPN.
  - L_{reg}^{obj} is the (smooth L_1) loss for the object.
  - We only compute regression loss when there is an object.
- Loss for segmentation task:
  - L_{seg} is (binary cross-entropy) loss for the segmentation.
  - We only compute segmentation loss when there is an object.

---

4. Semantic segmentation

a.

- Semantic segmentation classifies each pixel in an image into a class or object. The goal is to produce a dense pixel-wise segmentation map of an image, where each pixel is assigned to a specific class or object.
- Instance Segmentation involves identifying and separating individual objects within an image. The goal of instance segmentation is to produce a pixel-wise segmentation map of the image, where each pixel is assigned to a specific object instance.

b. Given a 5×5 image (x) and a 3×3 filter, the output (y) size is 3×3 (without padding). After vectorization, the image (x) is a $25D$ vector and the output (y) is a $9D$ vector, therefore the size of the matrix (F) is 9×25 ($\underline{y_{9\times 1}} = \underline{F_{9\times 25}}\ \underline{x_{25\times 1}}$).

c. The size of the transpose convolution matrix is 25×9.

d.

- We lose spatial resolution in the encoder which makes it hard to decode accurately. Therefore, we want to mix the features from both the encoder and decoder.
- The skip connections allow us to concatenate the encoder features with the corresponding decoder features so that the high-resolution features from the encoder can be used in the decoder to assemble a more precise output.

e. DeepLab has an encoder-decoder architecture and uses the Atrous Spatial Pyramid Pooling (ASPP) with skip connection. The convolutions are implemented as depth-wise separable convolutions.

f. The most common performance metric used in semantic segmentation is the mean IoU (mIoU) which is defined as the average IoU over all classes.