Course4-week 1

Edge detection

CNNs use edge detection as an example to explain how the convolution operation works. The early layers of a neural network can detect edges, while later layers may detect objects or complete objects like people's faces.
To detect edges in an image, a filter (3x3 matrix) is convolved with the image using the convolution operation.
The output of the convolution operation is a smaller matrix, which can be interpreted as a filtered image.
The elements of the output matrix are computed by taking element-wise products and adding them up for each position of the filter on the image.
The resulting matrix represents the detected edges in the image.
The convolution operation provides a way to specify how to find specific features, such as vertical edges, in an image. It uses a specific filter.
=>The filter highlights bright pixels on the left, disregards the middle, and detects dark pixels on the right.

Padding

Padding involves adding extra border pixels (usually zeros) around the input image before performing convolution.
cf. Without padding, the size of the output decreases after each convolutional operation, leading to a potential loss of information and extremely small output sizes in deep networks.

Formula for calculating the output size after padding: output dimension = (input dimension + 2 * padding - filter dimension + 1).
-Padding helps preserve the size of the original input and prevents excessive reduction in size during convolution.
-The two common choices for padding are "valid convolution" (no padding) and "same convolution" (padding to maintain the input size in the output).
-Odd-sized filters(보통 3x3, 5x5 사용) are preferred in computer vision due to their symmetric padding and the presence of a central pixel.

Convolutional network(example in image classification)

1. initial image size: 39 x 39 x 3 (width x height x number of channels).
2. first layer: uses 3 x 3 filters with a stride of 1 and no padding, resulting in 10 filters and an output size of 37 x 37 x 10.
3. second convolutional layer: introduced with 5 x 5 filters, a stride of 2, and no padding, resulting in 20 filters and an output size of 17 x 17 x 20.

4. third convolutional layer: using 5 x 5 filters, a stride of 2, no padding, and 40 filters, resulting in an output size of 7 x 7 x 40.

5. final step: flattening the 7 x 7 x 40 volume into a vector of 1,960 units, which is fed into a logistic regression or softmax unit for prediction.

−The choice of hyperparameters, such as the filter size, stride, padding, and number of filters, is crucial in ConvNet design.
−As the network goes deeper, the height and width of the activations tend to decrease, while the number of channels increases.
−Convolutional layers, pooling layers (also known as pool layers), and fully connected layers (FC layers) are the three common types of layers in ConvNets.
−Pooling layers and fully connected layers are simpler to define compared to convolutional layers.

Pooling layer

Convolutional Neural Networks (ConvNets) often use pooling layers in addition to convolutional layers.
Pooling layers are used to reduce the size of the representation, speed up computation, and enhance feature robustness.
Max pooling is a commonly used type of pooling. 4−It breaks the input into regions and takes the maximum value from each region to create the output.
Max pooling preserves features that are detected anywhere in the input regions.
Max pooling has hyperparameters: filter size (f) and stride (s). The filter size determines the size of the regions, and the stride determines the steps taken between regions.
Max pooling has no parameters to learn and is a fixed computation.
The output size of max pooling can be calculated using the formula n + 2p − f / s + 1, where n is the input size and p is the padding.
Max pooling can also be applied to 3D inputs, and the computation is done independently on each channel.
Average pooling is another type of pooling that takes the average value within each filter.
Max pooling is more commonly used than average pooling in neural networks, except in certain cases where average pooling may be used to reduce the representation's dimensions.
Pooling hyperparameters include filter size (f), stride (s), and an optional padding (p) parameter.
Pooling layers have no parameters to learn and act as fixed functions in the network.
The output size of a pooling layer depends on the input size, filter size, stride, and padding.
Pooling layers are often followed by fully connected layers in complex ConvNet architectures.

Course4-week 2

>pdf 파일

Course4-week 3

Object Localization

Object detection includes object localization and recognition.
Object localization: labeling an object in an image + drawing a bounding box around object
Classification with localization: neural network is trained to output both the object class label and the bounding box coordinates.
>The target label for classification with localization

1. the probability of an object's presence (pc)->object가 존재할 확률
2. the bounding box coordinates (bx, by, bh, bw)
3. the class probabilities (c1, c2, c3)->최대 1개가 1

Landmark Detection

face recognition, pose detection 등의 작업에 쓰임.
Landmarks in face recognition: specific points like corners of the eyes etc
Adding output units to a neural network enables it to output the X and Y coordinates of different landmarks.
>Landmark detection 을 응용하여 object detection 실행 가능

object detection

>sliding windows detection

-use an even larger window each time

-go through every region with the same window size and have it classified for 0 or 1

-stride 조절해서 속도 높임, 이는 performance를 손상시킬 수 있음

Convolutional implementation of sliding windows

>turning FC layer into convolutional layers

- The implementation of fully connected layers as convolutional layers involves using filters with the same size as the fully connected layer.

- The algorithm still has the weakness of inaccurate bounding box positions

Bounding box prediction

>yolo algorithm

- The YOLO (You Only Look Once) algorithm is introduced as a way to improve the accuracy of bounding box predictions in object detection.

-The image is divided into a grid, with each grid cell being a target for classification and localization.
-For each grid cell, an 8-dimensional label vector is defined, including indicators for the presence of an object, bounding box coordinates, and class probabilities.
-The YOLO algorithm assigns objects to grid cells based on the midpoint of each object.

Non max suppression

Non-max suppression is a technique used to ensure that each object is detected only once

works by considering the probabilities associated with each detection and selecting the one with the highest confidence as the most confident detection.

discards low-probability bounding boxes and repeatedly selects the box with the highest probability as a prediction.

semantic segmentation algorithm

- figure out what is every single pixel in the image rather than drawing bounding boxes it attempts to label every single pixel

application

-motivation for U-Net

: segment out an image exactly to certain parts of the body like lungs, heart etc

this can make it easier to daignose diseases or surgeons when doing surgerys

brain tumor detection: segment out tumor automatically

-segmenting out the car

two labels: 1 for car 0 for no car

output either 0 or 1 for every single pixel

more labels for more objects to detect

neural network architecture for new output of a matrix of class labels

:modify algorithm of object detection->get rid of the last few layers + dimension needs to get bigger in the last layers

as you go deeper the height and width will increase in the end in u-net(차원 작아졌다가 커짐)

key point) take a small set of activations and to blow it up to a bigger set of activations(use transpose convolution)

transpose convolution(key building block of u-net architecture)

:blows up dimension of input

ex)2*2 input -> 3*3 filter->4*4 output

apply 1 padding to output

place filter on the output instead of input

filter is multiplyed by each element of input(in this example multiplied 4 times)

ignore padding region and apply to output

where boxes overlap you add the outputs

u-net architecture

first part: normal convolutions(compress image,)

second part: transpose convolution

skip connections: earlier block of activations are copied directly to later activations->allows the neural network to take this very high resolution, low level feature information where it could capture for every pixel position and used to skip connection to pause that directly to this later layer. And so this way this layer has both the lower resolution, but high level, spatial, high level contextual information, as well as the low level.

u-net details

input image: h*w*3->to be simple only take height and num of channels(3)

normal convolution(relu, max pool) layers->transpose convolution layers(decrease number of channels)