Convolutional Neural Networks

dzlabs

September 2018

1 Week 1 - Foundations of Convolutional Neural Networks

Learn to implement the foundational layers of CNNs (pooling, convolutions) and to stack them properly in a deep network to solve multi-class image classification problems.

1.1 Convolutional Neural Networks

1.1.1 Computer Vision

Example applications: Image Classification, Object detection, Neural Style Transfer. High resolution pictures (e.g. 1000x1000px) will cause the NN to be too large as you will have a large input. To handle so much features, you need to better implement the **convolution operation**, which is one of the fundamental building blocks of convolutional neural networks.

1.1.2 Edge Detection Example

Matrix Convolutional operation: take one big matrix (e.g. A: 6x6) and multiply it by the **filter** or **mask** matrix (e.g. B: 3x3) to get a resulting matrix of (C: 4x4). First cell of the resulting matrix is:

$$\begin{aligned} C[1,1] = A[1,1] * B[1,1] + A[2,1] * B[2,1] + A[3,1] * B[3,1] \\ + A[1,2] * B[1,2] + A[2,2] * B[2,2] + A[3,2] * B[3,2] \\ + A[1,3] * B[1,3] + A[2,3] * B[2,3] + A[3,3] * B[3,3] \end{aligned} \quad (1)$$

Then move the small matrix inside A to the right and down until all cells of C are populated.

1.1.3 More Edge Detection

Learning to detect edges, treat the 9 numbers of the filter matrix (i.e. 3x3) as parameters that the Backpropagation will learn them.

1.1.4 Padding

If A is (n, n) B is (f, f), then the convolution product is (n-f+1, n-f+1). Also the cell in the middle of A is been used many times as it overlaps many matrices, on the other hand the top left cell is used only by one matrix, information is been lost. Problems with convolutions include:

- shrinking output
- throw away info from edge

A solution is padding or adding more cells in the corners of A. To decide how much to convolute, the are two techniques:

- Valid Convolution: no padding. (n, n) * (f, f) = (n-f+1, n-f+1)
- Same Convolution: Pad so that output size is the same as the input size.

$$n + 2p - f + 1 = n => p = \frac{f-1}{2} \quad (2)$$

In computer vision, f is usually an `odd` number.

1.1.5 Strided Convolutions

Instead of moving the filter in A to the right or down with 1 cell, use a `stride` value (e.g. $stride = 2$).

In mathematical literature, convolution is called corss-correlation.

1.1.6 Convolutions Over Volume

Images have many channels, like Red, Green and Blue, the filter/mask matrix will also have same number of channels, however the resulting matrix will have only one channel. E.g. a matrix of 6 x 6 x 3 convoluted by 3 x 3 x 3 will result in 4 x 4 x 1.

What if you wanted to use multiple filters at a time? you will just convolute every filter against the A matrix and you have as much result as filters.

Summay: (n, n, nc) * (f, f, nc) = (n-f+1, n-f+1, nc'). Where nc is number of channels, and nc' is the number of filters.

1.1.7 One Layer of a Convolutional Network

Example of building a Convolutional Neural Network: the input matrix is nothing by $A^{[i]}$ and the filter matrix becomes $W^{[i]}$. In the following equation there is a convolution product between w and a

$$z^{[1]} = w^{[1]} * a^{[1]} + b^{[1]} \tag{3}$$

$$a^{[1]} = g(z^{[1]}) \tag{4}$$

`Numbers of parameters in one layer?` If you have 10 filters that are 3x3x3 in one layer of a neural network, how many paramters does that layer have? 3x3x3 filter is 27 parameters, plus the bias, that gives 28 parameters, summing up we will have 280 parameters for the ten filters.

The number of parameters in a CNN is fixed which makes this network not overfitting.

Notation if layer l is a convolution layer:

- $f^{[l]}$ filter size
- $p^{[l]}$ padding
- $s^{[l]}$ stride
- $n_c^{[l]}$ number of filters
- Each filter is $f^{[l]}$ x $f^{[l]}$ x $n_c^{[l-1]}$
- Activations: $a^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$ $A^{[l]}$ =¿ M x $n_h^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$
- Weights: $f^{[l]}$ x $f^{[l]}$ x $n_c^{[l-1]}$ x $n_c^{[l]}$
- bias: $n_c^{[l]}$ - (1, 1, 1, $n_c^{[l]}$)
- input: $n_h^{[l-1]}$ =¿ $n_h^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$
- Output: $n_h^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$, $n_h^{[l]} = \left\lfloor \frac{n_h^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$

1.1.8 Simple Convolutional Network Example

Types of layer in a convolutional network

- Convolution (CONV)
- Pooling (POOL)
- Fully connected (FC)

1.1.9 Pooling Layers

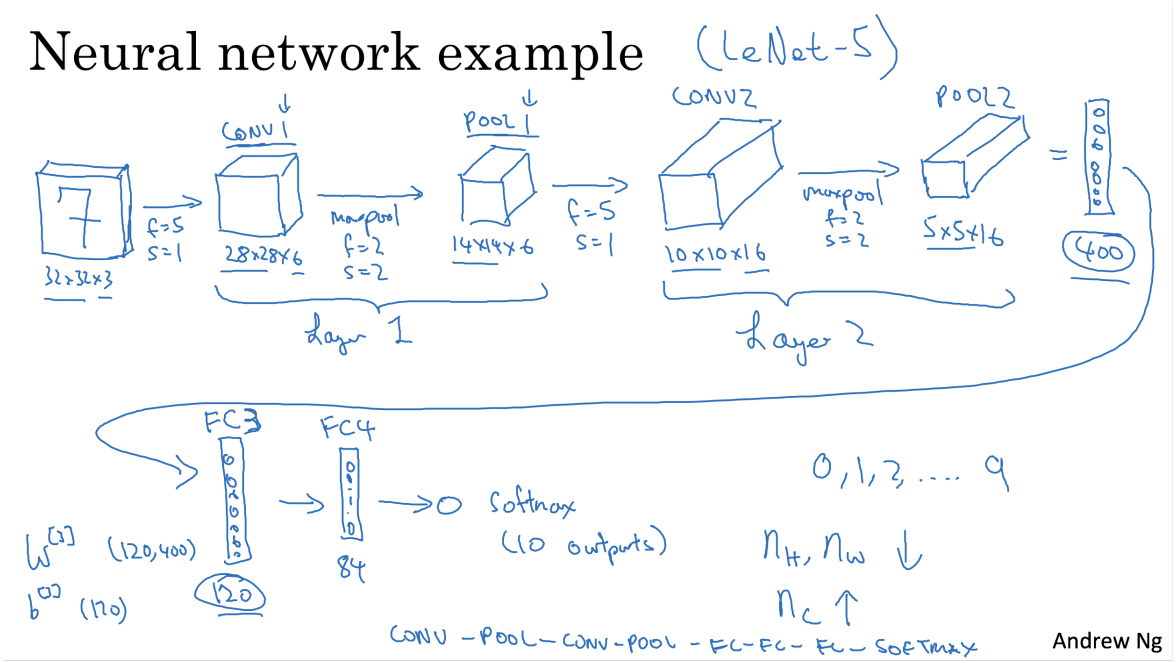
Types of pooling:

Max pooling uses the MAX function when aggregating cells

Avg pooling uses the AVG function when aggregating cells

Ex- Max pooling: for a 4x4 matrix into a 2x2 matrix, in the original matrix divide it into 2x2 regions then take the max in each of those regions which you will pass to the 2x2 output matrix. Max pooling has hyperparameters f and s but no parameters for gradient to learn.

1.1.10 CNN Example



	Activation shape	Activation Size	# parameters
Input	(32, 32, 3)	3 072	0
CONV1 (f=5, s=1)	(28, 28, 8)	6 272	208
POOL1	(14, 14, 8)	1 568	0
CONV2 (f=5, s=1)	(10, 10, 16)	1 600	416
POOL2	(5, 5, 16)	400	0
FC3	(120, 1)	120	48 001
FC4	(84, 1)	84	10 081
Softmax	(10, 1)	10	841

The number of parameters in the convolutional layer is relatively small, it goes up in the Fully Connected layers. The size of the activation matrix goes down as we go deeper in the NN.

1.1.11 Why Convolutions?

Parameter sharing: A feature detector (such as a vertical edge detector) that's useful in one part of the image is probably useful in another part of the image.

Sparsity of connections: In each layer, each output value depends only on a small number of inputs.

1.2 Practice Questions

QUIZ - The basics of ConvNets

1. What do you think applying this filter to a grayscale image will do?

- Detect horizontal edges
- Detect 45 degree edges (X 1)
- Detect image contrast (X 2)
- Detect vertical edges (X 3)

2. Suppose your input is a 300 by 300 color (RGB) image, and you are not using a convolutional network. If the first hidden layer has 100 neurons, each one fully connected to the input, how many parameters does this hidden layer have (including the bias parameters)?

- 9,000,001

- 9,000,100
- 27,000,001
- 27,000,100 (X)

3. Suppose your input is a 300 by 300 color (RGB) image, and you use a convolutional layer with 100 filters that are each 5x5. How many parameters does this hidden layer have (including the bias parameters)?

- 2501 (X 2) (5x5)*100+1=2501
- 2600 (X 1) (5x5+1)*100=2600
- 7500
- 7600

4. You have an input volume that is 63x63x16, and convolve it with 32 filters that are each 7x7, using a stride of 2 and no padding. What is the output volume?

- 16x16x32
- 29x29x16
- 29x29x32 (X) $\lfloor \frac{63+2p-f}{s} + 1 \rfloor = \lfloor \frac{63+0-7}{2} + 1 \rfloor = 29$
- 16x16x16

5. You have an input volume that is 15x15x8, and pad it using "pad=2." What is the dimension of the resulting volume (after padding)?

- 19x19x12
- 17x17x8
- 17x17x10
- 19x19x8 (X) $= 15 + 2p = 19$

6. You have an input volume that is 63x63x16, and convolve it with 32 filters that are each 7x7, and stride of 1. You want to use a "same" convolution. What is the padding?

- 1
- 2
- 3 (X) $63 = \lfloor \frac{63+2p-f}{s} + 1 \rfloor = \lfloor \frac{63+0-7}{1} + 1 \rfloor => 0 = 2p - 6 => p = 3$
- 7

7. You have an input volume that is 32x32x16, and apply max pooling with a stride of 2 and a filter size of 2. What is the output volume?

- 32x32x8
- 16x16x8
- 16x16x16 (X) Max pool divide by 2 the original size
- 15x15x16

8. Because pooling layers do not have parameters, they do not affect the backpropagation (derivatives) calculation.

- True (X 1)
- False (X 2)

9. In lecture we talked about "parameter sharing" as a benefit of using convolutional networks. Which of the following statements about parameter sharing in ConvNets are true? (Check all that apply.)

- It allows a feature detector to be used in multiple locations throughout the whole input image/input volume. (X 1 2)
- It allows parameters learned for one task to be shared even for a different task (transfer learning). (X 1)
- It allows gradient descent to set many of the parameters to zero, thus making the connections sparse.
- It reduces the total number of parameters, thus reducing overfitting. (X 2)

10. In lecture we talked about "sparsity of connections" as a benefit of using convolutional layers. What does this mean?

- Regularization causes gradient descent to set many of the parameters to zero.
- Each layer in a convolutional network is connected only to two other layers
- Each activation in the next layer depends on only a small number of activations from the previous layer. (X)
- Each filter is connected to every channel in the previous layer.

2 Week 2 - Deep convolutional models: case studies

Learn about the practical tricks and methods used in deep CNNs straight from the research papers.

2.1 Case studies

2.1.1 Why look at case studies?

It turns out a lot of the past few years of computer vision research has been on how to put together these basic building blocks to form effective convolutional neural networks. Classic networks:

- LeNet-5
- AlexNet
- VGG

Other NN: ResNet (152) layers, Inception network

2.1.2 Classic Networks

History of Deep learning networks.

2.1.3 ResNet

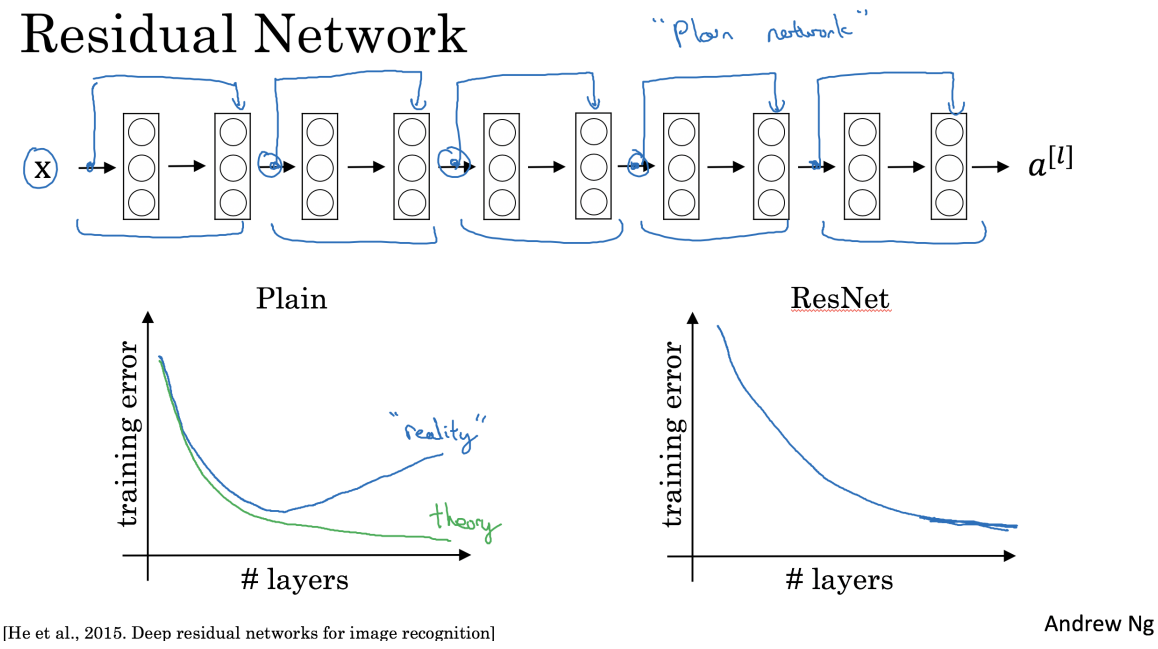
Residual Networks: are based on Residual Blocks where the activation of a layer is used again further deep in the network (creating **Shortcut / Skip connections**):

$z^{[l+1]} = W^{[l+1]}a^{[l]} + b^{[l+1]} => a^{[l+1]} = g(z^{[l+1]}) => z^{[l+2]} = W^{[l+2]}a^{[l+1]} + b^{[l+2]} => a^{[l+2]} = g(z^{[l+2]})$

Then reuse $a^{[l]}$ again in $a^{[l+2]}$, so the later becomes:

$$a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$$

In theory as we increase the number of hidden layers the NN does better and better. However, in practice it does well until a certain number of hidden NN after that the performance goes down. **ResNet** allows you build deep NN (e.g. thousands of hidden layers) while keep improving the performance.



[He et al., 2015. Deep residual networks for image recognition]

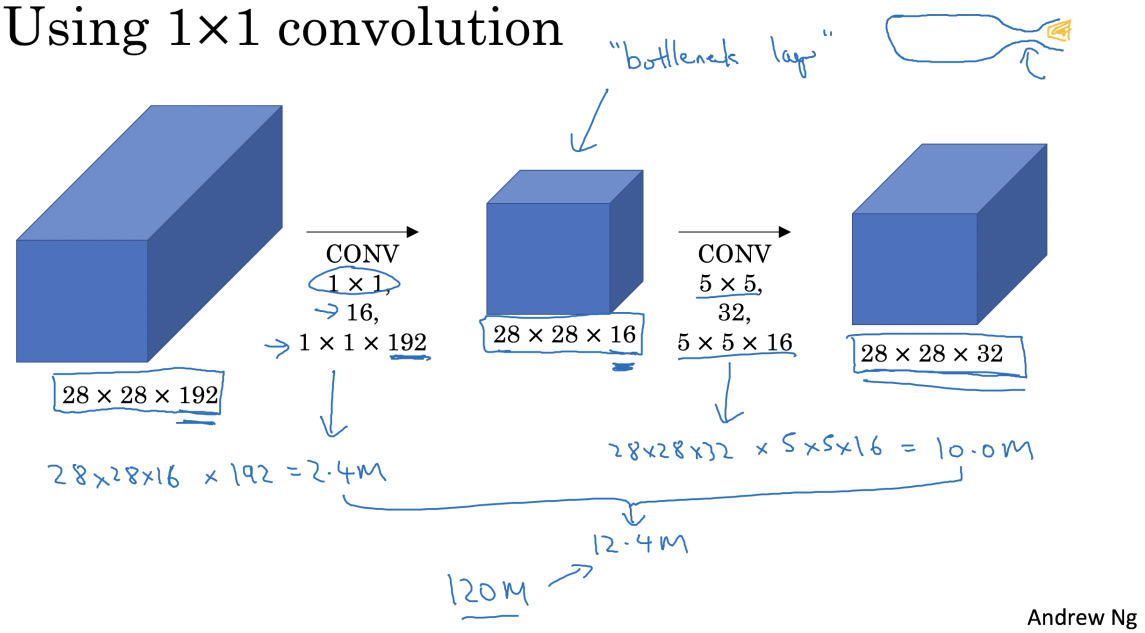
2.1.4 Why ResNets Work

2.1.5 Networks in Networks and 1x1 Convolutions

It is basically multiplying the input matrix by 2, this is in case of one channel. But in case of multiple channels, it's useful to shrunk the number of channels. 1x1 convolution is also called network in network.

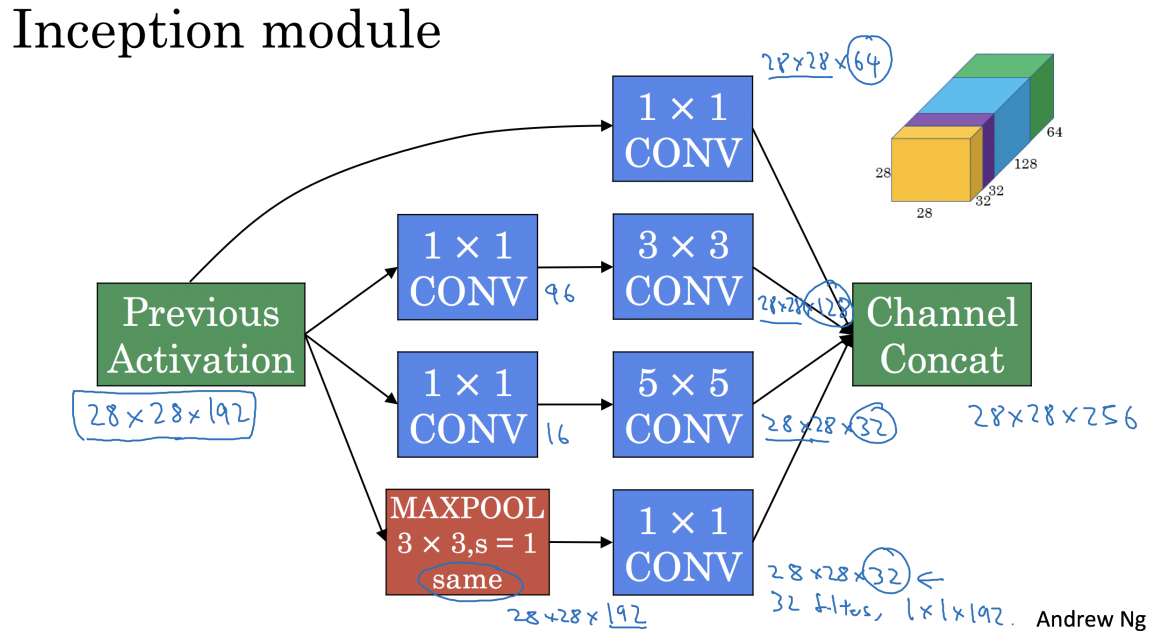
2.1.6 Inception Network Motivation

Reducing the number of channels with a **Bottleneck** layer which performs a 1x1 convolution:



2.1.7 Inception Network

Similar to ResNet, Inception Net are based on Inception block that are chained together to create a complete net. It also contains side branches to predict the output directly from the the hidden / intermediate networks.



2.2 Practical advices for using ConvNets

2.2.1 Using Open-Source Implementation

Pratcial advices on using the NN seeing in the previous videos. If you're developing a computer vision application, a very common workflow would be to:

- pick an architecture that you like, (e.g. from those learned in this course) or one that you heard about from a friend or from some literature.
- And look for an open source implementation and download it from GitHub to start building from there.

One of the advantages of doing so also is that sometimes these networks take a long time to train (e.g. multiple GPUs, very large dataset to pretrain). And that allows you to do transfer learning using these networks.

2.2.2 Transfer Learning

When building a computer vision application, rather than training the ways from scratch, from random initialization, you often make much faster progress if you download ways that someone else has already trained on the network architecture and use that as pre-training and transfer that to a new task that you might be interested in.

You can often download open source ways that took someone else many weeks or months to figure out and use it to initialize your NN.

2.2.3 Data Augmentation

Common augmentation method:

- Mirroring: e.g. mirror a picture on the vertical axes
- Random cropping: unless the cropped picture preserve properties of the original, e.g. Cropping the background in a cat picture is not a cat.
- color shifting: motivation if the sunlight is yellow or illumination, e.g. color distortion: changing red by green.
- Advanced color augmentation: PCA color augmentation: (in AlexNet paper)

Implementing distortions during training.

2.2.4 State of Computer Vision

Tips for doing well on benchmarks/winning competitions (not for production systems):

- **Ensembling:** Train serval networks (e.g. 3 to 15 NN) independently and average their outputs (i.e. average $\hat{y}$ not the weights)
- **Multi-crop at test time:** Run classifier on multiple vertions of test images and average results

Use open source code:

- Use architectures of networks published in the literature
- Use open source implementations if possible
- Use pretrained models and fine-tune on your dataset

2.3 Practice questions

2.3.1 QUIZ - Deep convolutional models

1. Which of the following do you typically see as you move to deeper layers in a ConvNet?

- n_H and n_W increases, while n_C also increases
- n_H and n_W decrease, while n_C increases (X)
- n_H and n_W increases, while n_C decreases
- n_H and n_W decreases, while n_C also decreases

2. Which of the following do you typically see in a ConvNet? (Check all that apply.)

- Multiple CONV layers followed by a POOL layer (X)
- Multiple POOL layers followed by a CONV layer
- FC layers in the last few layers (X)

- FC layers in the first few layers

3. In order to be able to build very deep networks, we usually only use pooling layers to downsize the height/width of the activation volumes while convolutions are used with "valid" padding. Otherwise, we would downsize the input of the model too quickly.

- True (X 1)
- False (X 2)

4. Training a deeper network (for example, adding additional layers to the network) allows the network to fit more complex functions and thus almost always results in lower training error. For this question, assume we're referring to "plain" networks.

- True
- False (X)

5. The following equation captures the computation in a ResNet block. What goes into the two blanks above?

$$a^{[l+2]} = g(W^{[l+2]}g(W^{[l+1]})a^{[l]} + b^{[l+1]}) + b^{l+2} + \text{-------}) + \text{-------}$$

- 0 and $a^{[l]}$, respectively
- $a^{[l]}$ and 0, respectively (X)
- 0 and $z^{[l+1]}$, respectively
- $z^{[l]}$ and $a^{[l]}$, respectively

6. Which ones of the following statements on Residual Networks are true? (Check all that apply.)

- The skip-connection makes it easy for the network to learn an identity mapping between the input and the output within the ResNet block. (X)
- The skip-connections compute a complex non-linear function of the input to pass to a deeper layer in the network.
- A ResNet with L layers would have on the order of L^2 skip connections in total.
- Using a skip-connection helps the gradient to backpropagate and thus helps you to train deeper networks (X)

7. Suppose you have an input volume of dimension 64x64x16. How many parameters would a single 1x1 convolutional filter have (including the bias)?

- 2
- 1
- 4097
- 17 (X)

8. Suppose you have an input volume of dimension n_H x n_W x n_C . Which of the following statements you agree with? (Assume that "1x1 convolutional layer" below always uses a stride of 1 and no padding.)

- You can use a 1x1 convolutional layer to reduce n_C but not n_H, n_W. (X)
- You can use a 1x1 convolutional layer to reduce n_H, n_W, and n_C.
- You can use a pooling layer to reduce n_H, n_W, but not n_C. (X)
- You can use a pooling layer to reduce n_H, n_W, and n_C.

9. Which ones of the following statements on Inception Networks are true? (Check all that apply.)

- Inception networks incorporates a variety of network architectures (similar to dropout, which randomly chooses a network architecture on each step) and thus has a similar regularizing effect as dropout.
- Inception blocks usually use 1x1 convolutions to reduce the input data volume's size before applying 3x3 and 5x5 convolutions. (X)
- A single inception block allows the network to use a combination of 1x1, 3x3, 5x5 convolutions and pooling. (X)
- Making an inception network deeper (by stacking more inception blocks together) should not hurt training set performance. (X)

10. Which of the following are common reasons for using open-source implementations of ConvNets (both the model and/or weights)? Check all that apply.

- Parameters trained for one computer vision task are often useful as pretraining for other computer vision tasks. (X)
- The same techniques for winning computer vision competitions, such as using multiple crops at test time, are widely used in practical deployments (or production system deployments) of ConvNets.
- It is a convenient way to get working an implementation of a complex ConvNet architecture. (X)
- A model trained for one computer vision task can usually be used to perform data augmentation even for a different computer vision task.

3 Week 3 - Object detection

Learn how to apply your knowledge of CNNs to one of the toughest but hottest field of computer vision: Object detection.

3.1 Detection algorithms

3.1.1 Object Localization

Defining the target label y

1. pedestrian
2. car
3. motorsycle
4. background

Need to output the bounding box for the detected object (center point of the object b_x, b_y, height b_h, width b_w) and a class label (1-4)

y is a vector that starts with p_c whether there is an object in the image or not. It's a supervisod learning algorithm, your labeled data should contain also the boxes.

$$\mathcal{L}(\hat{y}, y) = \begin{cases} (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + ... + (\hat{y}_8 - y_8)^2 if y_1 = 1 \\ (\hat{y}_1 - y_1)^2 if y_1 = 0 \end{cases}$$

3.1.2 Landmark Detection

Example detect emotion from detecting landmarks (e.g. eyes, nose) in a person's face. Detecting pose of a person, determine the key landmarks: arm, chase, etc.

3.1.3 Object Detection

Car detection example: crop pictures of cars to build a labeled dataset that you can train with a ConvNet to detect this object (i.e. car). With **Sliding Window** algorithm, you can take a region and move it on your original picture then feed this region to your ConvNet to detect a car there.

A disadvantage of the sliding window is it's high computation cost as you try different sizes of region and having to move these regions across the picture.

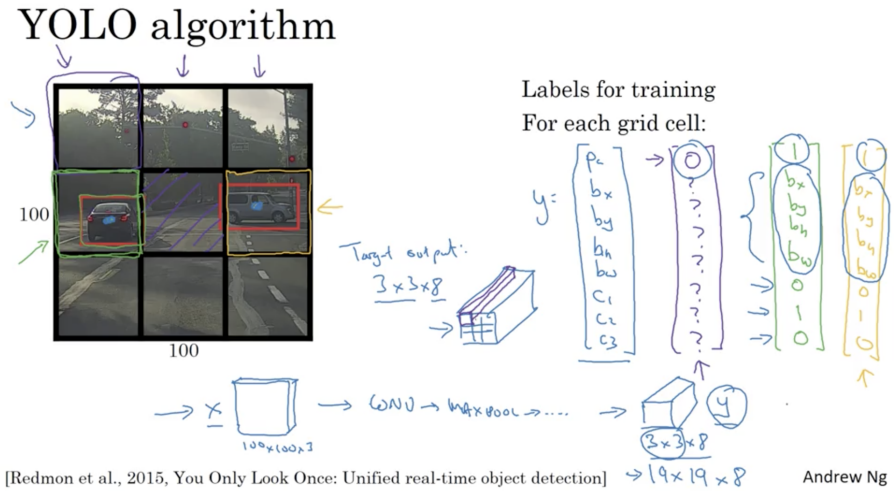
3.1.4 Convolutional Implementation of Sliding Windows

Bsically, this algorithm turns the fully connected layers into convolution layers then pass the whole image and result is a cell of every possible region you would have cropped with the previous algorithm.

The convolution implementation is it combines all four into one form of computation and shares a lot of the computation in the regions of image that are common, instead of instead of forcing you to run four propagation on four subsets of the input image independently.

One problem of this algorithm is that the resulting bounding boxes are not accurate.

3.1.5 Bounding Box Predictions



3.1.6 Intersection Over Union

Function for evaluating a detection algorithm. IOU computes the intersection of the bounding boxes divided by the size of the union of both boxes (the real one, and the predicted one).

$$IOU = \frac{\text{size of intersection}}{\text{size of union}}$$

Correct if $IOU \geq 0.5$

3.1.7 Non-max Suppression

It's way to make sure an object is detected by your algorithm only once. This by taking the max of the p_c probabilities of the bounding boxes.

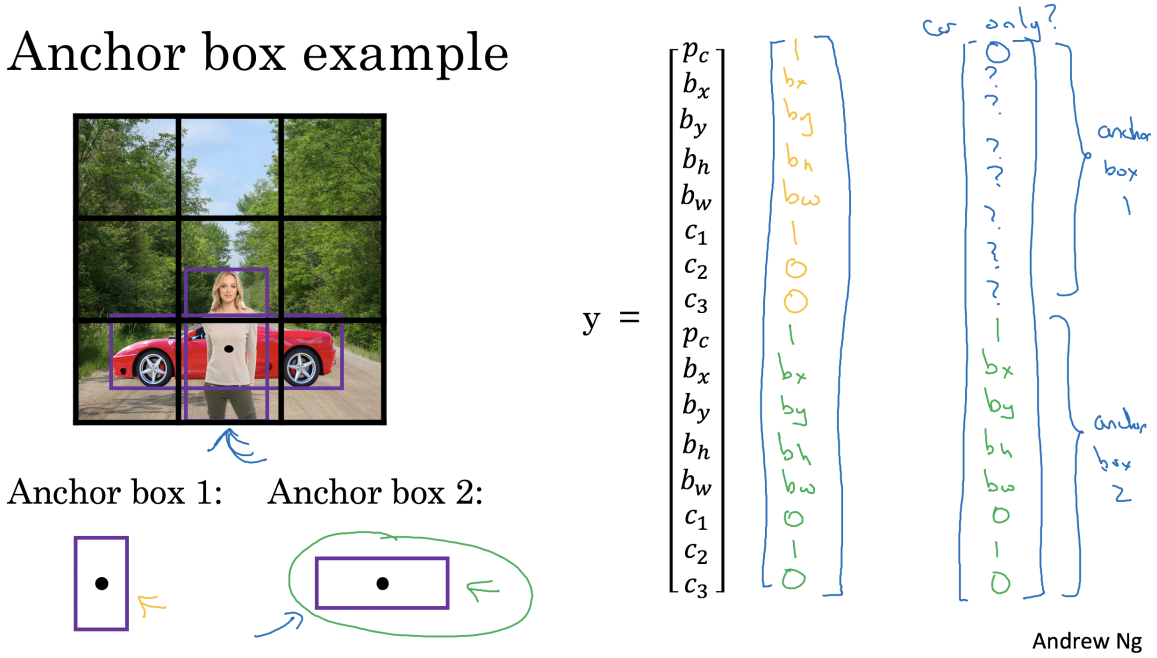
Algorithm:

- divide the input picutre in a grid of 19x19 cells
- the output will be 19x19x8, where each cell is 8 length vector (see above)
- discard all boxes with probablity $p_c \leq 0.6$
- while there are any remaining boxes:
  - Pick the box with the largest p_c, output that as a prediction
  - Discard any remaining box with $IOU \geq 0.5$ with the box output in the previous step.

3.1.8 Anchor Boxes

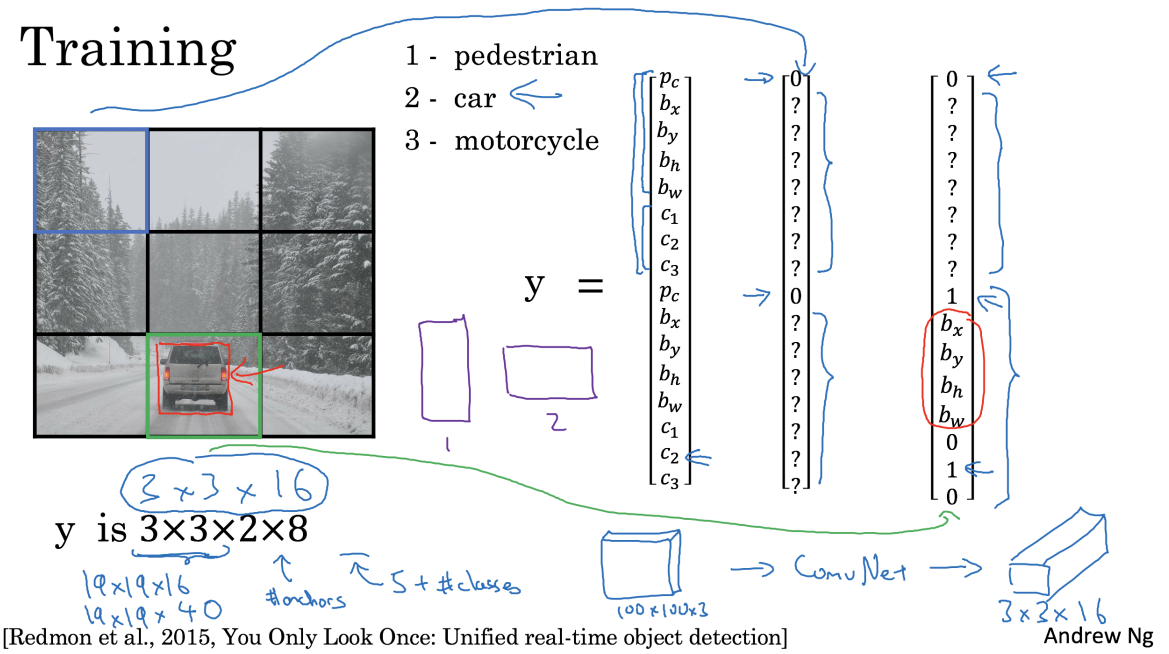
One of the problem object detection algorithm is that if a pedestrain and a car has same center, then the algorithm may pick one object over the other and not both. To avoid this use anchors of different shapes.

- **Previously:** Each object in training image is assigned to grid cell that contains that object's midpoint.
- **With two anchor boxes:** Each object in training image is assigned to grid cell that contains object's midpoint and anchor box for the grid cell with highest IoU.



This algorithm allows your learning algorithm to specialize better, in detecting fat, skinny objects.

3.1.9 YOLO Algorithm



[Redmon et al., 2015, You Only Look Once: Unified real-time object detection]

Outputting the non-max supressed outputs

- For each grid call, get 2 predicted bounding boxes.
- Get rid of low probability predictions.
- For each class (pedestrian, car, motorcycle) use non-max suppression to generate final predictions.

3.1.10 (Optional) Region Proposals

Run a segmentation algorithm to detect what could be an object, blobs (e.g. 2000), then run ConvNet on those regions.

The based Rregion CNN is quite slow, alternative Faster algorithms

- **R-CNN:** Propose regions. Classify proposed regions one at a time. Output label + bounding box.

- **Fast R-CNN:** Propose regions. Use convolution implementation of sliding windows to classify all the proposed regions.
- **Faster R-CNN:** Use convolutional network to propose regions.

3.2 Practice questions

3.2.1 QUIZ - Detection algorithms

1. You are building a 3-class object classification and localization algorithm. The classes are: pedestrian (c=1), car (c=2), motorcycle (c=3). What would be the label for the following image? Recall $y = [p_c, b_x, b_y, b_h, b_w, c_1, c_2, c_3]$.

- $y = [1, 0.3, 0.7, 0.3, 0.3, 0, 1, 0]$ (X)
- $y = [1, 0.7, 0.5, 0.3, 0.3, 0, 1, 0]$
- $y = [1, 0.3, 0.7, 0.5, 0.5, 0, 1, 0]$
- $y = [1, 0.3, 0.7, 0.5, 0.5, 1, 0, 0]$
- $y = [0, 0.2, 0.4, 0.5, 0.5, 0, 1, 0]$

2. Continuing from the previous problem, what should y be for the image below? Remember that "?" means "don't care", which means that the neural network loss function won't care what the neural network gives for that component of the output. As before, $y = [p_c, b_x, b_y, b_h, b_w, c_1, c_2, c_3]$.

- $y = [1, ?, ?, ?, ?, 0, 0, 0]$
- $y = [0, ?, ?, ?, ?, 0, 0, 0]$
- $y = [1, ?, ?, ?, ?, ?, ?, ?]$
- $y = [?, ?, ?, ?, ?, ?, ?, ?]$
- $y = [0, ?, ?, ?, ?, ?, ?, ?]$ (X)

3. You are working on a factory automation task. Your system will see a can of soft-drink coming down a conveyor belt, and you want it to take a picture and decide whether (i) there is a soft-drink can in the image, and if so (ii) its bounding box. Since the soft-drink can is round, the bounding box is always square, and the soft drink can always appears as the same size in the image. There is at most one soft drink can in each image. Here're some typical images in your training set:

What is the most appropriate set of output units for your neural network?

- Logistic unit (for classifying if there is a soft-drink can in the image)
- Logistic unit, b_x and b_y
- Logistic unit, b_x, b_y, b_h (since $b_w = b_h$) (X 1)
- Logistic unit, b_x, b_y, b_h, b_w (X 2)

4. If you build a neural network that inputs a picture of a person's face and outputs N landmarks on the face (assume the input image always contains exactly one face), how many output units will the network have?

- N
- 2N (X)
- 3N
- N^2

5. When training one of the object detection systems described in lecture, you need a training set that contains many pictures of the object(s) you wish to detect. However, bounding boxes do not need to be provided in the training set, since the algorithm can learn to detect the objects by itself.

- True (X 1)

- False (X 2)

6. Suppose you are applying a sliding windows classifier (non-convolutional implementation). Increasing the stride would tend to increase accuracy, but decrease computational cost.

- True
- False (X)

7. In the YOLO algorithm, at training time, only one cell —the one containing the center/midpoint of an object— is responsible for detecting this object.

- True (X)
- False

8. What is the IoU between these two boxes? The upper-left box is 2x2, and the lower-right box is 2x3. The overlapping region is 1x1.

- 1/6
- 1/9 (X 2) $\frac{1}{1-1+2*2+2*3}$
- 1/10 (X 1)
- None of the above

9. Suppose you run non-max suppression on the predicted boxes above. The parameters you use for non-max suppression are that boxes with probability ≤ 0.4 are discarded, and the IoU threshold for deciding if two boxes overlap is 0.5. How many boxes will remain after non-max suppression?

- 3
- 4 (X 1)
- 5 (X 2)
- 6
- 7

10. Suppose you are using YOLO on a 19x19 grid, on a detection problem with 20 classes, and with 5 anchor boxes. During training, for each image you will need to construct an output volume yy as the target value for the neural network; this corresponds to the last layer of the neural network. (yy may include some "?", or "don't cares"). What is the dimension of this output volume?

- 19x19x(5x25) (X 2)
- 19x19x(20x25)
- 19x19x(5x20) (X 1)
- 19x19x(25x20)

4 Week 4 - Special applications: Face recognition & Neural style transfer

Discover how CNNs can be applied to multiple fields, including art generation and face recognition. Implement your own algorithm to generate art and recognize faces!

4.1 Face Recognition

Face verification vs. face recognition

- **Verification**
  - Input image, name/ID
  - Output whether the input image is that of the claimed person
- **Recognition**
  - Has a database of K persons
  - Get an input image
  - Output ID if the image is any of the K persons (or "not recognized")

4.1.1 One Shot Learning

One-shot learning problem; For most face recognition applications you need to be able to recognize a person given just one single image, or given just one example of that person's face.

The problen is you don't have enough dataset to get a CNN working, Learning from one example to recognize the person again.

Learning a "similarity" function wiht a NN $d(img1, img2)$ = degree of difference between images for verification

$$\text{If } d(img1, img2) \begin{cases} \leq \pi \text{ same} \\ \geq \pi \text{ different} \end{cases}$$

4.1.2 Siamese Network

Goal of learning Parameters of NN define an encoding $f(x^{(i)})$ (a vector of sie 128). Learn the parameters so that:

- if $x^{(i)}$, $x^{(j)}$ are the same person, $\left|\left|f(x^{(i)}) - f(x^{(j)})\right|\right|^2$ is small
- if $x^{(i)}$, $x^{(j)}$ are different persons, $\left|\left|f(x^{(i)}) - f(x^{(j)})\right|\right|^2$ is large

4.1.3 Triplet Loss

Learning Objective:

- A: anchor
- P: positive
- N: negative
- $d(A, P) = ||f(A) - f(P)||^2$
- α is a margin used to enusre the NN will not chose trivial solutions, e.g. $f(img) = 0$ or any constant.

We want:

$$||f(A) - f(P)||^2 + \alpha \leq ||f(A) - f(N)||^2 \tag{5}$$

$$||f(A) - f(P)||^2 + \alpha - ||f(A) - f(N)||^2 \leq 0 \tag{6}$$

Loss function: Given 3 images A, P (same person as Anchor), N (different person than Anchor)

$$\mathcal{L}(A, P, N) = max(||f(A) - f(P)||^2 + \alpha - ||f(A) - f(N)||^2, 0) \tag{7}$$

So that the NN does not care how is the first part of max is negative.

$$J = \sum_{i=1}^{m} \mathcal{L}(A^{(i)}, P^{(i)}, N^{(i)}) \tag{8}$$

Training set: 10k pictures of 1k persons. If you have only one picture of this person then you cannot use this system.

Choosing the triplets A, P, N

During training, if A, P, N are chosen randomly, $d(A, P) + \alpha \leq d(A, N)$ is easily satisfied.

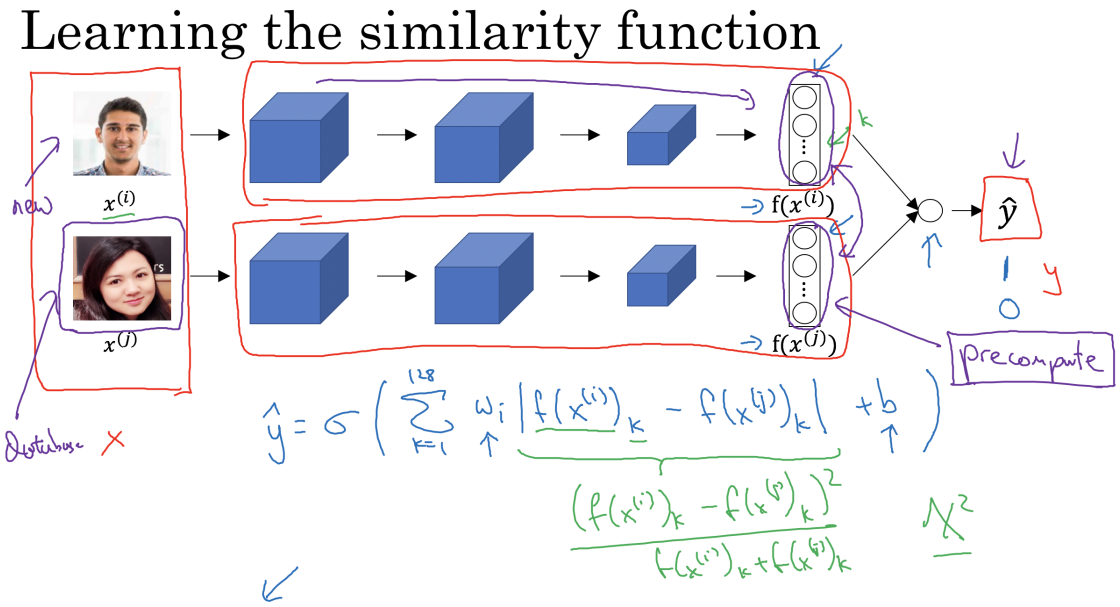
Choose triplets that're hard to train on, so that training algorith try hard to separate the two distances and avoid $d(A, P) \approx d(A, N)$

4.1.4 Face Verification and Binary Classification

Learning the similarity function

$$\hat{y} = \sigma(\sum_{k=1}^{128} w_i \left| f(x^{(i)})_k - f(x^{(j)})_k \right| + b) \tag{9}$$

The input is a pair of images and the output is 0 or 1 dependening on whether you inputed similar pictures or not.



[Taigman et. al., 2014. DeepFace closing the gap to human level performance] Andrew Ng

4.2 Neural Style Transfer

4.2.1 What is neural style transfer?

Using an input picture C (Content) using a style image S (Style), combine them to generate a new image (G) in this style.

4.2.2 What are deep ConvNets learning?

Visualizing what a deep network is learning Pick a unit in layer 1, Find the nine image patches that maximize the unit's activation. Repeat for other units.

What you find for layer 1 is that it's learning shades, layer 2 it will looks for texture or complicated, etc.

4.2.3 Cost Function

To build a Neural style transfer algorithm we need to define a cost function that tells how good is the output image.

$$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$$

Algorithm to find the generated image G

1. Initiate G randomly, G: 100 x 100 x 3
2. Use gradient descent to minimize $J(G)$, $G = G - \frac{d}{dG}J(G)$

4.2.4 Content Cost Function

$$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G) \tag{10}$$

- Say you use hidden layer l to compute content cost.
- Use pre-trained ConvNet. (E.g., VGG network)
- Let $a^{[l](C)}$ and $a^{[l](G)}$ be the activation of layer l on the images
- if $a^{[l](C)}$ and $a^{[l](G)}$ are similar, both images have similar content.

$$J_{content}(C, G) = \frac{1}{2}\left|\left|a^{[l](C)} - a^{[l](G)}\right|\right|^2 \tag{11}$$

But it's really just the element-wise sum of squares of differences between the activations in layer l, between the images in C and G. And so, when later you perform gradient descent on J(G) to try to find a value of G, so that the overall cost is low, this will incentivize the algorithm to find an image G, so that these hidden layer activations are similar to what you got for the content image.

4.2.5 Style Cost Function

Meaning of the style of an image

Say you are using layer l's activation to measure "style". Define "style" as **correlation** between activations across channels.

How correlated are the activations across different channels? How often they occur together in parts of an image or not. If we use the degree of correction across channels of an image, then we can do is compute the degree of correction in the generated image this channel is correlated. This will tells you how often in the generated image a given texture occurs.

Style matrix

Let $a^{[l]}_{i,j,k}$ = activation at (i, j, k), i.e. height, width, channel. $G^{[l]}$ is $n_c^{[l]}$ x $n_c^{[l]}$

For the style image, compute the non normalized loss function as follows:

$$G^{[l](S)}_{kk'} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a^{[l](S)}_{i,j,k} a^{[l](S)}_{i,j,k'}$$

For the generated image, compute the non normalized loss function as follows:

$$G^{[l](G)}_{kk'} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a^{[l](G)}_{i,j,k} a^{[l](G)}_{i,j,k'}$$

$$J^{[l]}_{style}(S, G) = \left|\left|G^{[l](S)} - G^{[l](G)}\right|\right|^2_F$$

Style cost function at layer l

$$J^{[l]}_{style}(S, G) = \frac{1}{(2n_H^{[l]} n_W^{[l]} n_C^{[l]})^2} \sum_k \sum_{k'} (G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'})$$

Summing up at all layers, the overal cost function becomes;

$$J_{style}(S, G) = \sum_l \lambda^{[l]} J^{[l]}_{style}(S, G)$$

4.2.6 1D and 3D Generalizations

You have learned a lot about ConvNets, everything ranging from the architecture of the ConvNet to how to use it for image recognition, to object detection, to face recognition and neural-style transfer.

Many of the ideas we learned here can be applied to 1D or 3D data, not just images.

4.3 Practice questions

4.3.1 QUIZ - Special applications: Face recognition & Neural style transfer

1. Face verification requires comparing a new picture against one person's face, whereas face recognition requires comparing a new picture against K person's faces.

- True (X)
- False

2. Why do we learn a function $d(img1, img2)$ for face verification? (Select all that apply.)

- Given how few images we have per person, we need to apply transfer learning.
- This allows us to learn to recognize a new person given just a single image of that person. (X)
- We need to solve a one-shot learning problem. (X)
- This allows us to learn to predict a person's identity using a softmax output unit, where the number of classes equals the number of persons in the database plus 1 (for the final "not in database" class).

3. In order to train the parameters of a face recognition system, it would be reasonable to use a training set comprising 100,000 pictures of 100,000 different persons.

- True
- False (X)

4. Which of the following is a correct definition of the triplet loss? Consider that $\alpha > 0$. (We encourage you to figure out the answer from first principles, rather than just refer to the lecture.)

- $max(||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 - \alpha, 0)$
- $max(||f(A) - f(N)||^2 - ||f(A) - f(P)||^2 - \alpha, 0)$
- $max(||f(A) - f(N)||^2 - ||f(A) - f(P)||^2 + \alpha, 0)$
- $max(||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha, 0)$ (X)

5. Consider the following Siamese network architecture:
The upper and lower neural networks have different input images, but have exactly the same parameters.

- True
- False

6. You train a ConvNet on a dataset with 100 different classes. You wonder if you can find a hidden unit which responds strongly to pictures of cats. (I.e., a neuron so that, of all the input/training images that strongly activate that neuron, the majority are cat pictures.) You are more likely to find this unit in layer 4 of the network than in layer 1.

- True (X)
- False

7. Neural style transfer is trained as a supervised learning task in which the goal is to input two images (x), and train a network to output a new, synthesized image (y).

- True
- False (X)

8. In the deeper layers of a ConvNet, each channel corresponds to a different feature detector. The style matrix $G^{[l]}$ measures the degree to which the activations of different feature detectors in layer l vary (or correlate) together with each other.

- True (X)

- False

9. In neural style transfer, what is updated in each iteration of the optimization algorithm?

- The neural network parameters
- The pixel values of the generated image $\mathcal{G}$ (X)
- The pixel values of the content image $\mathcal{C}$
- The regularization parameters

10. You are working with 3D data. You are building a network layer whose input volume has size 32x32x32x16 (this volume has 16 channels), and applies convolutions with 32 filters of dimension 3x3x3 (no padding, stride 1). What is the resulting output volume?

- 30x30x30x32 (X)
- 30x30x30x16
- Undefined: This convolution step is impossible and cannot be performed because the dimensions specified don't match up.