1. What are the advantages of a CNN for image classification over a completely linked DNN?

2. Consider a CNN with three convolutional layers, each of which has three kernels, a stride of two, and SAME padding. The bottom layer generates 100 function maps, the middle layer 200, and the top layer 400. RGB images with a size of 200 x 300 pixels are used as input. How many criteria does the CNN have in total? How much RAM would this network need when making a single instance prediction if we're using 32-bit floats? What if you were to practice on a batch of 50 images?

3. What are five things you might do to fix the problem if your GPU runs out of memory while training a CNN?

4. Why would you use a max pooling layer instead with a convolutional layer of the same stride?

5. When would a local response normalization layer be useful?

6. In comparison to LeNet-5, what are the main innovations in AlexNet? What about GoogLeNet and ResNet's core innovations?

7. On MNIST, build your own CNN and strive to achieve the best possible accuracy.

8. Using Inception v3 to classify broad images. images of different animals can be downloaded. Load them in Python using the matplotlib.image.mpimg.imread() or scipy.misc.imread() functions, for example. Resize and/or crop them to 299 x 299 pixels, and make sure they only have three channels (RGB) and no transparency. The photos used to train the Inception model were preprocessed to have values ranging from -1.0 to 1.0, so make sure yours do as well.

9. Large-scale image recognition using transfer learning.

a. Make a training set of at least 100 images for each class. You might, for example, identify your own photos based on their position (beach, mountain, area, etc.) or use an existing dataset, such as the flowers dataset or MIT's places dataset (requires registration, and it is huge).

b. Create a preprocessing phase that resizes and crops the image to 299 x 299 pixels while also adding some randomness for data augmentation.

c. Using the previously trained Inception v3 model, freeze all layers up to the bottleneck layer (the last layer before output layer) and replace output layer with appropriate number of outputs for your new classification task (e.g., the flowers dataset has five mutually exclusive classes so the output layer must have five neurons and use softmax activation function).

d. Separate the data into two sets: a training and a test set. The training set is used to train the model, and the test set is used to evaluate it.

Answer:

1. The advantages of a CNN over a completely connected DNN for image classification:

* CNNs are designed to handle image data, so they can capture spatial information better than fully connected networks.
* They use parameter sharing and convolutional layers, which allow them to learn features that are invariant to translation, rotation, and other spatial transformations.
* They are computationally efficient, since they reuse the same set of parameters across different regions of the input image.

1. Calculation of the total number of parameters and RAM requirements for a given CNN:

* The number of parameters can be calculated as the sum of the product of the number of filters, their spatial dimensions, and the number of input channels, for each convolutional layer, plus the product of the number of neurons in each fully connected layer and the number of neurons in the previous layer.
* The RAM requirements depend on the size of the input images, the number of parameters, and the batch size. For a single instance prediction, the RAM requirements are roughly proportional to the number of parameters. For a batch of images, the RAM requirements are proportional to the product of the batch size, the number of parameters, and the size of a single image.

1. Solutions to GPU memory issues during training:

* Reduce the batch size.
* Use smaller input images.
* Use smaller models or remove some layers.
* Use mixed precision training.
* Use gradient accumulation across multiple batches.

1. Advantages of max pooling over convolutional layers:

* Max pooling reduces the spatial dimensions of the feature maps, which helps to reduce the computational cost of subsequent layers.
* Max pooling introduces some degree of translation invariance by taking the maximum value over local regions, which helps to capture some of the key features of an image.

1. Use cases for local response normalization:

* Local response normalization was introduced as a way to promote competition among neurons in the same feature map, which can help to improve the generalization performance of the network.
* However, it has largely been replaced by other normalization techniques such as batch normalization, which have been shown to be more effective.

1. Core innovations of LeNet-5, AlexNet, GoogLeNet, and ResNet:

* LeNet-5 was one of the earliest CNN architectures and introduced the concept of convolutional layers and max pooling.
* AlexNet was the first CNN to use rectified linear units (ReLU) as activation functions and to use dropout to prevent overfitting.
* GoogLeNet introduced the inception module, which uses multiple filters of different sizes in parallel to capture features at different scales.
* ResNet introduced the residual block, which allows the network to learn identity mappings and avoid the vanishing gradient problem.

1. Building a CNN for MNIST:

* Start with a simple architecture consisting of a few convolutional and pooling layers, followed by one or more fully connected layers.
* Experiment with different architectures, layer sizes, activation functions, and optimization algorithms to find the best combination.
* Use techniques such as data augmentation, dropout, and weight regularization to prevent overfitting.
* Use a validation set to monitor the performance of the network during training and adjust the hyperparameters accordingly.

1. once the images of different animals have been downloaded and resized/cropped to 299 x 299 pixels with 3 channels (RGB), they need to be preprocessed to have values ranging from -1.0 to 1.0. This can be done by dividing the pixel values by 255 and then subtracting 0.5 from the result. The preprocessed images can then be fed into the Inception v3 model for classification.
2. the following steps can be taken:

* a. Gather a training set of at least 100 images for each class. This can be done by identifying your own photos based on their position (beach, mountain, area, etc.) or by using an existing dataset, such as the flowers dataset or MIT's places dataset.
* b. Create a preprocessing phase that resizes and crops the image to 299 x 299 pixels while also adding some randomness for data augmentation. This can be done using techniques such as random cropping, flipping, and rotation.
* c. Use the previously trained Inception v3 model and freeze all layers up to the bottleneck layer. The bottleneck layer is the last layer before the output layer and contains the most important features learned by the model. Replace the output layer with the appropriate number of outputs for your new classification task. For example, if the flowers dataset has five mutually exclusive classes, the output layer must have five neurons and use a softmax activation function.
* d. Separate the data into two sets: a training set and a test set. The training set is used to train the model, and the test set is used to evaluate it. The data should be randomly split into training and test sets, with typically around 80% of the data used for training and 20% used for testing. It's important to ensure that the same proportion of each class is present in both the training and test sets.