Homework 2

October 3, 2020

Convolutional Networks

Please submit a PDF for answers to Question 1. For all responses to Questions 2-4 include it within the respective notebooks and submit a zip of the Q1 PDF along with the two notebooks.

Question 1

(a) Convolutional networks make two assumptions about the structure of the input data. Say (1) what these assumptions are, (2) what advantages we gain by using a convolution when they are true. Give an example of a data set where these assumptions would not hold.

(b) Let $x[\cdot]$ and $y[\cdot]$ be two 1-D signals of length n. Consider a module which computes the cross-correlation between these two input signals with circular boundary conditions:

$$z[k] = \sum_{i=0}^{n-1} x[i] \cdot y[(i+k) \bmod n], \quad k \in \{0, \cdots, n-1\}.$$

Give the coefficients of the Jacobian of $z[\cdot]$ with respect to $x[\cdot]$ and $y[\cdot]$, *i.e.* give (1) $\frac{\partial z[k]}{\partial x[i]}$ and (2) $\frac{\partial z[k]}{\partial y[i]}$ for $k, i \in \{0, \cdots, n-1\}$.

(c) Neural networks typically map inputs to a higher-dimensional space in their early layers. Having points linearly separable makes classification and other types of prediction easier.

Say we are given a set of 1-D signals $\boldsymbol{x}^{(i)} \in \mathbb{R}^{100}$. Consider a fully-connected layer $f(\cdot)$ given by:

$$f(\boldsymbol{x}^{(i)}) = \sigma(\boldsymbol{W}\boldsymbol{x}^{(i)}),$$

where $\boldsymbol{W}$ is a 1000×100 weight matrix and $\sigma(\cdot)$ is a point-wise non-linearity.

Also consider a convolutional layer $g(\cdot)$ with 10 feature maps given by:

$$g(\boldsymbol{x}^{(i)}) = \sigma([\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_{10}]),$$

where $\boldsymbol{z}_j = \boldsymbol{x}^{(i)} * \boldsymbol{w}_j$ for some convolutional kernel $\boldsymbol{w}_j$ of size 3. Assume each $\boldsymbol{x}^{(i)}$ is padded with a zero at each end.

For $f(\cdot)$ and $g(\cdot)$, give:

i. The dimensionality of the output space.

ii. The number of trainable parameters in the layer.

iii. The computational complexity of computing the forward pass (number of operations).

Question 2

Here we will learn to build a convolutional neural network from scratch and train it on the provided dataset (CIFAR-10). The data is already loaded and provided to you. A brief description for each question is provided below, refer to the associated colab for more details.

1. First we will write a generic training loop that will be used throughout the assignment. Using the structure provided, fill in the relevant parts.

2. Write the object oriented version of ShallowNet Sequential model that is given

3. Instantiate the model and run this using an SGD optimizer, with the appropriate loss function for classification

4. Build a convolution network using the given specifications.

5. Report results of training using SGD optimizer for both ShallowNet and SimpleConvnet. What do you observe?

6. Batch normalization.

   - Modify the model class to include batch normalization
   - Plot the the training curves (training loss vs epochs, training accuracy vs epochs) using SGD (lr 1e-3) with and without batch normalization. Comment on the difference.
   - Run the same two networks with an Adam optimizer (lr 1e-4). Plot the the training curves (training loss vs epochs, training accuracy vs epochs) with and without batch normalization. Comment on the difference.
   - Once you choose an optimizer and see that it does train, make sure your model has enough capacity by overfitting on one batch of the data set. You should be able to get 100% train accuracy.

7. Modify the model class to include residual connections

8. Plot the training curves with and without the residual connection. Comment on the difference
9. Reducing overfitting. Experiment with different data transforms. Report learning curves and final accuracies.
10. Experiment with learning rate schedulers of your choice, report results on StepLR.
11. Experiment with a range of learning rates and optimizers, as well as the parameter in the learning rate scheduler for StepLR. Report the following plots:
    - Learning curves (training and validation loss for 5 different learning rate with SGD optimizer)
    - Learning curves (training and validation loss for 5 different learning rate with Adam optimizer)
    - Learning curves (training and validation loss for 5 different gamma parameter for the StepLR)
12. Load the model that gave you best validation accuracy and report results on the test set

Question 3

1. Generate saliency maps for given input image. Show results for the image url provided and optionally try with your own images too.
2. Generate an image "X_fooling" that the pre-trained model will classify as the target label "target_y". Achieve this by gradient ascent on the normalized target class score. Try for different "target_y" classes (they must all be one of 1000 ImageNet classes [1] and show results on a few. Now take a different pre-trained image and see if this fooling image also fools this other pre-trained model.

Question 4

Here we will see how to use a model that was trained using images at a low resolution for localisation of objects in images at higher resolution. Plot the classifier response to the last feature map of the model before average pooling.

[1] https://gist.github.com/yrevar/942d3a0ac09ec9e5eb3a