CS 512 – Assignment 3

Convolutional neural networks

Due by 11/9/2019

Review questions

1. Neural networks
   (a) Explain the template matching interpretation of a linear classifier. Explain the decision boundary interpretation of a linear classifier.
   (b) Explain how a similarity score can be converted to probability in the two class classification case and in a K class classification case.
   (c) Write expressions for L1 and L2 loss, Hubber loss, and cross entropy loss.
   (d) Explain the purpose of regularization in the loss function.
   (e) Explain the reason for taking a step in the opposite direction of the gradient in the gradient descent algorithm.
   (f) Explain the difference between gradient descent and stochastic gradient descent.
   (g) Explain how the learning rate can be selected in the gradient descent algorithm.
   (h) Explain the purpose of momentum in the gradient descent algorithm.
   (i) Explain the forward and backward pass in the back propagation algorithm. What is propagated in the backward pass?
   (j) Explain the difference between a fully connected layer and a convolution layer.
   (k) Explain the concept of dropout in a neural network.
2. Convolutional neural networks
   (a) Let I be a 4×4 RGB image where the R channel is all 1-s and G channel is all 2-s. The B channel has a value of 1 in its first row, a value of 2 in its second row, a value of 3 in its third row, and a value of 4 in its 4th row. Compute the convolution of this image with a $3 \times 3 \times 3$ filter having all ones without zero padding.
   (b) Repeat the previous question with zero padding.
   (c) Repeat the previous question when using dilated (atrous) convolution with a dilation rate of 2.
   (d) Explain the template matching interpretation of convolution.

(e) Explain how multiple scale analysis can be achieved with a fixed window size (using a pyramid).

(f) Explain how to compensate for spatial resolution decrease using depth (number of channels) and the purpose for doing so.

(g) Given a $128 \times 128 \times 32$ tensor and 16 convolution filters of size $3 \times 3 \times 32$, what will be the size of the resulting tensor when convolving without zero padding.

(h) Repeat the previous question when using a stride of 2.

(i) Explain how the number of channels can be reduced using a 1×1 convolution.

(j) Explain the interpretation of convolution layers and the difference between early and deeper convolution layers.

(k) Explain the purpose of pooling.

(l) Let I be an image as in the first question of this section. Write the result obtained using max pooling with a 2×2 filter with a stride of 2.

Programming questions

- In this assignment you need to implement a basic Convolutional Neural Network for classification. Your implementation needs to use a GPU framework. Specifically, Keras or TensorFlow. If you do not have access to GPU on your computer or elsewhere you can create a free account on google colab: https://colab.research.google.com/notebooks/welcome.ipynb
- You will train and evaluate a CNN to classify images of numbers in the MNIST dataset as either even or odd. Images in the MNIST dataset are labeled for 10 digit classes and so you will have to create labels of odd/even and discard the original labels. You may not use the original digit labels for training. Make sure that your program can save and load weights so that training can proceed with previous results. You will be required to log and plot some metrics as a function of training and evaluation iterations.

1. Construct and train CNN:

   (a) Load the MNIST set and split it into a training/validation/testing subsets of 55,000/10,000/5,000 examples respectively.

   (b) Convert the digit labels into odd/even labels and discard the original digit labels. You may not use the original digit labels for training.

   (c) Construct a network with two convolution layers with pooling, a dropout layer, and two fully connected layers. It is up to you to select the number of units in each layer.

   (d) Select the appropriate loss function, optimization algorithm (and its parameters), and evaluation metric.

   (e) Train the network and record the training and validation loss and accuracy measures.

   (f) Plot the training and validation loss as a function of epochs. Plot the accuracy as a function of epochs. In addition report the loss and accuracy values of the final training step.

2. Hyper-parameter Tuning: Evaluate different variations of the basic network as described below and measure performance. In your report, discuss your variations, compare the results you obtain and attempt to draw conclusions. Evaluate at least the following aspects:

   (a) Changing the network architecture (e.g. number of layers and/or organization of layers).

   (b) Changing the receptive field and stride parameters.

   (c) Changing optimizer and loss function (e.g. Adam optimizer).

   (d) Changing various parameters (e.g. dropout, learning rate, number of filters, number of epochs).

   (e) Adding batch and layer normalization.

   (f) Using different weight initializers (e.g. Xavier, He).

   (g) Evaluate the best validation model on the testing subset and report the results.

3. Inference: Write a program to use your pretrained custom CNN. The program should do the following:

   (a) Accept as input an image of a handwritten digit. Assume each image contains one digit.

   (b) Using OpenCV do some basic image preprocessing to prepare the image for your CNN: Resize the image to fit your model's image size requirement; Transform the grayscale image to a binary image (consider using the GaussianBlur() and adaptiveThreshold(), or any other type of binary thresholding that performs well); Display the original image and binary image in two separate windows.

   (c) Using your CNN classify the binary image (even/odd).

Submission instructions

Please follow the submission instruction of assignment 1.