The University of Texas at Austin
EE382V Scalable Machine Learning — Fall 2019

Homework Two

Dimakis   Due: Thursday, October 31, 3am.

Upload a pdf of your report. Upload also a separate zip file with all the code needed to replicate your results in the homework report. For final project proposals, replicate the proposal for all the students in each team.

Problem 1: Convolutions

1. Compute the output of a 1D convolution layer if the input is a single-channel vector [1,-1,3,4,4] and the kernal is [1,1] for stride =1 and padding =0.
2. Compute the output of a 1D convolution layer if the input is a single-channel vector [1,-1,3,4,4] and the kernal is [1,-1] for stride =1 and padding =1 (padding one means on both sides)
3. Write pytorch code that computes these outputs. Create the corresponding tensors and use torch.nn.Conv1d and ConvTranspose1d.

Problem 2: Training a CIFAR-10 CNN classifier.
In this problem we will follow a tutorial to build a Convolutional neural network that classifies images from the CIFAR-10 dataset and then modify it.

1. Start by implementing and running this tutorial. `https://www.stefanfiott.com/machine-learning/cifar-10-classifier-using-cnn-in-pytorch/` Show the accuracy you get and also the confusion matrix you obtain for the 10 classes.
2. Show the convolutional layer kernels you learned in the first convolutional layer.
3. Replace the architecture with AlexNet (`https://github.com/icpm/pytorch-cifar10/blob/master/models/AlexNet.py`) and report accuracy. Tune hyperparameters to get the best possible accuracy on a test set.
4. Using the first CNN you trained, build another binary classifer that classifies images as 'Plane' or 'Not Plane'. Compute the AUC of this binary classifier. Retrain this classifier to improve the AUC for this binary classification task. Report how you did that and what is the best AUC you were able to obtain.

Problem 3: Adversarial attacks: Horses and Cars.
Consider the first CNN model that you trained in the previous problem. Take an image x_1 of a horse that is classified correctly as a horse. It is possible to modify the input pixels very little (say 5-10 percent)) and fool the model that this image is a car. For the target attack class (Car in this example), compute the gradient of the class logit with respect to the input pixels. Modify the image in this gradient direction, while keeping each pixel to be perturbed by 0.1 (since each pixel

in [-1,1]) from the original horse image x. Create and show 8 adversarial perturbed horse images you created that are classified as cars.

Problem 4: Experimenting with Fast.AI (Extra credit 20 points).

1. **Paperspace**

   Create an account on paperspace or a cloud service of your choice. You should make sure that you are able to install all the libraries and tools needed for Fast.ai. If you choose to do this on `paperspace.com`, then:

   (a) Create an account on paperspace.
   (b) Log in.
   (c) Go to Gradient in the toolbar in the left.
   (d) Create Notebook.
   (e) Paperspace Fast.AI 1.0 (V3).
   (f) Choose the P5000 machine ($0.78/hour).
   (g) Launch, and go to: course-v3 $\longrightarrow$ nbs $\longrightarrow$ dl1 and open up the first lesson: lesson1-pets.ipynb.

2. **Fast.ai**

   Fast.ai has created a large library of tools that make setting up and training a neural network very easy, especially through the use of *transfer learning*. The notebook you opened in the previous exercise corresponds to the first lesson under "Practical Deep Learning for Coders." Watch this lesson, and follow along on the notebook.

3. Make sure you understand the key elements of using Fast.ai, including understanding how to access the help documentation. Specifically:

   (a) Figure out: ImageDataBunch: this is the main data structure that is used.
   (b) Figure out: create_cnn. This is the way you will create a "learner" that you will then train for some number of epochs. The key here is how easy it is to download a pre-set architecture (e.g., the notebook starts with resnet34, but has various others as well).

4. Explain what is a residual network, and the basic motivation for using it. Also explain what are the main elements of resnet34 and resnet50. How many layers, how many neurons total, how many weights; and then anything else you want to say.

5. Transfer learning using Fast.ai and create_cnn: Please explain how *pretrained* resnet34 is modified to get the network that the notebook ultimately trains (i.e., explain what are the last layers that are added).

6. Download a NOT pre-trained resnet34, and then by playing with the number of epochs and learning rates (possibly different learning rates across layers), see how low you can get the error. Can you get below 20%?

7. And for the main part of this HW: download (and label) your own data set of your choie, create a classification problem, and then use the main tools/ideas of this notebook to build a classifier. It does not need to be a multi-label classifier.

   For getting data, you may want to refer to the discussion here, for various tools that could be useful: `https://forums.fast.ai/t/tips-for-building-large-image-datasets/26688/36`.

Problem 5: Final project proposal.
Recall: the final project presentations will be on the last day of class, Saturday Dec 14th 1-5pm. Final Project presentations are on Dec 14th in class. Final project reports due December 15th midnight. Presentation duration: 10 minute presentations plus questions. The last problem of this homework is to write your project proposal (1-2 pages). Please include team members, what dataset you plan to use, what questions you plan to ask. Some preliminary results that show you have started experimenting with the data and show promise. One interesting direction is you plan to combine different datasets or search for dataset search engines for numerous resources. You will be graded on the project proposal.