Machine Learning Homework 4

May. 21, 2021

** Please note that all homework should be your own work. You should also not copy answers from other person's, books or internet resources.*
** I didn't proofread the questions. If you find any typos/errors, let me know.*

O Using the **PyTorch** **CNN** & **RNN** programs and datasets (Mnist & Cifar-10) uploaded at e-class, do the following experiments

1. CNN & MNIST

Refer to the CNN_Mnist program in e-class

1) change the current kernel size of the program to different size. (Change 'kernel_size' parameter of 'Conv2D' function.) Repeat this three times and compare the results.
2) remove pooling layer in the program (you can remove 'MaxPool2D' function) and compare the results.
3) change pooling layer in the program (ex. AvgPool2d, AdoptiveAvgPool2d, etc.) repeat this three times and compare the results. Refer to the following site.
https://pytorch.org/docs/stable/nn.html#pooling-layers
4) change the current activation function to other non-linear activation function (e.g. sigmoid, tanh, etc). You can do so by nn.Sigmoid() to nn.ReLU(), nn.Tanh(), etc. Repeat this five times and compare the results. Refer to the following site.
https://pytorch.org/docs/stable/nn.html#non-linear-activations-weighted-sum-nonlinearity
5) add dropout layers to the hidden layer and compare the results. Refer to the follwing site.
https://pytorch.org/docs/stable/nn.html#dropout-layers
5-1) (*optional*) Divide the program into train mode and eval mode and then, compare the result with 6) and write your thoughts on why you divided it into train mode and eval mode.
6) change the current optimization method to other optimization methods (e.g. adam, adaGrad, RMSProp, adaDelta, etc). You can use torch.optim.Adam, etc. Repeat this three times and compare the results.
7) now add the Xavier weight initialization method and compare the results. (use torch.nn.init.xavier_uniform)
8) choose ONE other parameters of CNN program (e.g. number of hidden nodes, epochs, batch normalization, etc). Change the value of this parameter and compare the results.

9) (*optional*) choose Adam optimization method and use L2 (ridge) regularization method this time. You can do so by setting 'weight_decay' value in optimization method (torch.optim.Adam) to a tiny number (e.g. 1e-5). Compare the results of using regularization.

2. CNN & CIFAR-10

Refer to the CNN_Cifar-10 program in e-class

1) repeat the question 1)-9) in Question 1.
* Tip: For the subquestion 2), (removing pooling layers) you need to modify the sizes of layers based on the tensor size. For instance, when using pooling layer, the total size of your tensor is [4, 16, 5, 5] = 1600. However, if you remove the pooling layer, the total size of your tensor becomes [4, 16, 24, 24] = 36,864. You need to modify the input size of 'x.view' and 'fx1' in your code according to the size of tensor
* Refer to the lab class for more info about this tip.

3. RNN & Mnist

(the way changing parameters in RNN is the same as that of CNN unless specified explicitly)

1) refer to the RNN_Mnist program in e-class
2) change the number of hidden nodes in the program three times and compare the results.
3) change the current optimization method to other optimization methods (e.g. adam, adaGrad, RMSProp, adaDelta, etc). Repeat this three times and compare the results.
4) change LSTM to GRU (or vice versa). Compare the results.
5) choose ONE other parameters of RNN program (e.g. batch_size, epochs, etc). Change the value of this parameter and compare the results.
6) compare the accuracy of RNN for Mnist with that of CNN.

7) (*optional*) choose Adam optimization method and use L2 (ridge) regularization method this time. You can do so by setting 'weight_decay' value in optimization method to a tiny number (e.g. 1e-5). Compare the results of using regularization.
8) (*optional*) use dropout technique and compare the results.
8-1) (*optional*) Divide the program into train mode and eval mode and then, compare the result with 8) and write your thoughts on why you divided it into train mode and eval mode.

9) save your checkpoint (save STATE_DICT of your model. your checkpoint must contain EPOCH and STATE_DICT of the model and optimizer.) and then, load the saved checkpoint and make sure it works. (don't save your **all model**. you have to save your **parameters only**.)

Hand In

1) upload the following files at e-class.
   - **report file, ii) program source file, iii) data file**

Due: 6/2(Wed)