AISE3010: Assignment 2

Arnav Goyal - 251244778

Malak Al-Hanafi - 251142157

February 26, 2024

The Submission

Our submission is a `.zip` file that extracts to the following directories:

- `aise3010_assignment2_q1.py` - referred to as the 'q1' script
- `aise3010_assignment2_q2.py` - referred to as the 'q2' script
- `models.py`
- `checkpoints/...`
- `data/...`

The `models.py` script holds the architecture for the models used in the Script, specifically `'Net2'` for part 1 and `'MidwayClassifer'` for part 2.

The question 1 script (`aise3010_assignment2_q1.py`) trains and tests the entire custom CNN architecture, it allows the user to train/test from scratch or train/test from a 'train checkpoint' stored at `checkpoints/Net2.pth`. The question 2 script (`aise3010_assignment2_q2.py`) extracts a feature midway through the convolutional part of the `Net2` CNN model, and trains a MLP classifier using these features as data. You can also train/test from scratch or train/test from a 'train checkpoint' stored at `checkpoints/MidwayClassifier.pth`. The `data` folder stores the CIFAR-10 dataset, which we used in this assignment.

Testing/Grading The Submission

To grade/use the submission you have two choices, we recommend saving some time by using the train checkpoints we provided:

- Use the train checkpoints - See Option 1
- Train both networks from scratch - See Option 2

The following sections will talk about the models/workings of the script.

Option 1: Using The Train Checkpoints

This is the easy way to test this submission, essentaially we have stored the state dictionaries from our pretrained models as `.pth` files in the `checkpoints` folder so you don't have to train them again! *Note:* These were trained with a batch size of 32, 20 epochs, 0.001 learning rate, and 0.9 momentum.

To proceed with this option, first open the q1 script, and enter `1` into the console to start from a checkpoint. The script will then load the checkpoint `Net2.pth` before prompting you again, at this point enter a `1` into the console to test from this loaded checkpoint. The console will then print out the test accuracy of the entire designed network. This completes question 1. *Note*: our checkpoint provides an accuracy of about 61.82%

For question 2, open up the q2 script and enter a `1` into the console to load from the provided checkpoint at `checkpoints/MidwayClassifier.pth`. This will load our pre-trained MLP classifier trained on the midway extracted features from the model in question 1. After this, you will be prompted for input again, at this point enter a `1` to test from this checkpoint. This first extracts features midway from the q1 model, and feeds them into the MLP classifier, and prints the accuracy of this classifier. This completes question 2. *Note:* Our checkpoint provides a testing accuracy of about 50.99%

Note: Feel free to play around while running the scripts, to further train from checkpoints, or see Option 2 to train and save new checkpoints from scratch – everything *should* work fine. . .

Option 2: Training From Scratch

This is the long way, regardless, open up script q1 and enter a `2` into the console when prompted, this lets the script know you arent using the checkpoints, and are opting to train for 20 epochs instead. At this point the training will begin and after about 108 years it will finish, you will then be shown a training curve, and upon closing it the program will ask if you would like to overwrite the current checkpoint, I would suggest entering a `1` to save these weights (this will overwrite the ones we provided, but you can always replace them with ours from the original submission's files). At this point testing will begin and will display accuracy once it is complete. This completes question 1.

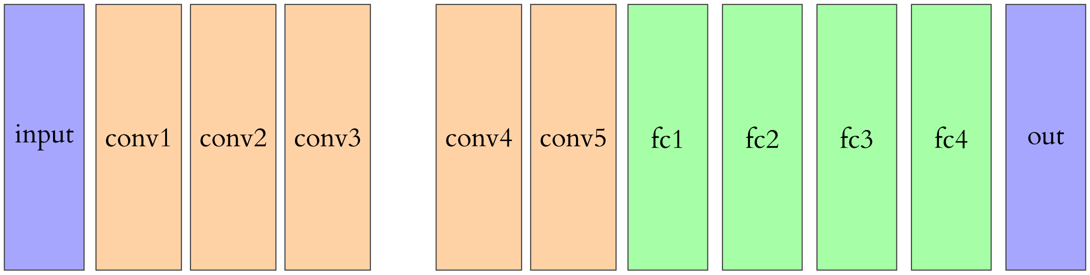
For question 2, run script q2 and enter `2` to train the midway feature classifier from scratch. This will start the training process, and displays the training curve after training. After closing the plot window, you will be prompted again, enter a `1` to save these weights and overwrite ours, similar to how you did in q1. The testing process will then begin and it will display accuracy at the end.

The Models

We use two models: `'Net2'` is our designed CNN for question 1, and `'MidwayClassifier'` is the classifier we trained to predict classes based on the extracted features midway through the forward pass of the `Net2` model's convolutional layers. Diagrams are shown below for ease of understanding

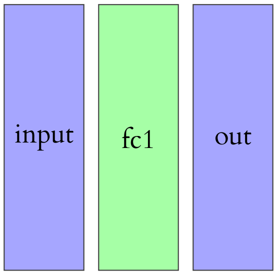
Model 1: The Custom Designed CNN – `'Net2'` for Question 1

Note that each orange conv block represents a convolution layer + relu + max pooling. Note that each green fc block features a fully-connected layer + relu activation



Model 2: The Midway Classifier - `'MidwayClassifier'` for Question 2

This is an Single-Layer Network designed to train/predict classes of extracted features from an input feature taken between conv3 and conv4 of the `Net2` CNN (shown above). Note that each green fc block features a fully-connected layer + relu activation



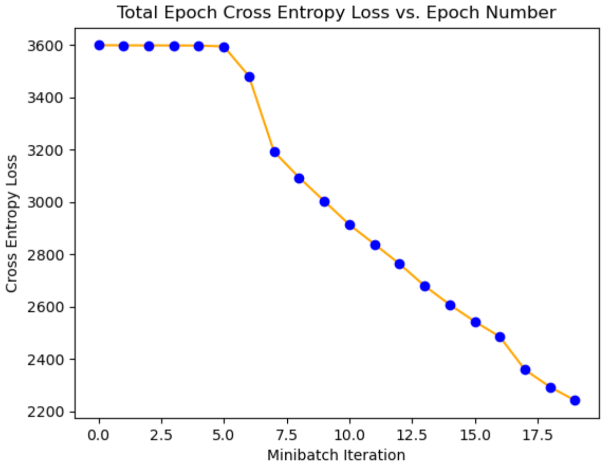
The detailed layer parameters of both models are contained within the `models.py` script and will not be re-explained here for sake of brevity (this report is so long already).

The important part is the the q1 script trains **Model 1** with CIFAR-10 and tests this model as well, and that the q2 script uses CIFAR-10 on the trained Model 1 to get the midway features from between layers conv3 and conv4, and then feeds this extracted feature into **Model 2** for training and/or testing purposes.

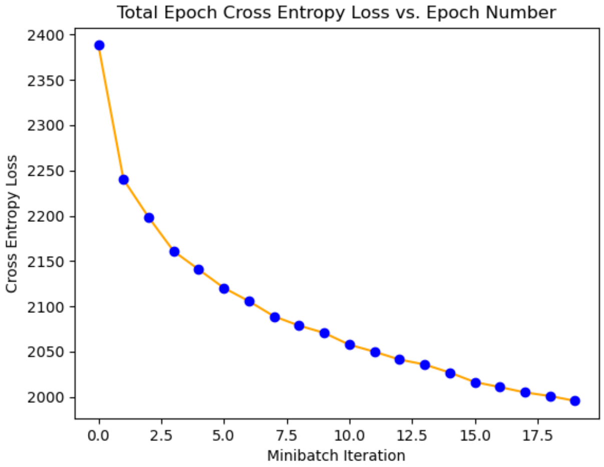
Analysis & Improvement

The results (training accuracy) we get make sense due to the differences in both models. Model 1 is vastly more complex than Model 2, and hence it achieves more accuracy on this complex image dataset (CIFAR-10). We could also train for some more time, as it seems both models haven't plateau'd in terms of the training curve as of yet. These curves are attached below. Realistically we should also take care to not overfit to our training dataset. This is esepcially more likely with Model 2 due to its small size. Regardless, training for more epochs does seem to somewhat increase the test accuracy, so we could do that or use a different (maybe better?) architecture in Model 1 to increase this accuracy as well.

Plot 1: Training Curve from Question 1 Script



Plot 2: Training Curve from Question 2 Script


Total Epoch Cross Entropy Loss vs. Epoch Number
Cross Entropy Loss
Minibatch Iteration