11785 HW4P2

Name: Xinkai Chen

AndrewID: xinkaic

Code: 11785_hw4p2_xinkaic.ipynb

Steps:

1. Download data and unzip them
2. Import libraries and run on GPU
3. Convert images to tensors, define dataloaders and load training data and dev data

- Dataset is pretty straightforward; in the collate_fn I padded inputs and labels together.
- Dataloader has batch size of 64 (I followed the recitation code)

4. Define network

- The network is Listen, Attend, Spell, but it's hard to implement
- I used the hidden sizes as suggested in the writeup
- I used three layers of pLSTM in the encoder, and two hidden layers in the decoder, as suggested
- I used 10% of the teacher forcing; I tried using Gumbel noise as well but then it does not work too well
- I didn't pretrain the decoder first as a language model, but then training end-to-end works okay
- I used the tricks of weight tying and lock dropout of 0.5 to work with the hidden layers

5. Define training parameters

- Criterion: CrossEntropyLoss(). It took me a while to make the mask correctly and get the masked loss to compute the perplexity.
- Learning rate: starting with 0.001, *0.95 after every 10 epochs, although it does not seem to help too much
- Optimizer: Adam with weight decay of 5e-6 and the learning rate above

6. Training

- The distance reaches the A cut-off after around 48 epochs (I think I was one of the few ones who passed the A-cutoff on time, but it did drop due to the public score)
- Validate after each epoch, printing the edit distance so that I know the model is working

7. Evaluate results and tune hyperparameters

- I tried different techniques like weight tying and lock dropout. And then I evaluate the model on the validation set. Also I draw the attention plot to see if my model works correctly.

8. When the validation distance is okay, load the test set and predict on test set. I tried implementing random search but it turned out that greedy search worked quite well.
9. Generate submission.csv and submit to Kaggle.