LING 575K HW4

Due 11PM on Apr 29, 2021

In this assignment, you will

- Develop your understanding of feed-forward networks for text classification
- Implement the Deep Averaging Network
- Understand a more sophisticated optimizer than vanilla SGD
- Explore some regularization techniques

1 Implementing the Deep Averaging Network + Loss [40 pts]

Q1: Implement bag of words representation The input to a DAN model is a *bag of words*: a vector (a batch of such vectors, really) with counts of the vocabulary items at the corresponding indices (see slide 14 in lecture 6). [5 pts]
In `data.py`, implement the building of this representation in `SSTClassificationDataset.example_to_tensors`.

Q2: Implement DeepAveragingNetwork.forward In `model.py`, you will find the skeleton of a Deep Averaging Network (with two hidden layers). In particular, we have implemented the initialization of the Module. You need to implement the forward method, which takes two inputs: a batch of bag-of-words representations and a list of lengths, and then outputs scores for each class label for each row in the batch. See the doc-string for more information. Note: the edugrad repository (and the slides for HW4) has an example of implementing forward for an MLP that you can use for inspiration. [15 pts]

Q3: Cross Entropy Loss In `ops.py`, you will find some ops already defined, and some for you to implement.

- Implement the `exp` op. [6 pts]
- Implement the `softmax_rows` method. Note: you should be using already defined ops/methods to define this method, so there's no separate backward here. [4 pts]
- Implement `cross_entropy_loss`. The same note above applies here. [10 pts]

2 Implementing Adagrad [15 pts]

Recall from the lecture on DANs the update rule for Adagrad:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\alpha}{\sqrt{G_{t,i} + \epsilon}} g_{t,i}$$

$$G_{t,i} = \sum_{k=0}^{t} g_{k,i}^2$$

$$\text{where } g_{t,i} := \nabla_{\theta_{t,i}} \mathcal{L}(\theta_t)$$

In `optim.py`, please implement this update rule by incrementing $G_{t,i}$, computing the adjusted learning rate, and finally updating the parameter values. Note: you can look at edugrad.optim.SGD for an example step method.

3 Running the Deep Averaging Network [20 pts]

In `run.py`, you will find a basic training loop for building a DAN and training it on the Stanford Sentiment Treebank. There are several command-line arguments specifying different arguments. The default arguments for various hyper-parameters are:

- Hidden dimension: 50
- Embedding dimension: 50
- Batch size: 32
- Number of epochs: 6
- Word dropout: 0.0 [i.e. no word dropout is applied]
- L_2 regularization: 0.0 [i.e. no regularization is applied]

Each run will print to stdout (and, therefore, the output file you specify in your condor job script) the training and dev set loss for each epoch, and then the final model's accuracy on the dev set.

Q1: Run 1, default arguments Run the main training loop by calling `run.py` with all of the default arguments. Record the outputs of this run (per epoch train/dev loss, final dev accuracy) here:

In 2-3 sentences, describe any trends that you see in the training and dev set losses over the course of training, and any differences between the two. What do these trends suggest to you? [5 pts]

Q2: Run 2, L_2 regularization Recall that L_2 regularization adds a penalty to the loss function corresponding to the size of the model's parameters:

$$\mathcal{L}'(\theta, y) = \mathcal{L}(\theta, y) + \lambda \|\theta\|^2$$

Run the main training loop by calling `run.py`, but with the L_2 parameter set to 1×10^{-5} (this corresponds to λ in the above equation). Record the outputs of this run (per epoch train/dev loss, final dev accuracy) here:

In 2-3 sentences, describe any trends that you see in the training and dev set losses over the course of training, and any differences between the two. What do these trends suggest to you? [5 pts]

Q3: Run 3, L_2 regularization + Word Dropout Recall that word dropout randomly drops some percentage of word embeddings from the input. Run the main training loop by calling `run.py`, but with the L_2 parameter set to 1×10^{-5} and word dropout set to 0.3. Record the outputs of this run (per epoch train/dev loss, final dev accuracy) here:

In 2-3 sentences, describe any trends that you see in the training and dev set losses over the course of training, and any differences between the two. What do these trends suggest to you? [5 pts]

Q4: Cross-run comparison What trends do you notice in these metrics across the three runs? What do they suggest to you about the impact of these hyperparameters on the model's performance? [5 pts]

4 Testing your code

In the dropbox folder for this assignment, we will include a file `test_all.py` with a few very simple unit tests for the methods that you need to implement. You can verify that your code passes the tests by running `pytest` from your code's directory, with the course's conda environment activated.

Submission Instructions

In your submission, include the following:

- readme.(txt|pdf) that includes your answers to §3.
- `hw4.tar.gz` containing:
  - run_hw4.sh. This should contain the code for activating the conda environment and your three run commands for §3 above. You can use run_hw2.sh from the previous assignment as a template.
  - data.py
  - model.py
  - ops.py
  - optim.py