Homework 3

CS 4803/7643 Deep Learning - Homework 3

In this homework, we will implement vanilla recurrent neural networks (RNNs) and Long-Short Term Memory (LSTM) RNNs, and apply them to image captioning on COCO. We will also build the forward pass of a Transformer for classification.

Note that Parts 1-3 are adapted from the Stanford CS231n course, and Part 4 is unique to Georgia Tech's course.

Download the starter code here.

Setup

Assuming you already have homework 2 dependencies installed, here is some prep work you need to do. First, download the data (you will need about 4GB of disk space, and the download takes some time):

```
cd cs231n/datasets
./get_assignment_data.sh
```

Note: this homework is compatible with and has been tested on Python 3.6.2+ and PyTorch 1.1+ on Linux and Mac. Make sure you have all dependencies listed in requirements.txt installed.

Part 1: Captioning with Vanilla RNNs (15 points)

Open the `RNN_Captioning.ipynb` Jupyter notebook, which will walk you through implementing the forward and backward pass for a vanilla RNN, first 1) for a single timestep and then 2) for entire sequences of data. Code to check gradients has already been provided.

You will overfit a captioning model on a tiny dataset and implement sampling from the softmax distribution and visualize predictions on the training and validation sets.

Part 2: Captioning with LSTMs (15 points)

Open the `LSTM_Captioning.ipynb` Jupyter notebook, which will walk you through the implementation of Long-Short Term Memory (LSTM) RNNs, and apply them to image captioning on MS-COCO.

Part 3: Train a good captioning model (10 points, Extra Credit for CS4803, Regular Credit for CS7643)

Using the pieces you implement in parts 1 and 2, train a captioning model that gives decent qualitative results (better than the random garbage you saw with the overfit models) when sampling on the validation set.

Code for evaluating models using the BLEU unigram precision metric has already been provided. Feel free to use PyTorch for this section if you'd like to train faster on a GPU.

Write a text comment in the notebook explaining what you tried in your model. Also add a cell that trains and tests your model. Make sure to include the call to evaluate_model which prints out your highest validation BLEU score (>0.3) for full credit.

Here are a few pointers:

- Attention-based captioning models
  - Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. Xu et al., 2015
  - Knowing When to Look: Adaptive Attention via A Visual Sentinel for Image Captioning. Lu et al., CVPR 2017
- Discriminative captioning
  - Context-aware Captions from Context-agnostic Supervision. Vedantam et al., CVPR 2017
- Novel object captioning
  - Deep Compositional Captioning: Describing Novel Object Categories without Paired Training Data. Hendricks et al., CVPR 2016
  - Captioning Images with Diverse Objects. Venugopalan et al., CVPR 2017

Part 4: Classification with Transformers (15 points)

Open the `Transformer_Classification.ipynb` Jupyter notebook, which will walk you through implementing a Transformer for classification in PyTorch. Note that you will need to include your code in `transformer.py` as a PDF when submitting this assignment (detailed below).

Note: we will be grading code for this question. The tests provided in the notebook are intended to help you get on the right track, but might not catch all possible mistakes, so make sure to follow the instructions carefully with your implementations!

Submit your homework

First, combine all of your PDFs into one PDF, in the following order: Make sure that you have run all cells in each notebook and that the output is displayed in the PDFs.

1. Your solutions to questions in PS3
2. Your RNN_Captioning converted PDF
3. Your LSTM_Captioning converted PDF
4. Your Transformer_Classification converted PDF
5. Your transformer.py code converted to a PDF

Please use nbconvert or the ‘Download as PDF’ option in Jupyter to convert the notebook into PDFs. This PDF will be submitted under the HW3 designation in Gradescope.

Run `collect_submission.sh`

```
./collect_submission.sh
```

Submit this ZIP to the HW3 Code designation in Gradescope.

Although we will not run your notebook in grading, you still need to **submit the notebook with all the outputs you generated**. Sometimes it will inform us if we get any inconsistent results with respect to yours.

References:

1. CS231n Convolutional Neural Networks for Visual Recognition