CS 4650 (Spring 2021) HW5 or Course Project

March 2021

Instructions

1. This final homework has three options **(choose one)**: both Option 1 and 2 (**individual, no group allowed**) are coding problems ; Option 3 is the open-ended course project on topic of student's choice (individual or group of up to 3 students).
2. We will be using Gradescope to collect your assignments. Please carefully read the following instructions for submitting to Gradescope.
   (a) Please upload code/pdf/report (notebook should be transformed to pdf with running output) under **HW5 - MyQA.py** and **HW5 - Notebook (PDF)** (if you choose Option 1), or **HW5 Programming MT** (if you choose Option 2), or **HW5 Course Project Report** and **HW5 Course Project Data/Code** (if you choose Option 3) on Gradescope.
   (b) Note: this is a large class and Gradescope's assignment segmentation features are essential. Failure to follow these instructions may result in parts of your assignment not being graded. We will not entertain regrading requests for failure to follow instructions.
3. We generally encourage collaboration with other students. You may discuss the questions and potential directions for solving them with another student. However, you need to write your own solutions and code separately, and not as a group activity. Please list the students you collaborated with on the submission site.

1 Option 1: Question Answering

Individual, no group allowed. In this assignment, you will implement a slight variant of the Bi-Directional Attention Flow (BiDAF) for Machine Comprehension. You should refer to the original paper as you implement the methods. The dataset you will use is The Stanford Question Answering Dataset (SQuAD 2.0). This dataset is composed of question-answer pairs that pertain to a particular passage from a Wikipedia article. Every passage has several questions, and each question has several answers. Furthermore, each answer is also annotated with its start and end position in the passage, and thus you will be using the answer's start and end position as labels to provide supervision. Feel free to explore and familiarize yourself with the data and check out Background, Challenges, Progress to learn more.

You will write your code in `MyQA.py` and `HW5.ipynb`. `MyQA.py` should be submitted to **HW5 - MyQA.py**. Please make sure you don't use or import any additional packages. Furthermore, `HW5.ipynb` should be converted into PDF and submitted to **HW5 - Notebook (PDF)**.

(a) Implement the `LSTMEncoder` class in `MyQA.py`. This will look very similar to the `CharPOSTagger` code you wrote in HW3 Q3. Instead of encoding characters using a CNN, as they do in Sec. 2.1 of Seo et al. (2017), you will just use another LSTM for simplicity. You will also ignore the "Highway Network" used in Seo et al. (2017) for simplicity. These are the only two deviations from the original paper, and the rest of the architecture will stay the same. Make sure you test whether the output dimensions are correct by running the notebook cell in *Part 1: Encoder*. Your implementation should work correctly with various hyperparameters, and this will be tested by the autograder. (6 points)

(b) Implement the `AttentionFlow` class in `MyQA.py`. More guidance is provided in `MyQA.py`. You might also need to refer to the original paper to better understand the approach. Make sure you test whether the output dimensions are correct by running the notebook cell in *Part 2: Attention Flow Layer*. Your implementation should work correctly with various hyperparameters, and this will be tested by the autograder. (10 points)

(c) Implement the `ModelingLayer`, `OutputLayer`, and `BiDAF` classes in `MyQA.py`. More guidance is provided in `MyQA.py`. You might also need to refer to the original paper to better understand the approach. Make sure you test whether the output dimensions are correct by running the notebook cells in *Part 3: Modeling Layer*, *Part 4: Output Layer*, and *Part 5: the BiDAF class*. Your implementation should work correctly with various hyperparameters, and this will be tested by the autograder. (10 points)

(d) Implement the train and eval functions in `HW5.ipynb`. You don't need to train the network for optimal accuracy. However, you should train it for 3 epochs to verify that your implementation works. Make sure the outputs are preserved when you convert your notebook into PDF. (14 points)

2 Option 2: Neural Machine Translation (part 2)

Individual, no group allowed. In this problem, you will implement a LSTM Seq2seq machine translation model with attention mechanism, and carry out hyper-parameter tuning. Besides, you need to implement the beam search algorithm and visualize the attention. You need to first copy the skeleton ipython notebook to your own Google Drive. After completion, you need to transform the notebook to pdf and submit your pdf file under **HW5 programming MT** on Gradescope. When making a submission, make sure all the outputs are present. You can reuse `make_batch` function, `predict_greedy` function, `encode` and `compute_loss` methods in `Seq2seqBaseline` Class, if you implemented them properly in your last project. Otherwise, you may make some adjustments in your previous code to make it work in this project.

(a) Implement a LSTM Seq2seq machine translation model with attention mechanism by completing the skeleton code provided in `Seq2seqAttention` Class in the notebook.

Under the default hyper-parameter setting, after 10 epochs you should be able to get at least a validation token accuracy above 62%. Using your implemented greedy search function, you should get above 34

BLEU on the validation set. Note that it takes around 9 minutes to train the model for 10 epochs with default setting on GPU. (12 points)

(b) Implement the Beam Search algorithm by completing the skeleton code provided in `predict_beam` function in the notebook. You should get an improvement of at least 0.3 BLEU over greedy search, and should reach above 34.3 BLEU with attention. (12 points)

(c) In order to improve your model performance, try the hyper-parameter tuning. Specifically, you are expected to make some modifications on embedding and hidden dimensions, number of LSTM layers, dropout rate, learning rate, batch size and training epochs, where training epochs should be no larger than 30. You will receive another 4/8 pts if your model is able to achieve above 35/36 BLEU on the validation set. (8 points)

(d) Implement the Attention Visualization part to visualize attention scores and describe your findings in the notebook. (8 points)

3 Option 3: Course Project

As an alternative, we allow student to work in groups on a research topic of student's choice, with the goal of gaining experience applying the NLP techniques presented in class to real-world datasets. Students should work in groups of 2 or 3 (if go solo, confirm w/ instructor first). Each group will **submit a report together with data/code** by May 4th (same as the late submission day for students who choose Option 1 or 2), **AND give a 5-10 minute oral presentation** at the final exam time (May 5th 2:40 PM - 5:30 PM). No later submission or presentation will be accepted.

In general, we recommend considering to re-implement or create state-of-the-art NLP models or create new interesting/novel applications. It should be quite substantial work (more than an individual programming homework as you are working as a group). For inspiration, here are some example NLP course projects done by students: `http://web.stanford.edu/class/cs224n/project.html`

There are also many NLP shared tasks with existing datasets and sometimes baseline systems at SemEval and WNUT workshop, as well as benchmarks such as GEM. If you want to get some Twitter data to work with, you can roughly follow the Twitter API tutorial.

Report (2-4 pages, not including references):

- Use LaTeX template from ACL 2021. Uncomment this line "%\aclfinalcopy" in the file.
- Include a statement of each group members' contribution, if working in a group.
- Include:
  - Title and Author(s)
  - Goal: What are you trying to do? Give an example of inputs/outputs or user interaction. How might it be helpful to people?
  - Method: How are you trying to solve the problem? What are existing approaches to the problem?
  - Experiments: What data do you use? What metric(s) do you use to measure success? What baseline method do you compare against? How well do your methods perform compared with the baseline, and why?
  - Conclusions: What did you learn from your experiments? Suggest future ideas.
  - Replicability: Submit (or include links to) all the code that you wrote and all the data that you used.
  - Related Work and Reference: This is absolutely necessary. You may use any existing code, libraries, etc. and consult any papers, books, online references, etc. for your project. However, you must cite your sources in your writeup and clearly indicate which parts of the project are your contribution and which parts were implemented by others.