MIT 6.806/6.864: Advanced Natural Language Processing **Spring 2021**

Homework 3

Due Date: Fri. April 9, 2021, 12:00 PM (noon) EDT (*Canvas submission*)

Seq2seq and Trees

Introduction. In this assignment, you will implement a neural machine translation (NMT) system using an RNN-based sequence-to-sequence (seq2seq) model and practice constituency based parsing for semantic interpretation. We provide scaffolding code for each part of the lab in two Jupyter notebooks; the code for Part 1 is in `6864_hw3_seq2seq.ipynb` and the code for Part 2 is in `6864_hw3_Trees.ipynb` (you can find both notebooks on Canvas). We recommend saving a copy of each notebook and using Google Colab to write and execute your code.

General report guidelines. Homework assignments should be submitted **anonymously** in a pdf with a maximum of four single-spaced pages for 6.806 (six for 6.864), with one section for each part of the assignment below. Please clearly mark whether the assignment is being submitted for 6.806 or for 6.864. The code should be appended to the end of the pdf (you can save the colab notebook as a pdf) and does not count towards length. Each section should describe the details of your code implementation and include whatever analysis and figures are necessary to answer the corresponding set of questions.

Part 1: Seq2seq

The first part of your lab report should discuss any important details and design decisions you made when implementing the seq2seq models and decoding algorithms. Additionally, you should design and conduct experiments to answer the following questions in your report:

(a) (*Theoretical*) Consider a *deterministic* HMM, $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}\}$, with N hidden states and an output vocabulary of size V. $\boldsymbol{A}$ and $\boldsymbol{\pi}$ are both *deterministic* (i.e. one entry in each row is 1, all others are 0); $\boldsymbol{B}$ doesn't have to be. When you sample, it will output a sequence $\boldsymbol{O} = o_1, o_2, ..., o_T$.

Now consider an RNN of the form:

$$
\begin{aligned}
h_t &= f(W_1 x_{t-1} + W_h h_{t-1} + b_h) \\
s_t &= g(W_2 h_t + b_x) \\
x_t &\sim s_t \text{ (read: } x_t \text{ is sampled from } s_t)
\end{aligned}
$$

Set up the RNN such that its samples $\boldsymbol{X} = x_1, x_2, ..., x_T$ are distributed identically to the samples $\boldsymbol{O}$ from the HMM.

Note: A complete solution will clearly define values for the RNN's parameters based on the HMM values and explain which parts of your RNN's structure correspond to which parts of an HMM. It will also explain how the mechanism of sampling will take place with your RNN and explain why this is the same as in the HMM.

(b) (**6.864 students only**; *Theoretical*) What would you need to change to your answer above in order to handle an HMM with non-deterministic transitions? (A couple of sentences is fine here.)

(c) (*Experimental*) **Analyzing Decoded Output.** Let's explore the outputs of the model that you've trained! Using you EncoderDecoder model, identify 2 examples of errors that it produces. The two examples you find should be different error types from one another. For each example you should:

(i) Write the source sentence in Vietnamese.

(ii) Write the target English translation.

(iii) Write your model's English translation.

(iv) Identify the error in the model's translation.

(v) Provide a reason why the model may have made the error (either due to a specific linguistic construct or specific model limitations).

(vi) Describe one possible way we might alter the system to fix the observed error.

Note: Fluency in Vietnamese is definitely not required! It might help to use google translate to see what each "word" or chunk of "words" corresponds to (in Vietnamese, spaces do not always indicate word boundaries).

(d) (*Conceptual*) Is there another decoding algorithm that might work better? Why? (a single sentence answer is fine)

(e) (*Experimental*) First, in one sentence, describe what a BLEU score is. Then, try to improve your BLEU score. For example, you could try stacking more RNN layers, switching cell types, trying out attention (warning: not for the faint of heart) or applying bi-directionality to the encoder. Describe what you try, even if it doesn't show improvement.

Tip: The TA's preliminary implementation achieves a BLEU score of around 6. You don't have to surpass that number (although it's pretty simple to do so)–this is just to give you some sense of what a baseline should get. Training on the entire training set takes some time. So tune your hyperparameters on a smaller training set (you can do so by changing sampling when creating the data loader). By adding an attention mechanism, we get a BLEU score of around 16.

Part 2: Trees

(a) *(Theoretical)* In this question, you will be tasked with writing a *Context Free Grammar* (CFG) for a subset of English language that includes simple intransitive sentences (1) and those with an arbitrary amount of object relative ("that") clauses (2-4). (Note: the "that" is optional in normal English but required for this problem.)

Example Sentences:

(1) The man ran

(2) The man [that the cat chased] ran

(3) The man [that the dog [that the cat feared] chased] ran

(4) The man [that the dog [that the cat [that the man loved] feared] chased] ran

Our Vocabulary:

- Nouns = *{man,dog,cat}*
- Intransitive verbs = *{meowed, barked, ran}*
- Transitive verbs = *{feared, chased, loved}*
- Determiner = *{the}*
- Conjunction = *{that}*

(i) For the first part of the problem assume that there can be an infinite number of embedded object-relative clauses in a sentence. Note that the top level sentence always has an intransitive verb. Please provide the CFG rules that can generate this subset of the English language (please ignore the brackets). Here is a rule to get you started: S $\rightarrow$ NP VP .

(ii) Now, let's consider the case where only up to double nesting of object-relative clauses is allowed (i.e. (3) is fine but (4) isn't). How could you modify your grammar to obtain this language? Please provide the CFG rules.

Part 2 of your lab report should briefly discuss any implementation details that were important to filling out the code part of the homework. Then use the code to answer the following questions:

Part A

(a) (*Experimental*) What are your final scores? (Please provide `F1`, `exact_match`, and `tree_match` results). What percent of the predictions are well formed?

(b) (*Experimental*) We generate the word and span embeddings with a bi-directional LSTM. Why is this better than using a uni-directional LSTM?

(c) (*Experimental*) Can you suggest one potential method to improve the current algorithm for generating span embeddings? Explain why it is possible to do better.

Part B (6.864 only)

(a) (**6.864 students only**; *Experimental*) What are your final scores? Did you see any improvement over the baseline model in Part A? What's the reason for the drop in F1 scores? (Please provide `F1` and `exact_match`, `tree_match` results.)

(b) (**6.864 students only**; *Experimental*) In results, find some trees that have the correct tree structure but incorrect labeling. Then, print the labeling mistakes for those examples. Qualitatively comment on the mistakes you observe.

(c) (**6.864 students only**; *Experimental*) In training, we label some random spans without labels as `None`. How does that help with our decoding?

(d) (**6.864 students only**; *Theoretical*) Is it easy to frame the whole problem as a seq2seq task? How would you do it?