Unsupervised Language Learning 2015-16: Assignment 2

1 Models

In this task, you will implement an Encoder-Decoder. This architecture consists of two RNNs. The first RNN (encoder) reads the input sequence and encodes it in a fixed-length vector. This vector will be used to initialize the second RNN (decoder) that generates the output sequence. The encoder and the decoder are trained jointly to maximize the conditional log-likelihood of the output sequence given the input sequence. You are required to implement the encoder-decoder system with both standard RNNs and GRUs[1]. The work by Cho et al. [1] is a good starting point to understand the architecutre. For additional reference, you can consider the sequence-to-sequence model proposed by Sutskever et al. [2] which implements a similar architecture but with a slightly more complicated type of recurrent units.

2 Data

We use the (20) babi tasks dataset v1.1 for this assignment[2]. Babi is a synthetic dataset designed for developing and evaluating machine learning models on a set of toy tasks that are prerequisite for reasoning in natural language. [3] There are 20 sub-tasks defined in this dataset, for each sub-task you will find two files, one for training and the other for testing. It is important to thoroughly read the project's webpage: `https://research.facebook.com/researchers/1543934539189348`.

3 Tasks

In general, the input and output are two separate sequences of variable size. In the first part of the assignment we want you to use the encoder-decoder architecture to reverse the input sequence to the output. In the second part you will use your implemented models for a more interesting task of factoid question answering.

3.1 Learn to reverse

For the first part of the assignment you need to train an encoder-decoder to reverse the sequences. In this case, your encoder encodes the input sequence to a vector and the decoder conditioned on this vector will replicate the input sequence. You can train your model on each line of the training files and test it on each line of the test files. Use only the files for the first 5 tasks (those that their names start with qa1,qa2,...,qa5). You should implement two versions of your model, one with RNNs and the other with GRUs.

1. What is the performance of both models on the training and the test set? Which model is doing better?

[1] GRUs will be presented during the practice section on April 25.

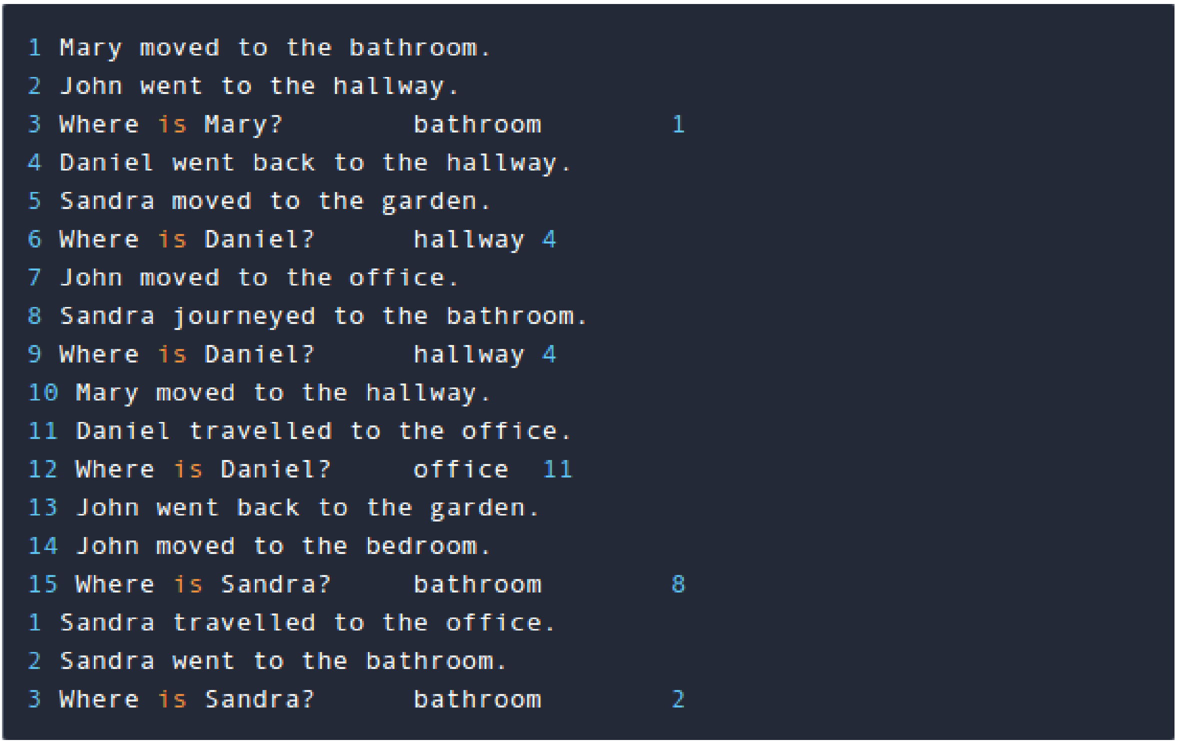
[2] You only need the English subset of the dataset. There are two of them, use the smaller one.

2. Group the input sequences in your dataset based on their sequence length and draw the distribution of the model performances for each group, also report the statistics of each distribution.
3. Do you see any common pattern in errors of your model predictions?

3.2 Learn to answer

In this part, you will use your model (RNN and GRU) to solve toy factoid question answering tasks on all the 20 sub-tasks in the babi dataset. Each file in the contains several stories and each story is consisted of a set of statements (context) followed by few questions. How do you use an encoder-decoder to solve this task? One approach is to concatenate the context sentences and the questions and feed the result to the model. The encoder encodes the input (question + context)[3] and the decoder generates the answer (output sequence of length 1).[4]

For example:



```
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary?        bathroom        1
4 Daniel went back to the hallway.
5 Sandra moved to the garden.
6 Where is Daniel?      hallway 4
7 John moved to the office.
8 Sandra journeyed to the bathroom.
9 Where is Daniel?      hallway 4
10 Mary moved to the hallway.
11 Daniel travelled to the office.
12 Where is Daniel?     office  11
13 John went back to the garden.
14 John moved to the bedroom.
15 Where is Sandra?     bathroom        8
1 Sandra travelled to the office.
2 Sandra went to the bathroom.
3 Where is Sandra?      bathroom        2
```

For the second question (line 6), the input sequence is the concatenation of lines 1,2,4,5 and the question itself.[5]

1. Report the test and train accuracy for each one of the 20 tasks separately.
2. Report the test and train accuracy for joint training on all the tasks.
3. Explain the variance of error distribution across different tasks? What makes a task harder than the others?

[3] The order of concatenation matters, the question should come first.

[4] You should consider the question mark as a seperate token.

[5] The line numbers are not part of the sequence.

4. What modifications to the model do you suggest to improve the accuracy?
5. Which type of regularizations can be applied to your model? Use at least one regularization method in your model for doing the experiments.

4 Model selection

The right way to tune the hyper-parameters of your model is to split your dataset to three disjoint segments, namely: training, validation and test set. However, for the sake of simplicity you can use your training set to tune the parameters but it is strongly encouraged to leave out a portion of each task's training set and use it as the validation set for tuning. You are free to try different initialization of your parameters or optimization algorithms. Our suggestion is to first start with plain stochastic gradient descent with learning rate in the order of 10^{-3}. For the size of the hidden layers of RNNs, you can try [8,16] for the first task and [32,64] for the second task.

5 Submission

The deadline for the assignment is **May 17th, 23:59**. Please submit via Blackboard. Since no late submission is allowed, submitting a partial solution is better than submitting late.

Your submission should include:

- Your source code. Your code should work in two modes of train and test with the necessary inputs passed as command-line arguments or config files. Please include a sufficient set of instructions and examples in a README file.
- Your best trained model.
- The output of your best model on the training and the test set.
- The report. The length of the report should be 7-8 pages (e.g., Times 11pt).

6 Theano environment

For this assignment, you are required to use Theano, a python library for symbolic math calculations. The best resource to make yourself familiarize with Theano is the project's website which provides extensive documentations, tutorials and source code examples. For installing Theano, please consult with its HOWTO page. To handle its dependencies, Anaconda and Jupyter, are two projects that provide most of the dependencies that Theano are built upon, you might consider using either of them for more convenience.

References

[1] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[2] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

[3] Weston, J., Bordes, A., Chopra, S., and Mikolov, T. (2015). Towards AI-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698.