Due Date : March 29th, 2018

Instructions
- *For all questions, show your work!*
- *This part (practical) is to be done in teams of 2 or 3.*
- *Use a document preparation system such as LaTeX.*
- *Submit your answers electronically via the course studium page.*

(65 points) Neural Turing Machines
In this task you will implement the Neural Turing machine as described in [GWD14]. The goal is to better understand attention and memory augmented networks, and the difficulties encountered when training them.

1. **Filling in the Gaps** The paper covers the abstract ideas and goes into detail about how the read/write heads are computed, but does not mention several details. Here, we attempt to cover some of the missing details.

   Important Note : The goal is to fill in the details not covered by the paper with reasonable assumptions.

   (a) The output of the controller at each time step consists of parameters that have constraints. For example, $\beta \in (0, \infty)$. Using equations, describe how you have constrained the output to satisfy them. Briefly justify your choice.

   (b) Present a diagram showing how you think the following are dependent on each other. This will inform your implementation later.
   - the input $\mathbf{x}_t$
   - the memory $\mathbf{M}_{t-1}$ and $\mathbf{M}_t$
   - the output of the read head $\mathbf{r}_{t-1}$ and $\mathbf{r}_t$
   - the erase and add vectors $\mathbf{e}_t$ and $\mathbf{a}_t$
   - the output of the controller $\mathbf{o}_t$

2. **Implement the Neural Turing Machine** Implement both a feedforward (FEEDFORWARD-NTM) and LSTM controller (LSTM-NTM). Also implement an LSTM for the same task (LSTM). This will be the baseline you will compare the performance of the NTM models with.

   - The task is *only* the **copy** task from the paper.
   - **Input data** A sequence of random, 8-dimension binary vectors concatenated with a binary indicator for the end of sequence, with sequences no longer than 20 ($T \leq 20$).

$$(\mathbf{x}_1, \ldots, \mathbf{x}_T, \mathbf{x}_{T+1}),$$

   where

$$\begin{aligned}
\mathbf{x}_t &= (x_{t,1}, x_{t,2}, \ldots, x_{t,8}, 0), & \\
x_{t,i} &\sim \text{Bernoulli}(0.5), & \text{for } t \in \{1, \ldots, T\} \\
\mathbf{x}_{T+1} &= (0, 0, \ldots, 0, 1), & \text{for } t = T + 1
\end{aligned}$$

- Use the cross-entropy loss.
- **For all models :** Use one layer, with a dimension of 100.
- **For the *-NTM models :** Use only 1 read head and 1 write head.

(a) Report the total number of parameters of all the models, including the baseline.

(b) Perform a training hyper-parameter search (learning rate, batch size, etc.) to ensure that the loss converges. (You do not have to perform an exhaustive grid search, just provide the hyper-parameters you eventually used. Hint : You can download the source of the paper here [1], which contains additional comments pertaining to the hyperparameters used.)

Plot the training curves for all three models (using the chosen hyperparameters.)

(c) **Generalisation to longer sequences** One of the benefits of the NTM over a vanilla LSTM is the ability to learn a simple algorithm that generalise to larger sequences.

Test your models on sequences of $T \in \{10, 20, 30, 40, \ldots, 100\}$, with 20 different inputs for each T. Plot average loss vs. T. State what you expected of the experiment and why, then comment on the results.

(d) **Visualising the read and write heads/attention** We can visualise the read and write heads to get an idea of what algorithm is learned for the task.

Plot the write and read head/attention for an input sequence of $T = 10$. State what you expected to see and why, then comment on the results.

(e) **Understanding the shift operator** Discuss the relationship between the shift operator and convolutions (Note that you do not have to implement shifts this way.)

The purpose of this question is to ensure that students understand the code that they are working with. Modify the code-base so that the shift operator only allows forward shifts. In your answer, show the snippet of code before and after the modification.

Références

[GWD14] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv :1410.5401*, 2014.

1. https ://arxiv.org/format/1410.5401