Differentiable Neural Computer Learning Note

Zhengzhong Liang

2018-04-30

1 Ultimate Topics to Study

1.1 What type of task does this agent try to finish

1.2 What is the function of memory in this task

1.3 What tyep of tasks can we train the agent to do

1.4 What parts and functions can we add to this agent

2 Learning Note 20180506

Many ideas of differentiable neural computer is from the work of neural turing machine. So figuring neural turing machine is very helpful to the study of DNC. In neural turing machine, the role of read head or write head is to generate a weight vector.

2.1 The Read Process

The read vector is generated by a linear combination of the vectors in memory block (linear combination is achieved trough a weight vector). And the weight vector is computed according to the read key?

2.2 Addressing Mechanism

The idea of addressing mechansim: determine which memory block to write to and read from. The read and write are finished by calculating the weights. So the calculation of weights is called addressing. Two types of addressing are involved in DNC. One is called content based. The main idea is to compare the cosine similarities between the input vector and memory block. And based on this similarity we can determine which memory block to erase, write or read. Another mechanism is temporal based. It memorize the order of how the memories are modified. For example in some sequences, the indices of memory block that are modified are $1, 3, 4, 5, 2$, then a temporal link of $1, 3, 4, 5, 2$ will be established.

3 Learning Note 20180505

3.1 Agent Task Analysis

3.1.1 What is the agent trying to learn?

This agent tries to learn a pattern with timestep 9 and dimension 6 (acoording to the default print result). This pattern is generated by input sequence, which is tensor with shape $9 \times 16 \times 6$. So I

3.1.2 Is there any law used to generate this pattern?

3.2 Experiments

3.2.1 Try to pring the sequence of each time step of a single sample in the batch

The following variables should be printed: input sequence, output sequence of controller, write heads, memory, usage, read heads, access output. refer to paper, use a simple example to figure out how each part is collaberating with other parts

4 Code Structure

4.1 About Computation Graph

The basic graph is defined in “DNC.py”, some peripheral parts of graph is defined in “run_model”function and “train.py”. The main parts of computation graph of DNC is organized as following:

5 Code Features

5.1 The Structure of Return Value of DNC Core

Table 1: The Sructure and Size of State Tuple

Level 1	Level 2	Level 3	Level 4
DNCState	access_output $(16 \times 4 \times 16)$		
	access_state	memory $(16 \times 16 \times 16)$	
		read_weights $(16 \times 4 \times 16)$	
		write_weights $(16 \times 1 \times 16)$	
		linkage	link $(16 \times 1 \times 16 \times 16)$
			precedence_weights $(16 \times 1 \times 16)$
		usage (16×16)	
	controller_state	hidden (16×64)	
		cell (16×64)	

Table 2: Important Parameters (from Code)

item	value	meaning
hidden size	64	The number of neurons of LSTM (controller)
memory size	16	the number of memory slots
word size	16	the length of each memory slot
num write heads	1	the number of write heads
num read heads	4	the number of read heads

5.2 The Illustration of DNC (Unrolled by Time)

Table 3: Tensor Size and Explanation

tensor name	tensor size	explanation
access output	$16 \times 4 \times 16$	batch size, # of read heads, # of memory slots
memory	$16 \times 16 \times 16$	batch size, # of memory slots, slot length
read weights	$16 \times 4 \times 16$	batch size, # of read heads, # of memory slots
write heads	$16 \times 4 \times 16$	batch size, # of writes heads, # of memory slots
link		
precedence weights		
usage	16×16	batch size, # of memory slots
hidden	16×64	batch size, # of LSTM neurons
cell	16×64	batch size, # of LSTM neurons

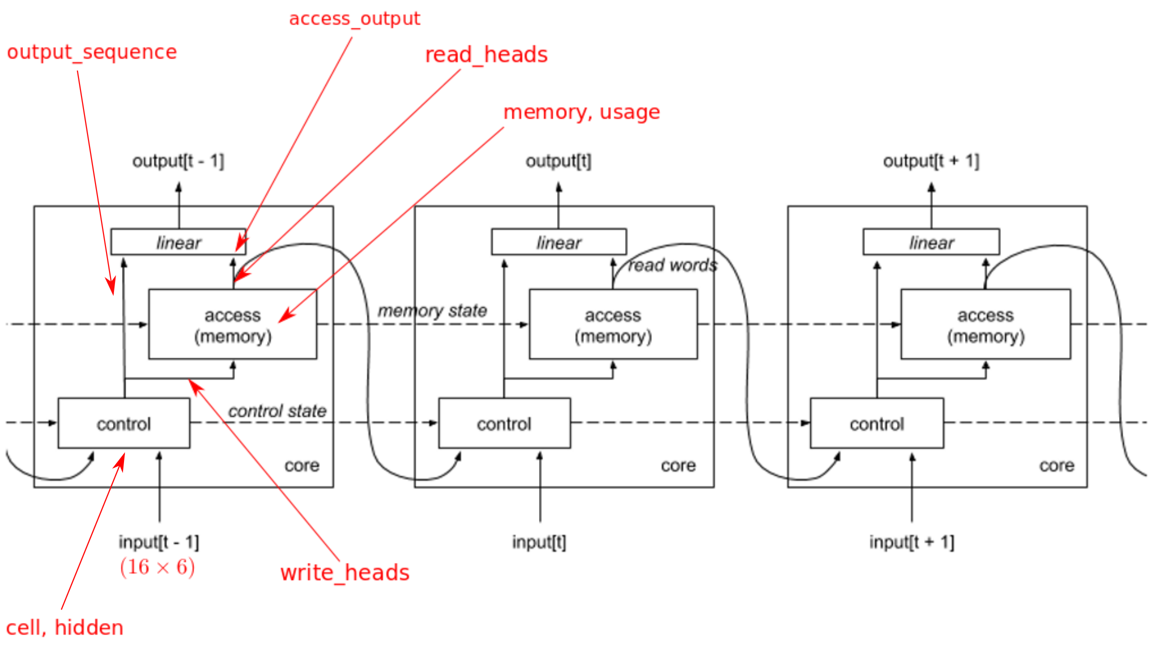

Figure 1: The Computation Graph of DNC

6 Question

6.1 The memory and controller are defined, but are not connected

They are connected in the build function, which is called by dynamic rnn function.

6.2 The build function is not used

They are used in dynamic rnn function.

6.3 The functionality of dynamic rnn

It connects the different parts of the computation graph of DNC, so that a complete graph will be established after calling this function.

6.4 In every loop a new DNC is created?

No, the DNC is only built once.

6.5 How the computation graph is built in tensorflow?

Anything before a session starts is considered as computation graph.

6.6 What does build do in the code

Connects different parts of computations graph. Build function should be orginally a method of snt.AbstractModule. The abstract module is define here.

As one can see, the build function is a method in this class.

6.7 Draw UML for the code

See the UML diagram in the folder.

6.8 Why there is a init function and a build function in each class

init function creates the components of the computation graph, while the build function connects these components. The init function is called when the object of DNC is initiated. The build function is called in the tf.nn.dynamic_rnn function.

6.9 The sequence of calling of build function

DNC, Access, Freeness, ConsineWeights, TemporalLinkage, ConsineWeights.

6.10 Draw the inheritance relation ship (and attributes) starting from AbstractModule

6.11 Figure out how abstract class works in python (e.g. why no init function)

Question solved. The method in abstract class can have nothing but a return function. And inherited class can have the method overriding the method in abstract class.

6.12 Another implementation of DNC

Is available here

6.13 Is the input presented to DNC at each time step?

Yes the input is presented to DNC at each time step.

6.14 What is the timestep of DNC

The timestep of DNC is 9. That is, the DNC is trying to learning a sequence with length 9.

6.15 What is the size of each memory block in DNC memory?

The memory block is 16×16 (16 memory slots, and 16 words each slot).

7 Questions about DNC

7.1 Equivalence between Neural network, Turing machine and brain?

It is believed that RNN is Turing Complete (Turing complete seems mean that something is equivalent to Turing machine.). And it is believed by some people that brain is not more powerful than Turing machine. So that we can conclude that brain$\neq$RNN=Turing Machine.

However, it is questionable then that why we need external memory attacked to RNN if RNN alone is Turing complete.