ARTICLE

doi:10.1038/nature20101

Hybrid computing using a neural network with dynamic external memory

Alex Graves[1]*, Greg Wayne[1]*, Malcolm Reynolds[1], Tim Harley[1], Ivo Danihelka[1], Agnieszka Grabska-Barwińska[1], Sergio Gómez Colmenarejo[1], Edward Grefenstette[1], Tiago Ramalho[1], John Agapiou[1], Adrià Puigdomènech Badia[1], Karl Moritz Hermann[1], Yori Zwols[1], Georg Ostrovski[1], Adam Cain[1], Helen King[1], Christopher Summerfield[1], Phil Blunsom[1], Koray Kavukcuoglu[1] & Demis Hassabis[1]

【强化学习 80】DNC

张楚珩
清华大学 交叉信息院博士在读

DNC 的全称是 differentiable neural computer。

原文传送门

Graves, Alex, et al. "Hybrid computing using a neural network with dynamic external memory." Nature 538.7626 (2016): 471.

特色

2016 年的一篇 nature，主要传达了以下想法。在传统的神经网络中，计算和存储是混在一起的，但是有很多任务中对于存储的需求更大，因此需要一个显式设计一个额外的 memory 来完成存储任务。额外的 memory 能够使得模型记住更为久远的信息（考虑 LSTM 所做的事情），同时它还使得模型能够处理更为复杂和具有结构的任务。为了能够端到端地训练该模型，让模型学会自动地去到 memory 中读写，整个的设计虽然十分精细（繁琐），但是全部可导。文章使用该模型完成了若干比较有挑战性的任务。

过程

1. 整体结构

DNC 的整体结构如下图所示。DNC 整体可以看做一个 RNN 网络，每步获得一个输入，同时经过神经网络的计算并且结合其内部状态，产生一个输出。其独特的地方在于，它显示地维护了一个 memory 空间（如 c 部分所示）。memory 中每一行代表一个条目，它是一个向量。神经网络可以产生一系列的控制指令来对该 memory 进行读写。

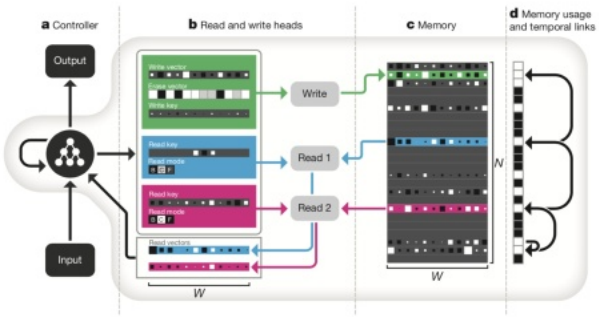

Figure 1 | DNC architecture. a, A recurrent controller network receives input from an external data source and produces output. **b**, **c**, The controller also outputs vectors that parameterize one write head (green) and multiple read heads (two in this case, blue and pink). (A reduced selection of parameters is shown.) The write head defines a write and an erase vector that are used to edit the $N \times W$ memory matrix, whose elements' magnitudes and signs are indicated by box area and shading, respectively. Additionally, a write key is used for content lookup to find previously written locations to edit. The write key can contribute to defining a weighting that selectively focuses the write operation over the rows, or locations, in the memory matrix. The read heads can use gates called read modes to switch between content lookup using a read key ('C') and reading out locations either forwards ('F') or backwards ('B') in the order they were written. **d**, The usage vector records which locations have been used so far, and a temporal link matrix records the order in which locations were written; here, we represent the r[illegible] to using directed arrows.

对于该 memory 的读写操作主要有一下两部分组成：

- Content-based read and write：读写中都会生成相应的查询向量，然后和 memory 中的所有条目做内积，找出最相似的条目进行读写。为了该过程是可导的，相应的操作都会使用 soft 的版本（比如 softmax）。
- Dynamic memory allocation and temporal memory linkage：为了更为高效地利用 memory，当 memory 满了之后，它会寻找重要度低并且不是刚刚才被写入的条目进行覆盖。由于存在这样重复利用的操作，条目之间的近邻关系就不再代表其逻辑上的近邻关系了，因此又设计了 temporal link matrix 来记录条目和条目之间的前后关系。同样，这些复杂的逻辑都被设计成为了可导的形式。

当然整体结构的设计还有各种细节的问题，大家感兴趣可以去看 Method，我反正看完之后感觉整个人都不好了。。。不过核心就是需要设计各个环节能够处理各种逻辑，同时整个过程需要可求导。模型搭建完成之后，可以把它看做一个黑盒子，使用方法和 RNN 类似。

下面讲文章做的几个实验，这些实验还比较有意思。

2. Synthetic question answering

从一百多个词的词库里面人工生成小故事和相应的问题，目标是训练神经网络能够回答相应的问题。一个任务的例子如下所示：

mary journeyed to the kitchen. mary moved to the bedroom. john went back to the hallway. john picked up the milk there. what is john carrying ? - john travelled to the garden. john journeyed to the bedroom. what is john carrying ? - mary travelled to the bathroom. john took the apple there. what is john carrying ? - -

The answers required at the '–' symbols, grouped by question into braces, are

{milk}, {milk}, {milk apple}

每次一个词对应的词向量会输入到 DNC 里面，当输入『-』对应的词向量的时候，DNC 输出的向量会被翻译为相应的词。由于使用了额外的 memory，使得 DNC 具有较强的推理能力，因此效果被文章选中的基准效果更好。

3. Graph

文章测试了 DNC 在一些列和图相关的任务上的表现，如下图所示。

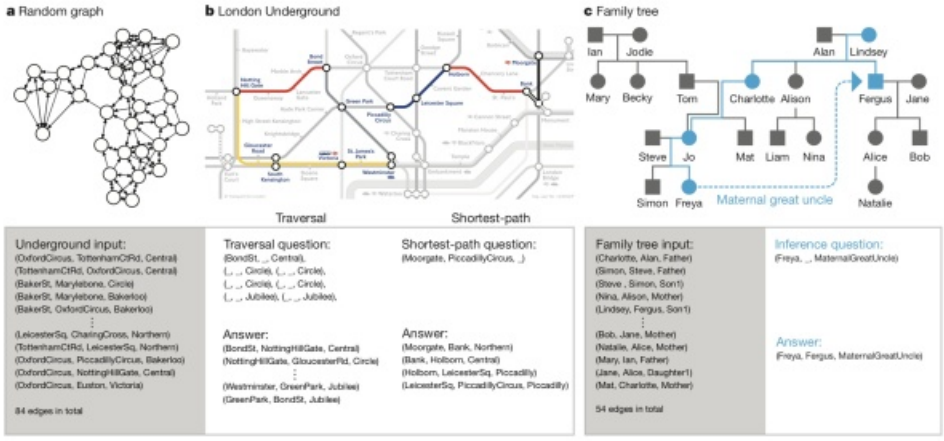

Figure 2 | Graph tasks. a, An example of a randomly generated graph used for training. **b**, Zone 1 interchange stations of the London Underground map, used as a generalization test for the traversal and shortest-path tasks. Random seven-step traversals (an example of which is shown on the left) were tested, yielding an average accuracy of 98.8%. Testing on all possible four-step shortest paths (example shown on the right) gave an average accuracy of 55.3%. **c**, The family tree that was used as a generalization test for the inference task; four-step relations such as the one shown in blue (from Freya to Fergus, her maternal great uncle) were tested, giving an average accuracy of 81.8%. The symbol sequences processed by the network during the test examples are shown beneath the graphs. The input is an unordered list of ('from node', 'to node', 'edge') triple vectors that describes the graph. For each task, the question is a sequence of triples with missing elements (denoted '_') and the answer is a sequence of completed triples.

文章测试了以下三种与图有关的任务：traverse、shortest path、inference。图之间的连边可以表示为 $(source, destination, relationship)$ 。traverse 任务的问题仅告知第一个和最后一个，要求补充 destination，并把这条路径一直往后延续；shortest path 任务的问题告知了前两个，需要 DNC 给出的答案从 source 出发找出最短的路径，直到到达 destination；inference 任务则定义了更多的 relationship（比如在家族树上的 inference 任务可能会定义『外甥女』这种非一步可达的关系），并且输入起始点和关系，要求输出相应的终点。

DNC 在测试的时候，分为以下三到四个阶段：

- Graph description：描述整个图，描述的过程分为若干步，每步只会给出一个边的对应关系；
- Query：给出相应的问题，可能包含若干的时间步骤；
- Planning：可能包含一些空白输入，同时也不要求网络输出的步骤，给网络一定的『思考』时间；
- Answer：给出相应的答案，答案也可能包含若干句，每一句是一个描述网络关系的 triplet。

下图给出了 DNC 完成某个任务的过程的示意图。

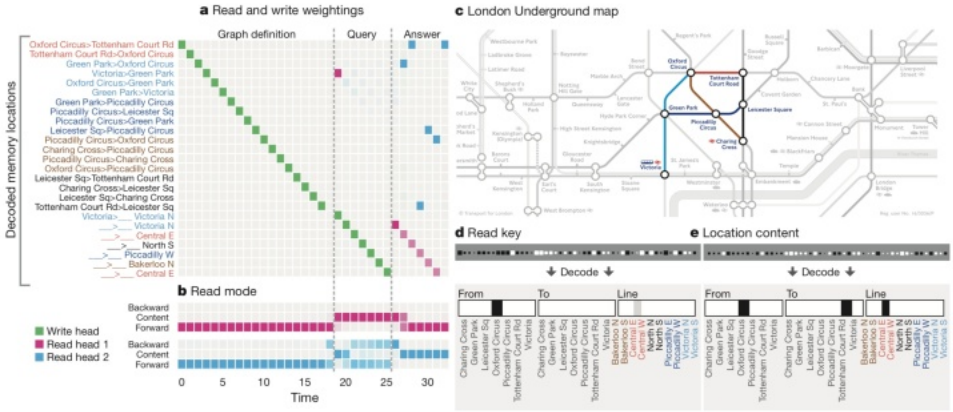

Figure 3 | Traversal on the London Underground. a, During the graph definition phase, the network writes each triple in the map to a separate memory location, as shown by the write weightings (green). During the query phase, the start station (Victoria) and lines to be traversed are recorded. The triple stored in each location can be recovered by a logistic regression decoder, as shown on the vertical axis. **b**, The read mode distribution during the answer phase reveals that read head 1 (pink) follows temporal links forwards to retrieve the instructions in order, whereas read head 2 (blue) uses content lookup to find the stations along the path. The degree of coloration indicates how strongly each mode is used. **c**, The region of the map used. **d**, The final content key used by read head 2 is decoded as a triple with no destination. **e**, The memory location returned by the key contains the complete triple, allowing the network to infer the destination (Tottenham Court Rd).

4. Block puzzle

这里使用 DNC 来解决了一个强化学习问题，该游戏叫做 SHRDLU，规则和汉诺塔差不多，只能把一个栈顶端的方块移动到另一个栈的顶端，目标是需要把原状态转化为目标状态。

该任务也分为三个阶段：第一个阶段描述各种可能的目标状态，每个目标状态包含若干个关系，每个关系描述两个方块之间需要满足的位置关系；第二个阶段指定所要求的目标，即需要它挪动方块以达到前面所描述的哪一个目标；第三个阶段为行动阶段，DNC 每次输出一个行动来挪动方块以完成目标。

该任务如下图所示。

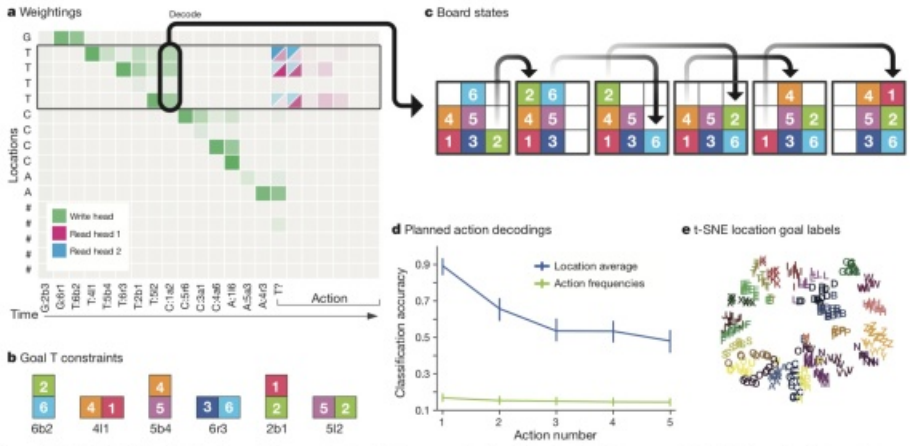

Figure 4 | Mini-SHRDLU analysis. a, In a short example episode, the network wrote goal-related information to sequences of memory locations. ('G', 'T', 'C' and 'A' denote goals; the numbers refer to the block number; 'b,' 'a,' 'l' and 'r' denote 'below', 'above', 'left of' and 'right of', respectively.) The chosen goal was T ('T?'), and the read heads focused on the locations containing goal T. **b**, The constraints comprising goal T. **c**, The policy made an optimal sequence of moves to satisfy its constraints. **d**, On 800 random episodes, the first five actions that the network took for the chosen goal were decoded from memory using logistic regression at the time-step after the goal was written (box in **a** with arrow to **c**). Decoding accuracy for the first action is 89%, compared to 17% using action frequencies alone, indicating that the network had determined a plan at the time of writing, many steps before execution. Error bars represent 5–95 percentile bootstrapped confidence intervals on validation data. **e**, Within trials, we average the location contents associated with each goal label into single vectors. Across trials, we create a dataset of these vectors and perform t-SNE (*t*-distributed stochastic neighbour embedding) dimensionality reduction down to two dimensions. This shows that each goal label is coded geometrically in the memory locations.

该任务使用了 Actor-critic 框架来解决，policy network 和 critic network 各使用了一个 DNC。同时采用了课程学习的方法来逐步学习。学习的效果如下图所示：从第一个图可以看出，使用了 memory 的 DNC 效果比直接使用 LSTM 更好；从第二个图可以看出，总体训练效果也不错。

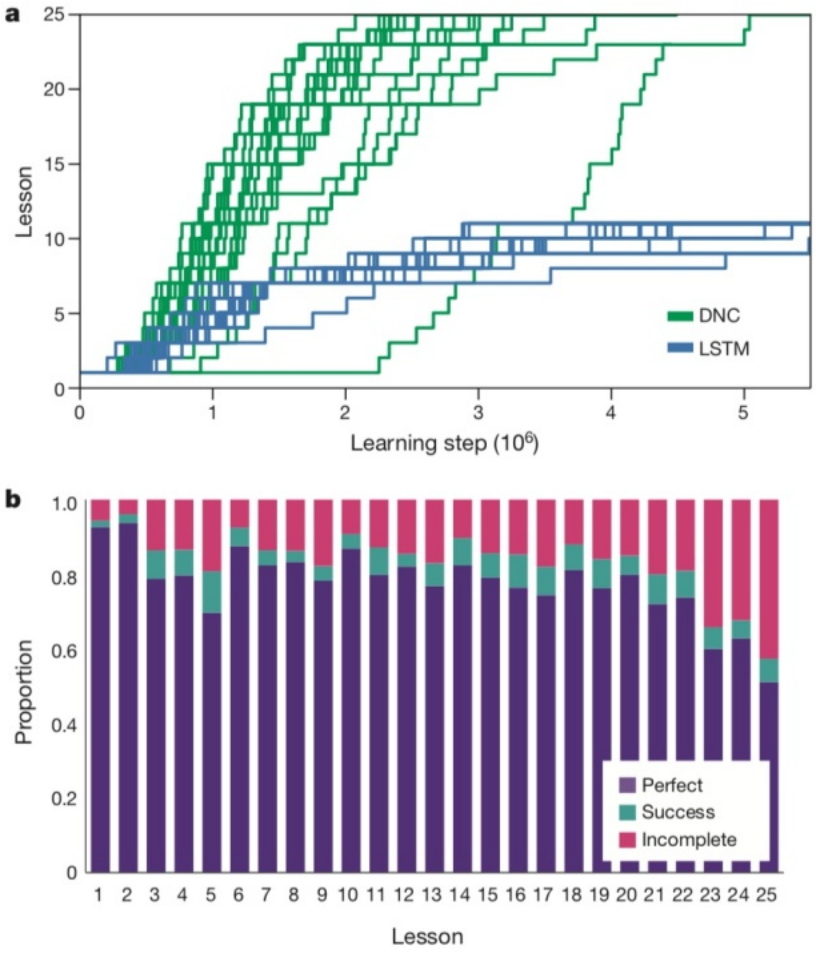

Figure 5 | Mini-SHRDLU results. a, 20 replicated training runs with different random-number seeds for a DNC and LSTM. Only the DNC was able to complete the learning curriculum. **b**, A single DNC was able to solve a large percentage of problems optimally from each previous lesson (perfect), with a few episodes solved in extra moves (success), and some failures to satisfy all constraints (incomplete).

发布于 2019-07-10

强化学习 (Reinforcement Learning)　深度学习（Deep Learning）

文章被以下专栏收录

强化学习前沿
读呀读paper

进入专栏