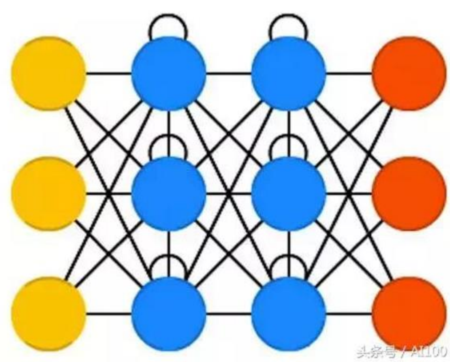
Neural Turing machines

Tim 16th July

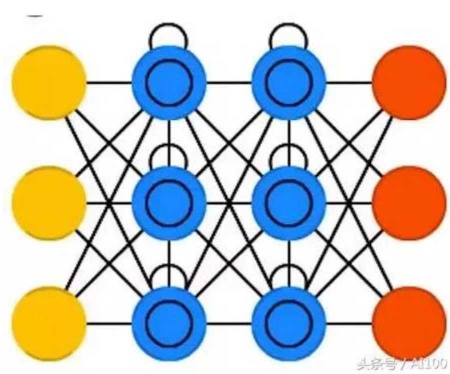
CONTENTS

RNN



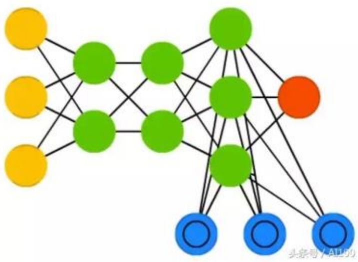
Problem:
Input order will affect the training result of neural network.
The gradient disappears (or the gradient explodes, depending on the activation function used), and the information disappears quickly over time

LSTM



Comparing with the RNN, LSTM has a structure named cell, which includes input gate, forgotten gate and output gate.

NTM



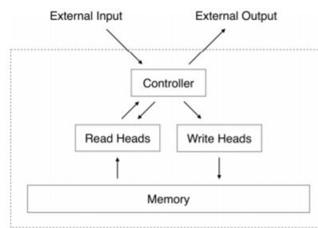


Figure 1: Neural Turing Machine Architecture. During each update cycle, the controller network receives inputs from an external environment and emits outputs in response. It also reads to and writes from a memory matrix via a set of parallel read and write heads. The dashed line indicates the division between the NTM circuit and the outside world.

2013) (Bahdanau et al., 2014) and program search (Hochreiter et al., 2001b) (Das et al., 1992), constructed with recurrent neural networks.

Read

$$\sum_i w_t(i) = 1, \qquad 0 \leq w_t(i) \leq 1, \forall i. \tag{1}$$

$$\mathbf{r}_t \longleftarrow \sum_i w_t(i)\mathbf{M}_t(i), \tag{2}$$

M_t represents the memory matrix in t moment. (N is the number of the row or address, M is the size of each address vector) The reading and writing weight of the W_t head in N addresses at the moment t, and since all weights are normalized, the internal element W_t(i) of the W_t vector is satisfied the equation 1. Then, the value that time t reads R_ t, which can be defined as the vector Mt (i) weighted sum of each address, the equation 2 has shown the weight and memory.

Write

$$\tilde{\mathbf{M}}_t(i) \longleftarrow \mathbf{M}_{t-1}(i)\left[\mathbf{1} - w_t(i)\mathbf{e}_t\right], \qquad (3)$$

$$\mathbf{M}_t(i) \longleftarrow \tilde{\mathbf{M}}_t(i) + w_t(i)\,\mathbf{a}_t. \qquad (4)$$

Inspired by the forget gate and input in LSTM, the write operation can be spited into two parts: erase the (erase) before adding the (add). Given the weight of the write head at the t moment W _ t, and an erasure vector e __ t where M elements are in the range of 0 ~ 1, then the memory vector of the t-1 moment will be adjusted by the formula 3 at t time. After the erase operation, the formula 4 will be computed.

Addressing Mechanisms

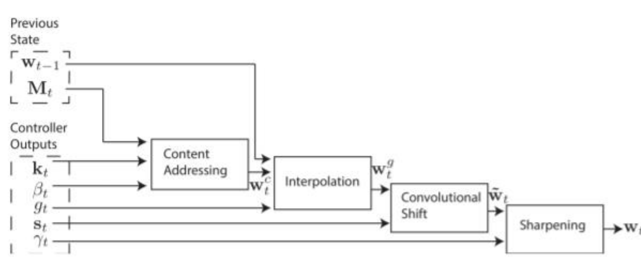


Figure 2: Flow Diagram of the Addressing Mechanism. The *key vector*, $\mathbf{k}_t$, and *key strength*, β_t, are used to perform content-based addressing of the memory matrix, $\mathbf{M}_t$. The resulting content-based weighting is interpolated with the weighting from the previous time step based on the value of the *interpolation gate*, g_t. The *shift weighting*, $\mathbf{s}_t$, determines whether and by how much the weighting is rotated. Finally, depending on γ_t, the weighting is sharpened and used for memory access.

Addressing Mechanisms

Weights are generated by the joint action of two addressing mechanisms and some other complementary mechanisms. The first mechanism, "content-based addressing," determines the degree of focus on memory addresses based on the similarity between the values provided by the controller and the current values. The other way is specified address addressing.

Focusing by Content

$$w_t^c(i) \longleftarrow \frac{\exp\Big(\beta_t K\big[\mathbf{k}_t, \mathbf{M}_t(i)\big]\Big)}{\sum_j \exp\Big(\beta_t K\big[\mathbf{k}_t, \mathbf{M}_t(j)\big]\Big)}. \qquad (5)$$

For content addressing, each reader first produces a M-length key vector K_t, and uses a similarity measure function K [. , .] Each row vector M _ t (i) is compared one by one. Content-based systems generate a normalized weighted list of w _ {t} ^ {c} based on similarity and the strength of key.

$$K\big[\mathbf{u}, \mathbf{v}\big] = \frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}|| \cdot ||\mathbf{v}||}. \qquad (6)$$

The similarity measure function uses cosine similarity.

Focusing by Location

$$\mathbf{w}_t^g \longleftarrow g_t \mathbf{w}_t^c + (1 - g_t)\mathbf{w}_{t-1}. \quad (7)$$

Prior to rotation, Each header also has a scalar representation of interpolation gate g_t, a value from 0 to 1, g is used as the w{ t-1} generated by the header at the moment prior to mixing and the weight list w{ t} ^ {c} generated by the content system at the current moment, and then the (gated) weight list w{ t} ^ {g} after outbound control is derived by formula 7.

Focusing by Location

$$\tilde{w}_t(i) \longleftarrow \sum_{j=0}^{N-1} w_t^g(j)\, s_t(i-j) \quad (8)$$

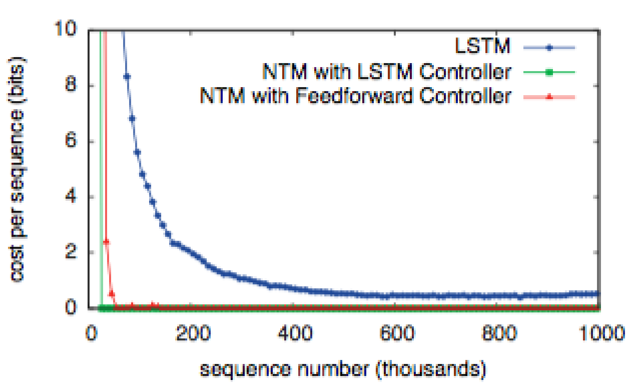
Another way is that let the controller give a single scalar to represent a lower bound of the former uniform distribution. If the memory address is 0 to N-1, use s_t to rotate w_ {t} ^ {g}, which can be represented by the cyclic convolution.

Focusing by Location

$$w_t(i) \longleftarrow \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_j \tilde{w}_t(j)^{\gamma_t}} \qquad (9)$$

If the displacement weight is not sharp, then the convolution operation in formula 8 can cause the weight to diverge with time. To solve this problem, each reader ends up with a scalar y _ t ≥ 1 for the final weight of sharpen.

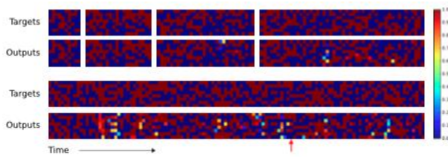
Experiments



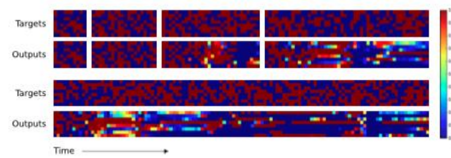
Copy Learning Curves.

Storing and accessing information over long periods of time across domains is a challenge for RNN and other dynamic architectures. The aim of the experiment was to test whether NTM was more competent than LSTM for a longer period of time.

Experiments



NTM



LSTM

The network is trained by a sequence of arbitrary 8-byte vectors with random lengths ranging from 1 to 20. The target sequence is an input copy, with no delimiter. The result is NTM can continue to replicate as the length increases, while LSTM fails quickly after more than 20.

Experiments

```
initialise: move head to start location
while input delimiter not seen do
   receive input vector
   write input to head location
   increment head location by 1
end while
return head to start location
while true do
   read output vector from head location
   emit output
   increment head location by 1
end while
```

pseudocode

The pseudocode shows the information between controller and memory. In terms of data structure, NTM has learned how to create and iterate arrays. Note that the algorithm combines content addressing (skip to start) and address addressing (moving along the sequence).

Conclusion

NTM is fast and efficient, which has the abilities of creation and iteration.

Formula 7 represents the moving depending on the last moment, which means that the long series can be computed.

Formula 9 represents the protection of the weight, as the time goes by.