Neural Turing Machines

KLab group meeting 11/25
Big DATA
Coming 4U

Are You A
Risk To The State?
Neural Turing Machines

Alex Graves  gravesa@google.com
Greg Wayne  gregwayne@google.com
Ivo Danihelka  danihelka@google.com

Google DeepMind, London, UK

Abstract

We extend the capabilities of neural networks by coupling them to external memory resources, which they can interact with by attentional processes. The combined system is analogous to a Turing Machine or Von Neumann architecture but is differentiable end-to-end, allowing it to be efficiently trained with gradient descent. Preliminary results demonstrate that Neural Turing Machines can infer simple algorithms such as copying, sorting, and associative recall from input and output examples.

Goal: “Solve intelligence”
Price tag: $400 million
Building a Learning Machine

Formal model of solving a computational problem

rules + memory

“Learning”
Input-Output mapping ~ rule

Input
Hidden
Output
Turing machine

- What can be computed?

- Computability = instructions that lead to completion of task

http://plato.stanford.edu/entries/turing-machine
Turing machine
Turing machine

1. Tape ("memory")
2. Read and write device ("head")
3. Keeps track of current state ("state register")
4. Instructions
   a. "If machine in state current and tape value is 0, go to state next and move left 1 space"
Turing machine - copy example
Neural networks

- 1950s - Perceptron
- 1969 - Proof that perceptron sucks! (Minsky)
- 1980s - Backpropagation
- 2000s-present - Fast computers, deep
Neural networks

Input | Hidden | Output
--- | --- | ---

L0 (Input) 512x512 → L1 256x256 → L2 128x128 → L3 64x64 → L4 32x32 → F5 → F6 (Output)
Feedforward neural networks

\[ a_j = \sum_{i=1}^{n} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \]

\[ z_j = h(a) = \frac{1}{1 + \exp(-\kappa a)} \]

\[ y_k = h\left(\sum_{j=1}^{q} w_{kj}^{(2)} z_j + w_{k0}^{(2)}\right) \]
Feedforward neural networks

\[ E(w) = \frac{1}{2} \sum_{i=1}^{n} \{ y(x_n, w) - t \}^2 \]

\[ w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w^{(\tau)}) \]
Feedforward neural networks
Feedforward neural networks
Backpropagation

\[ \frac{\partial E}{\partial w_{k,j}^{(2)}} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial w_{k,j}^{(2)}} = (y_k - t_k)z_j \equiv \delta_k z_j \]

For hidden layer -> output layer

\[ \frac{\partial E}{\partial w_{j,i}^{(1)}} = \frac{\partial E}{\partial a_j} \frac{\partial a_j}{\partial w_{j,i}^{(1)}} \equiv \delta_j x_i \quad \delta_j \equiv \frac{\partial E}{\partial a_j} = \sum_k \frac{\partial E}{\partial a_k} \frac{\partial a_k}{\partial a_j} = h'(a_j) \sum_k w_{k,j} \delta_k \]

For input layer -> output layer
Feedforward neural networks

\[ w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w^{(\tau)}) \]

Gradient descent
- Classic
- Conjugate gradient descent
- Stochastic gradient descent
Recurrent neural networks
Recurrent neural networks
Backpropagation through time

Problem: Vanishing/Exploding gradients!
Recurrent neural networks

Long Short-Term Memory

1. Input
2. Input gate
3. “Remember” gate
4. Output gate

Somewhat complicated, lots of parameters
Neural Turing Machines
Neural Turing Machines - memory

N chunks (rows) X M bits each (columns)
Neural Turing Machines - memory

Read from memory ("blurry")

\[ r_t \leftarrow \sum_i w_t(i) M_t(i), \]

Write to memory ("blurry")

\[ \tilde{M}_t(i) \leftarrow M_{t-1}(i) [1 - w_t(i) e_t]. \]
\[ M_t(i) \leftarrow \tilde{M}_t(i) + w_t(i) a_t. \]
Neural Turing Machines - memory

Addressing by content (similarity)

\[ w^c_t(i) \leftarrow \frac{\exp\left(\beta_t K[k_t, M_t(i)]\right)}{\sum_j \exp\left(\beta_t K[k_t, M_t(j)]\right)} \]

\[ K[u, v] = \frac{u \cdot v}{||u|| \cdot ||v||} \]

Addressing by location (shift)

\[ \tilde{w}_t(i) \leftarrow \sum_{j=0}^{N-1} w^g_t(j) s_t(i - j) \]

\[ w_t(i) \leftarrow \frac{\tilde{w}_t(i)^\gamma_t}{\sum_j \tilde{w}_t(j)^\gamma_t} \]
Neural Turing Machines - examples

● NTM can learn to do basic things
  ○ Copy
  ○ Associative recall
  ○ N-gram lookup
  ○ Sorting
● Better than LSTM alone
Neural Turing Machines - copy
Neural Turing Machines - copy
Neural Turing Machines - copy
Neural Turing Machines - copy

The graph illustrates the cost per sequence (bits) over the sequence number (in thousands) for different models:

- LSTM
- NTM with LSTM Controller
- NTM with Feedforward Controller

The graph shows a decreasing trend in cost per sequence as the sequence number increases, with LSTM generally having the lowest cost.
Neural Turing Machines - mult copy

**NTM**

Length 10, Repeat 20

<table>
<thead>
<tr>
<th>Targets</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Targets" /></td>
<td><img src="image2" alt="Outputs" /></td>
</tr>
</tbody>
</table>

Length 20, Repeat 10

<table>
<thead>
<tr>
<th>Targets</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Targets" /></td>
<td><img src="image4" alt="Outputs" /></td>
</tr>
</tbody>
</table>
Neural Turing Machines - mult copy

**LSTM**
Length 10, Repeat 20

<table>
<thead>
<tr>
<th>Targets</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Targets" /></td>
<td><img src="image2" alt="Outputs" /></td>
</tr>
</tbody>
</table>

Length 20, Repeat 10

<table>
<thead>
<tr>
<th>Targets</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Targets" /></td>
<td><img src="image4" alt="Outputs" /></td>
</tr>
</tbody>
</table>

Time
Neural Turing Machines - mult. copy
Neural Turing Machines - ass. recall
Neural Turing Machines - examples
Neural Turing Machines - examples