Optimization for Training Deep Models

Jinhwan Suk

Department of Mathematical Science, KAIST

May 28, 2020

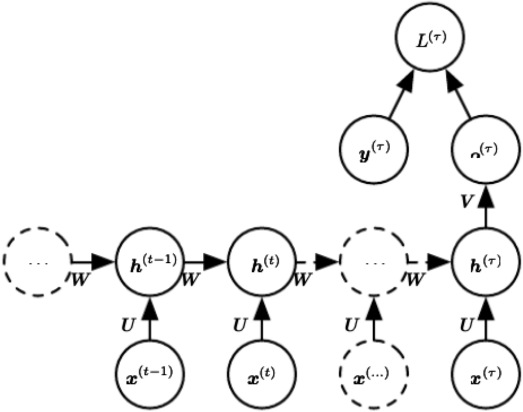
Contents

Contents

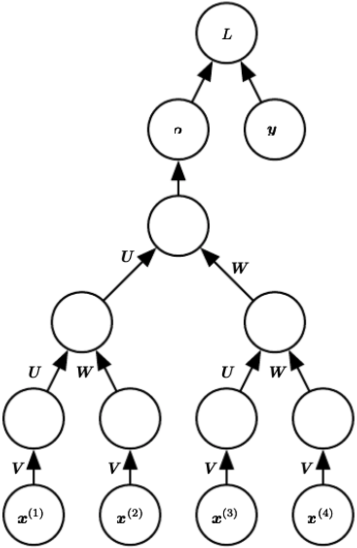
Recursive Neural Network

Recursive Neural Network $\neq$ Recurrent Neural Network (RNN)

Recursive Neural Network



Recursive Neural Network



Recursive Neural Network

The depth can be reduced from $O(\tau)$ to $O(\log \tau)$
$\Rightarrow$ long-term dependency $\downarrow$

Best structure of the tree??

1. fixed structure e.g. Balanced binary tree
2. unfixed structure
   1) use natural language parser 2) the learner to discover

Contents

The Challenge of Long-Term Dependencies

- RNN : $h^{(t)} = \sigma(b + W^T h^{(t-1)} + Ux^{(t)})$, $t = 1, 2, \ldots, \tau$

$$\frac{\partial h^{(t)}}{\partial h^{(t-1)}} = h^{(t)}(1 - h^{(t)})W$$

$$\frac{\partial h^{(t)}}{\partial h^{(t-n)}} = \left(\prod_{t-n+1}^{t} h^{(i)}(1 - h^{(i)}) \right) W^n$$

- If n is large, power of $h^{(t-n)}$ can be extremely big or small.

"The game became interesting as the players warmed up although it was boring for the first half."

Contents

Echo State Networks

Learning parameters : hidden → hidden, hidden → output

only learn the output weights ≈ kernel machine

Q : How do we initialize the weights so that a rich set of a histories can be represented in the output state?

A : spectral radius > 1

⚠ Consider saturation problem

Contents

Leaky Units and Other Strategies for Multiple Time Scales

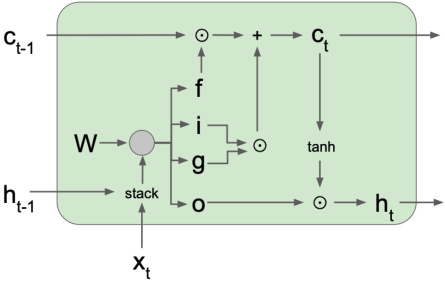
- Adding Skip Connections through Time
- Removing Connections
- Leaky Units and a Spectrum of Different Time Scales

$$h^{(t)} \leftarrow \alpha h^{(t-1)} + (1-\alpha)\sigma(W^T h^{(t-1)} + Ux^{(t)} + b)$$

$\Rightarrow$ Accumulate information

Contents

- Solve vanishing gradient problem
- Once that information has been used,
  it might be useful to forget the old state.
- Mechanism to forget the old state?

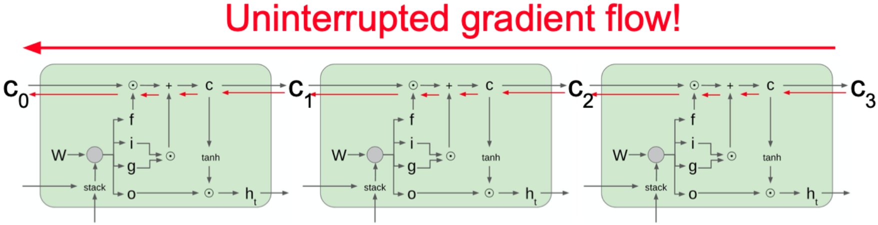


- **Memory cell** c_t is controlled by a forget gate **f** and input gate **g**

$$c^{(t)} = f \odot c^{(t-1)} + g \odot \sigma(b + Wh^{(t-1)} + Ux^{(t)})$$

- The output $h^{(t)}$ is controlled by c_t and output gate **o**

$$h^{(t)} = o \odot \tanh(c^{(t)})$$



$$h^{(t)} = u_t \odot h^{(t-1)} + (1 - u_t) \odot \sigma\left(b + W(r_t \odot h^{(t-1)}) + Ux^{(t)}\right)$$

- **u** stands for "update" gate and **r** for "reset" gate.
- $u_t = \sigma\left(b^u + W^u h^{(t-1)} + U^u x^{(t)}\right)$
- $r_t = \sigma\left(b^r + W^r h^{(t-1)} + U^r x^{(t)}\right)$
- faster computation, less parameter

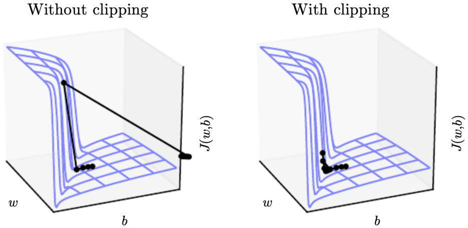
Contents

Optimization for Long-Term Dependencies

- Vanishing and exploding gradient problems
  - Second-order method, BFGS
  - Nesterov momentum + careful initialization
  - **LSTM + SGD**

Optimization for Long-Term Dependencies

- Clipping Gradients



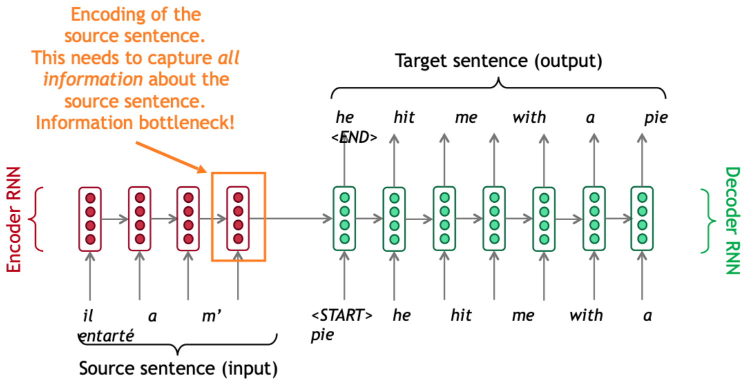
if $\|g\| > v$, then $g \leftarrow \frac{gv}{\|g\|}$

- Regularization term

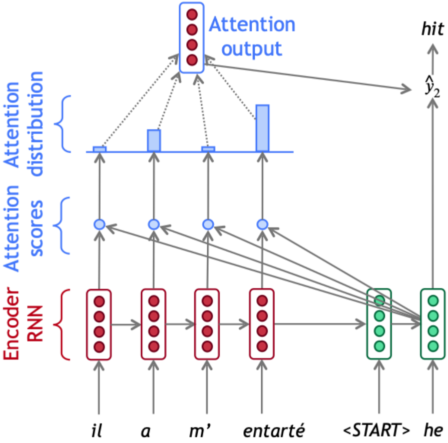
$$\Omega = \sum_t \left(\frac{\left\| (\nabla_{h^{(t)}} L) \frac{\partial h^{(t)}}{\partial h^{(t-1)}} \right\|}{\left\| \nabla_{h^{(t)}} L \right\|} - 1 \right)^2$$

Contents

Attention model



Core Idea : on each step of the decoder, use direct connection to the encoder to **focus** on a particular part of the source sequence



- Non-recurrent sequence model
- Instead, add "positional encodings" to the input embedding
- Self-attention

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

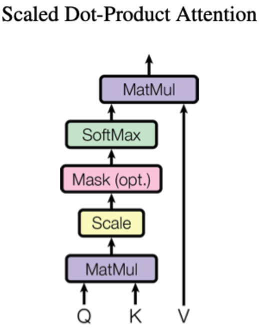
- Multi-Head Attention

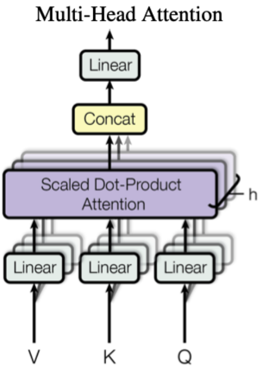
$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^o$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

Attention model

Attention is all you need (Transformer)





Attention model

Attention is all you need (Transformer)