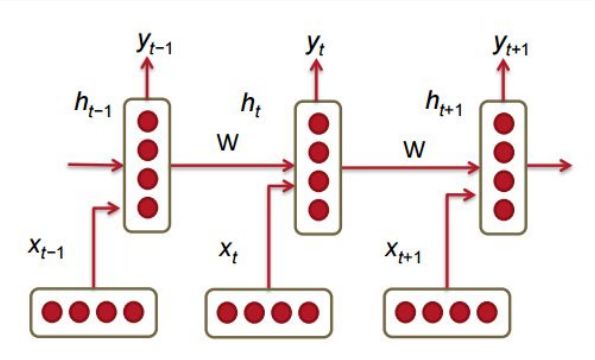
Recurrent Neural Networks

Deep Learning Reading Group
April 14, 2017
Yinong Wang

What are Recurrent Neural Networks (RNNs)?

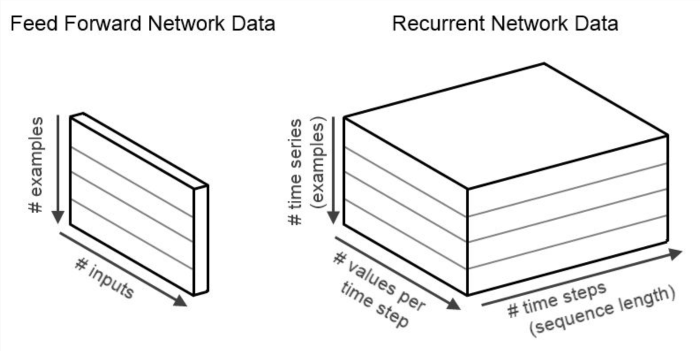
- RNNs are a type of neural network that share parameters across different positions/indices of time



http://cs224d.stanford.edu/lecture_notes/notes4.pdf

RNNs are useful for sequences

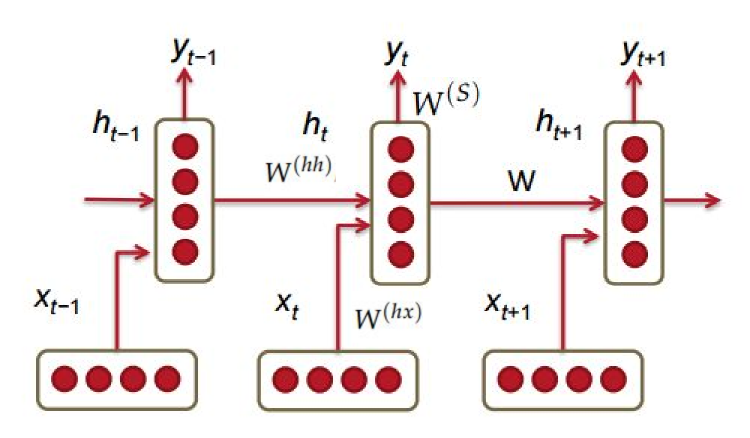
- Unlike standard NNs and CNNs, RNNs introduce a temporal element to the system
- Whereas CNNs introduce the idea of a spatial dependence, RNNs rely on the idea that the current input relies on previous inputs



https://deeplearning4j.org/usingrnns

RNN architecture

- Current input: x_t
- Current hidden state: h_t
- Nonlinear activation function: σ, or tanh
- Input-to-hidden weights: $W^{(hx)}$
- Hidden-to-hidden weights: $W^{(hh)}$
- Hidden-to-output weights: $W^{(S)}$
- Predicted output: $\hat{y}_t$



$$h_t = \sigma(W^{(hh)}h_{t-1} + W^{(hx)}x_{[t]})$$

$$\hat{y}_t = softmax(W^{(S)}h_t)$$

http://cs224d.stanford.edu/lecture_notes/notes4.pdf

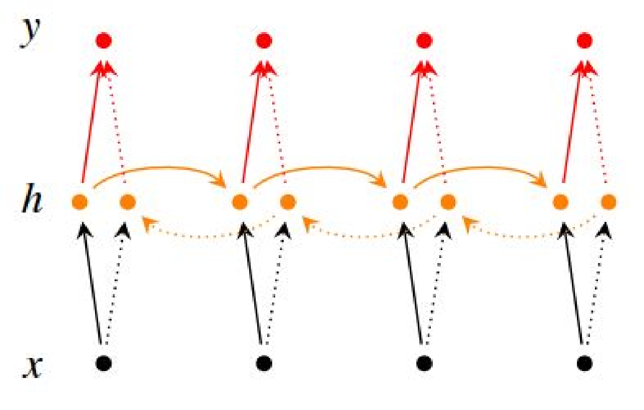
Bidirectional RNNs

- Looks at combination of previous and future inputs

$$\overrightarrow{h}_t = f(\overrightarrow{W} x_t + \overrightarrow{V}\,\overrightarrow{h}_{t-1} + \overrightarrow{b})$$

$$\overleftarrow{h}_t = f(\overleftarrow{W} x_t + \overleftarrow{V}\,\overleftarrow{h}_{t+1} + \overleftarrow{b})$$

$$\hat{y}_t = g(Uh_t + c) = g(U[\overrightarrow{h}_t; \overleftarrow{h}_t] + c)$$



http://cs224d.stanford.edu/lecture_notes/notes4.pdf

The Vanishing (Exploding) Gradient Problem

Sentence 1

"Jane walked into the room. John walked in too. Jane said hi to John"

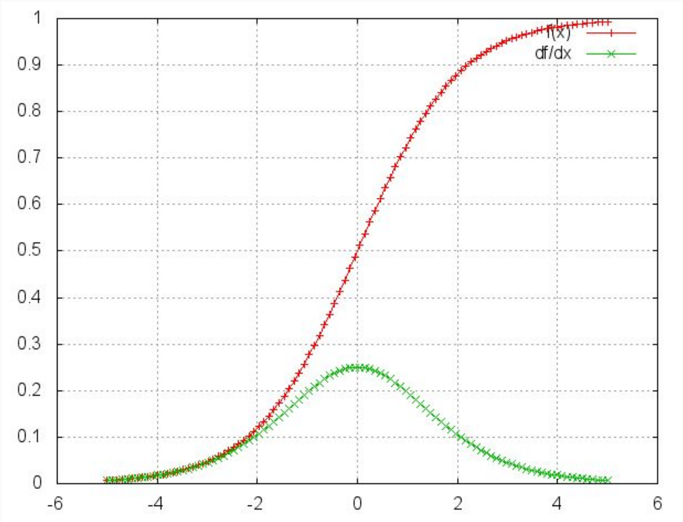
Sentence 2

"Jane walked into the room. John walked in too. It was late in the day, and everyone was walking home after a long day at work. Jane said hi to ___"

Exploring the Gradient Problem

- Saturation of neurons leads to derivatives of activation functions to be close to 0
- Gradient contribution from “far away” steps become 0
- Difficult to learn long-term dependencies
- Solutions:
    - clipping the gradient
    - ReLU activations
    - proper initialization of W
    - regularization
    - LSTMs

$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \sum_{k=1}^{t} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \boxed{\left(\prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} \right)} \frac{\partial h_k}{\partial W}$$



http://www.wildml.com/2015/10/recurrent-neural-networks-tutorial-part-3-backpropagation-through-time-and-vanishing-gradients/

Long Short-Term Memory units (LSTMs)

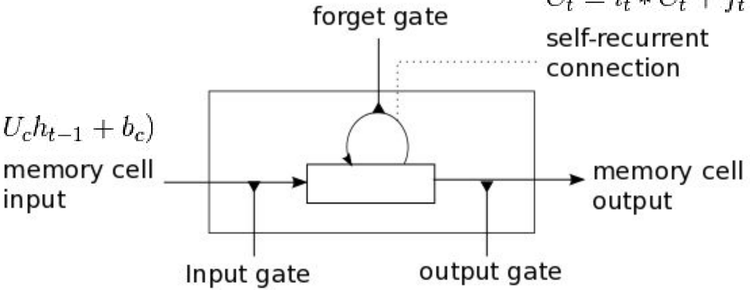
- Introduces gates: input, output, and forget
- Memory cell state

$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$

$C_t = i_t * \widetilde{C}_t + f_t * C_{t-1}$

$\widetilde{C}_t = tanh(W_c x_t + U_c h_{t-1} + b_c)$

$h_t = o_t * tanh(C_t)$



$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$

$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o C_t + b_o)$

Figure 1 : Illustration of an LSTM memory cell.

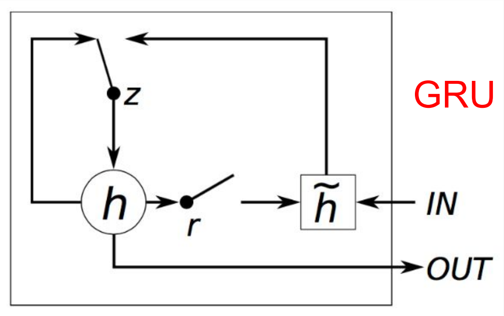
- x_t is the input to the memory cell layer at time t
- $W_i, W_f, W_c, W_o, U_i, U_f, U_c, U_o$ and V_o are weight matrices
- b_i, b_f, b_c and b_o are bias vectors

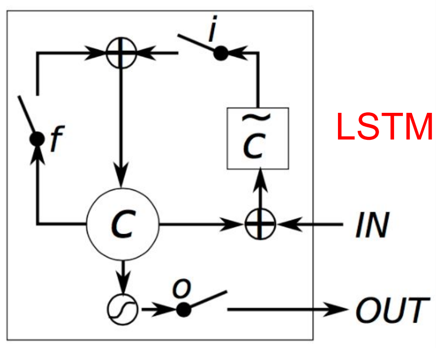
http://deeplearning.net/tutorial/lstm.html

Other types of RNNs

- Gated Recurrent Unit (GRU)
    - Simplified variant of LSTM
    - 2 gates, update (combination of input and output gates), and reset (similar to forget)
    - No internal memory
    - No additional nonlinear activation applied when computing output

$$z = \sigma(x_t U^z + s_{t-1} W^z)$$
$$r = \sigma(x_t U^r + s_{t-1} W^r)$$
$$h = tanh(x_t U^h + (s_{t-1} \circ r) W^h)$$
$$s_t = (1 - z) \circ h + z \circ s_{t-1}$$

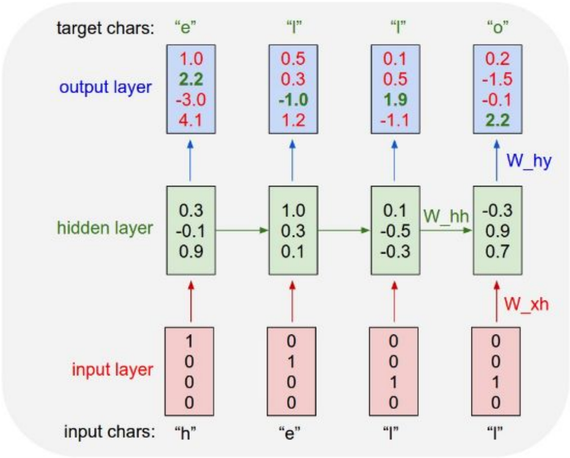




http://www.wildml.com/2015/10/recurrent-neural-network-tutorial-part-4-implementing-a-grulstm-rnn-with-python-and-theano/

Applications of RNNs

- Character level language models

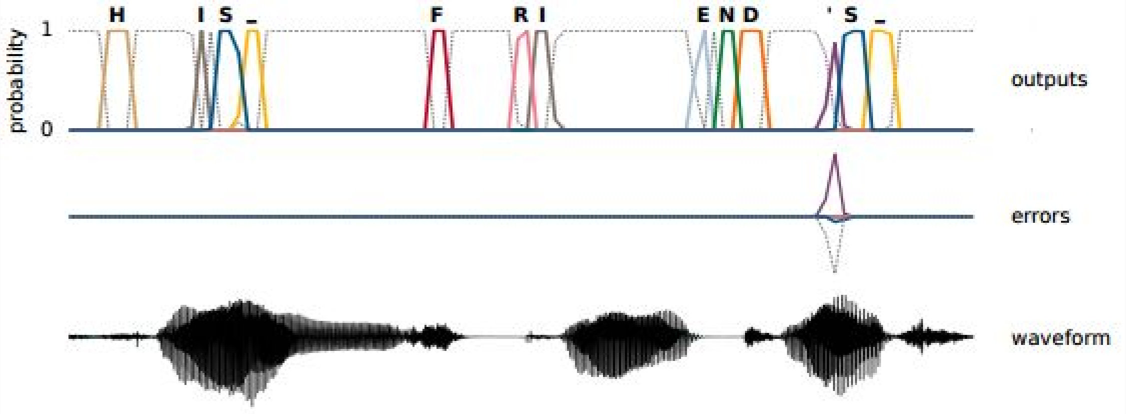


An example RNN with 4-dimensional input and output layers, and a hidden layer of 3 units (neurons). This diagram shows the activations in the forward pass when the RNN is fed the characters "hell" as input. The output layer contains confidences the RNN assigns for the next character (vocabulary is "h,e,l,o"); We want the green numbers to be high and red numbers to be low.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Applications of RNNs

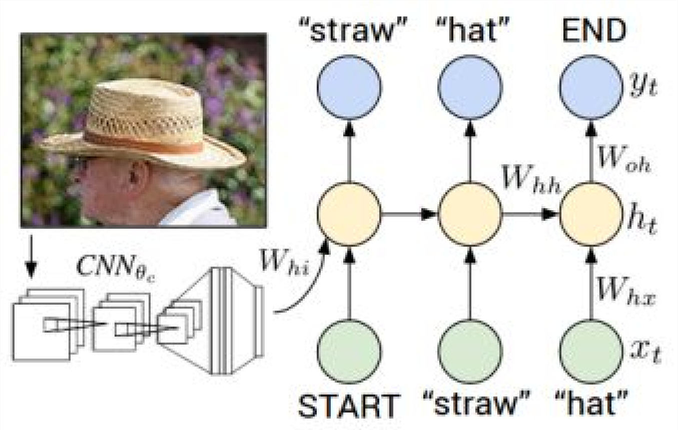
- "Towards End-To-End Speech Recognition with Recurrent Neural Networks"
- Bidirectional LSTM

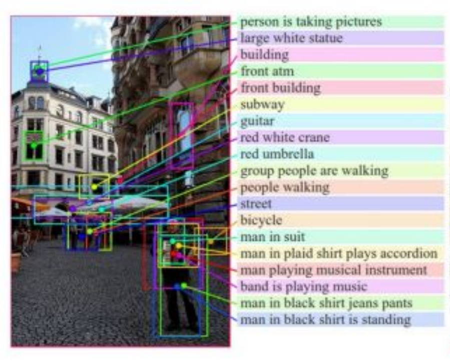


Graves and Jaitly, ICML (2014)

Combining RNNs and CNNs

- “Deep Visual-Semantic Alignments for Generating Image Descriptions”





Karpathy and Fei-Fei, CVPR (2015).

Trying out LSTMs

- IMDB dataset for sentiment analysis:
  - Collection of polarizing movie reviews from the Internet Movie Database (IMDB) website
  - Classification task for positive or negative movie review
  - https://github.com/fchollet/keras/blob/master/examples/imdb_lstm.py