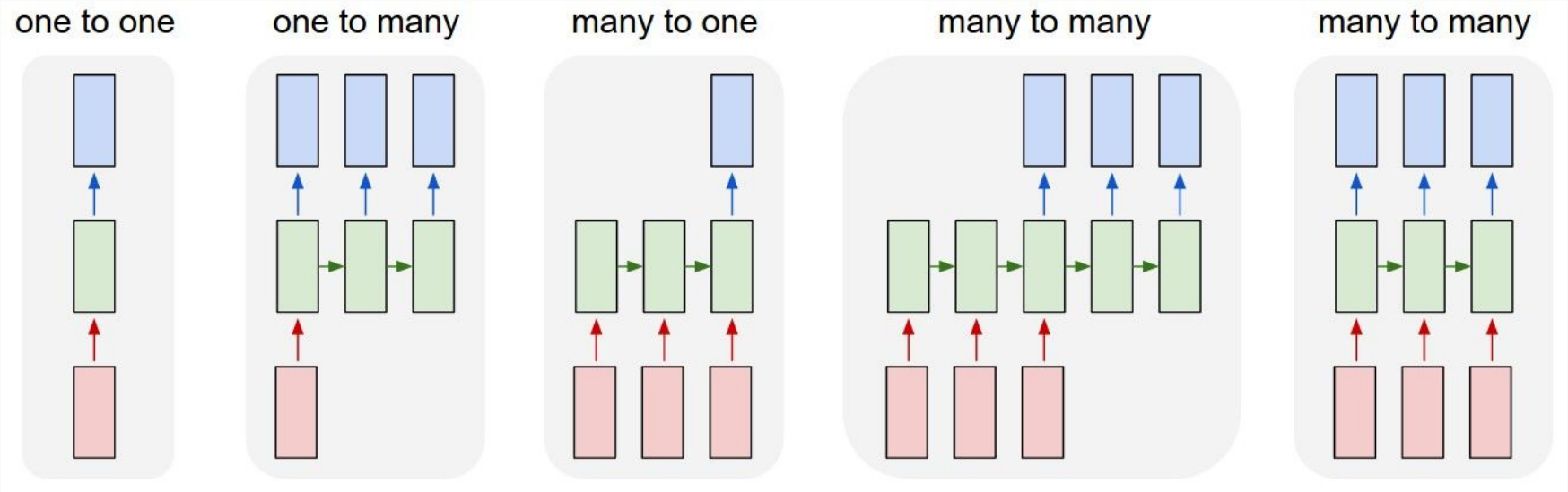
Recurrent Neural Networks

Michał Filipiuk

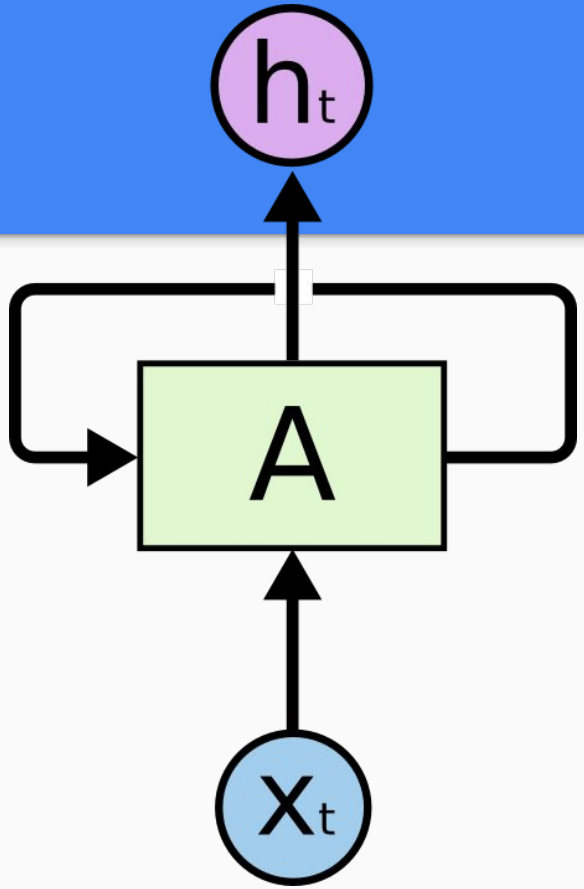
Uwaga: Nie będzie dużo matmy!

Po co te RNNy?

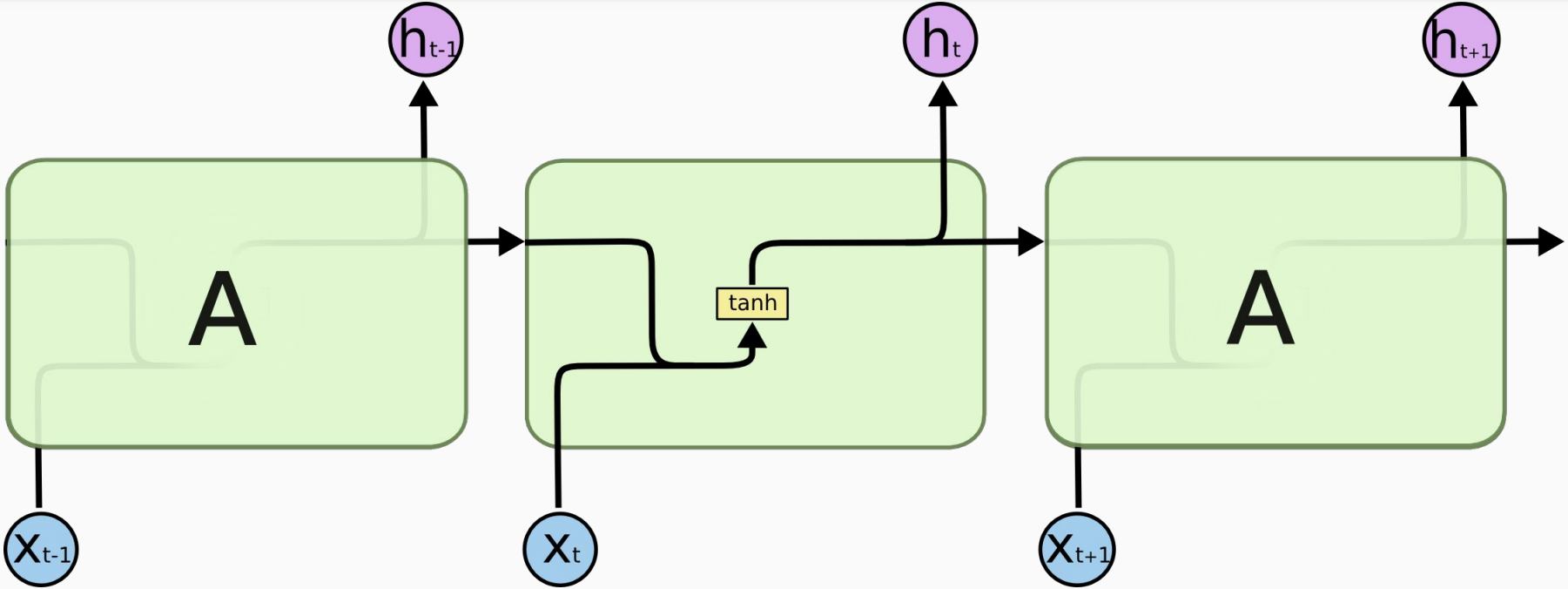


Trochę historii

- Pierwsze sieci RNNy pojawiają się już w 1986
- Niestety początkowo są tylko naukową ciekawostką z powodu swojej kapryśności
- Naukowcy tworzą dziesiątki różnych implementacji, ale tylko dwie z nich dożywają “dorosłości”:
  - Long short-term memory (1997)
  - Gated recurrent unit (2014)

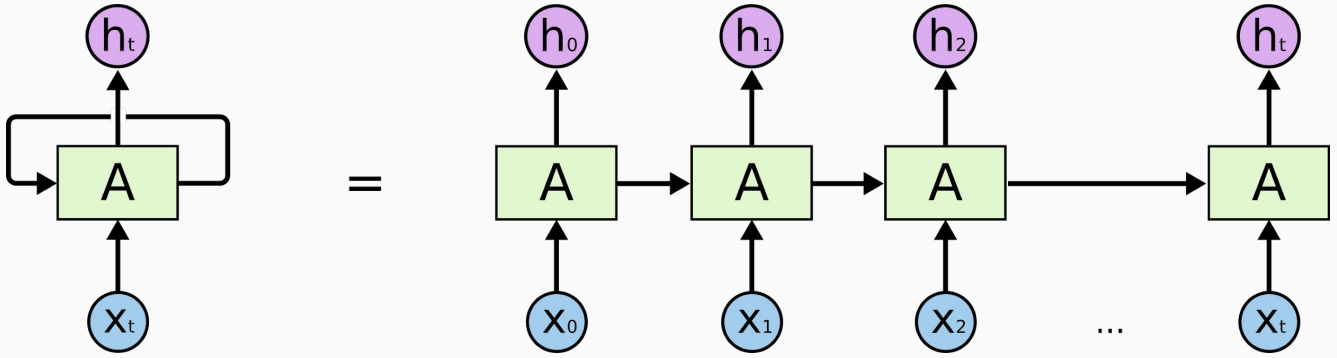


Vanilla RNN



Jak to się uczy?

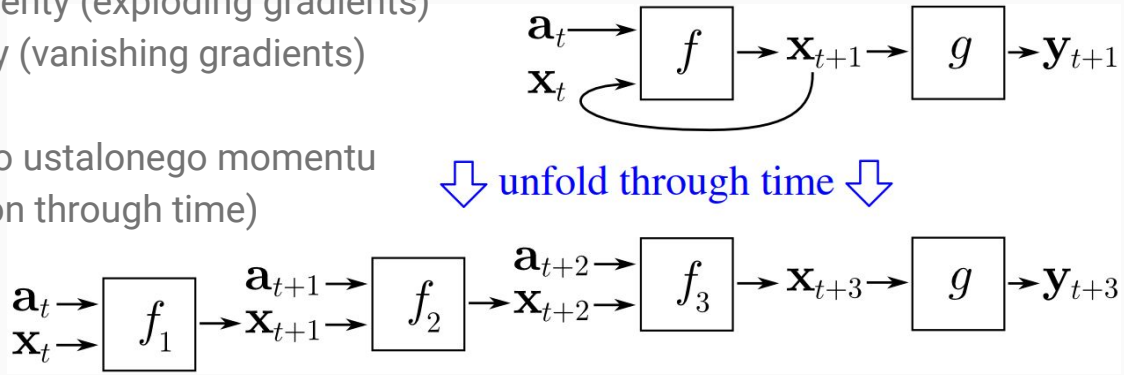
- “Rozwinięty” RNN jest skierowanym, grafem acyklicznym
- W związku z tym możemy dokonać backpropagacji jak w każdej sieci neuronowej! Super?



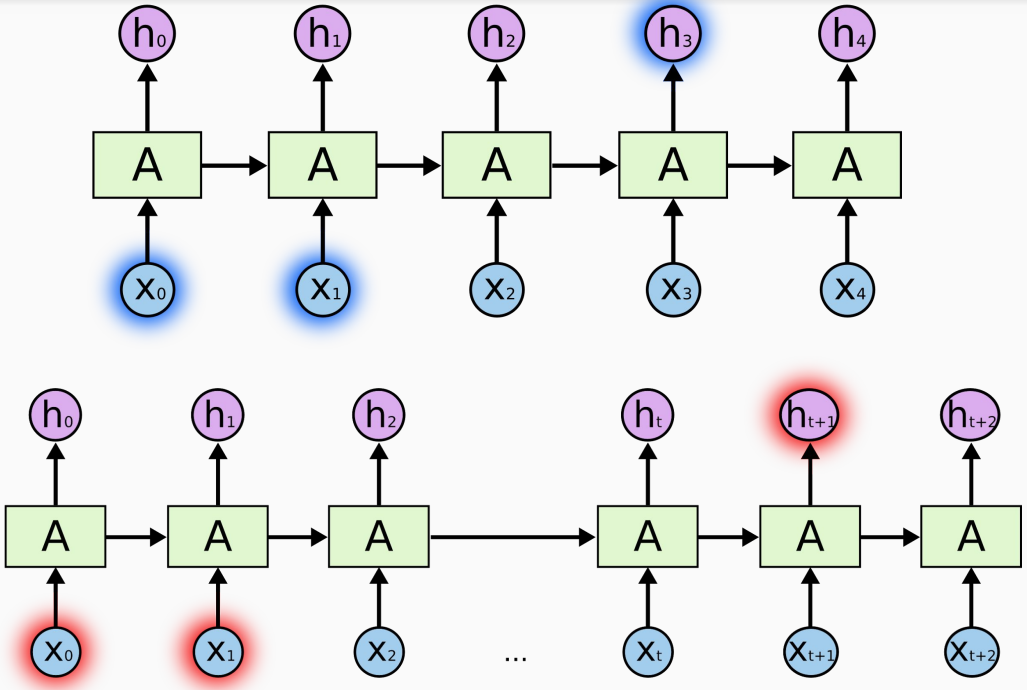

Well yes, but actually no

Jak to się uczy?

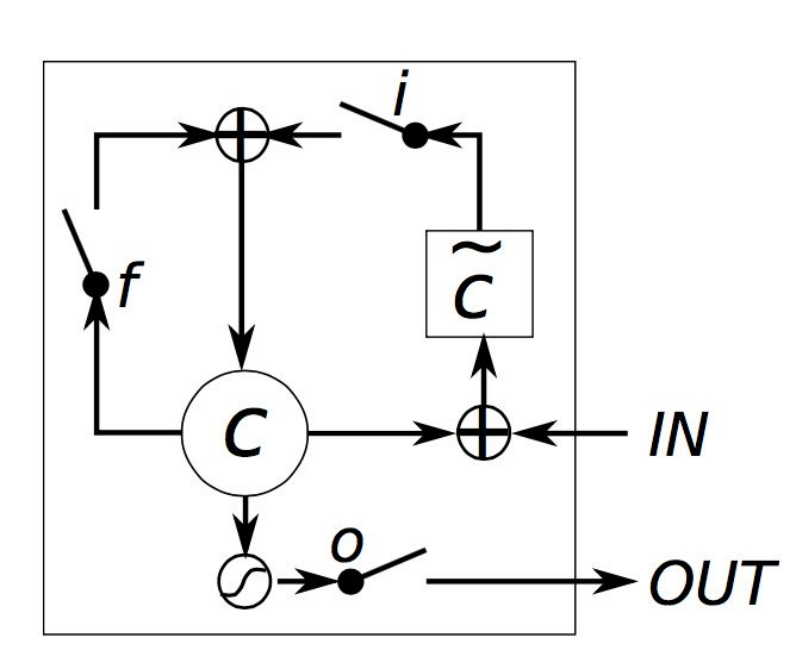
- “Rozwinięty” RNN jest skierowanym, acyklicznym grafem (w skrócie DAG)
- W związku z tym możemy dokonać backpropagacji jak w każdej sieci neuronowej! Super?
  - Rozwijanie RNNów dla długich sekwencji powoduje duże problemy z gradientami:
    - “Eksplodujące” gradienty (exploding gradients)
    - “Znikające” gradienty (vanishing gradients)
  - Rozwiązanie?
    Rozwijać tylko do pewnego ustalonego momentu (truncated backpropagation through time)

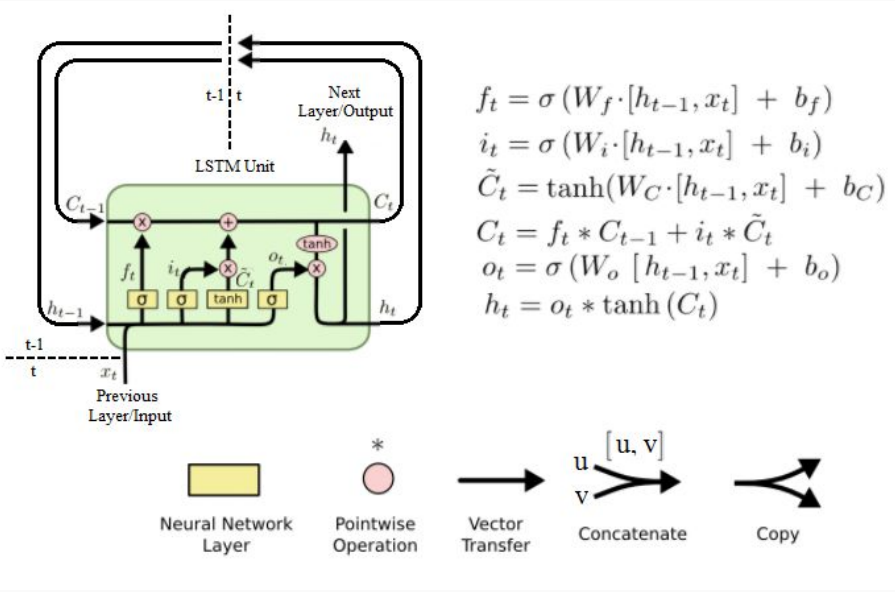


Jak to się uczy?

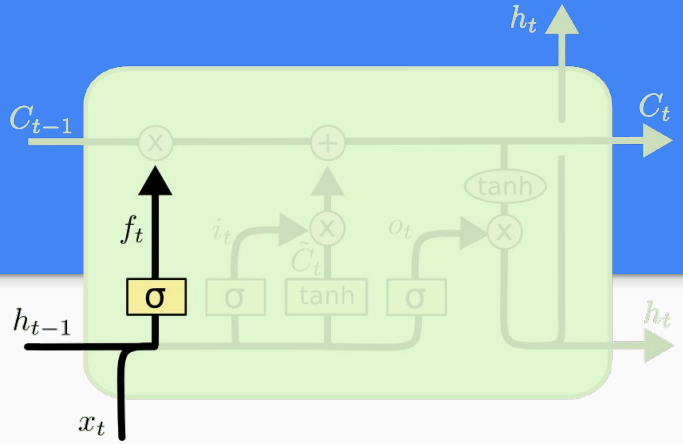


Long short-term memory

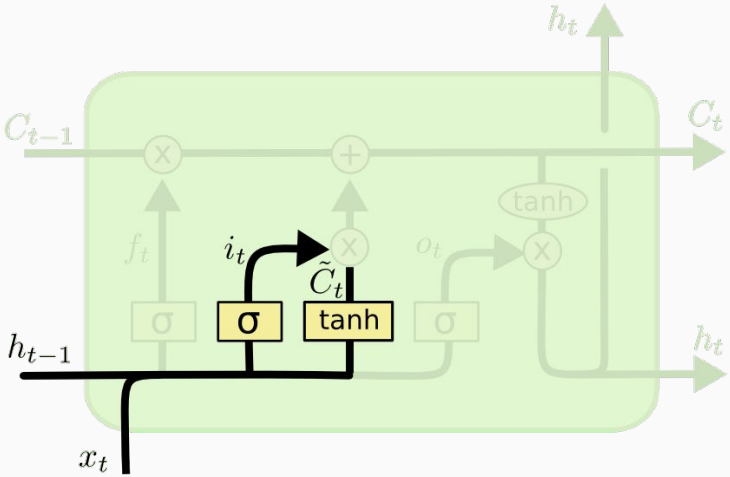




$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

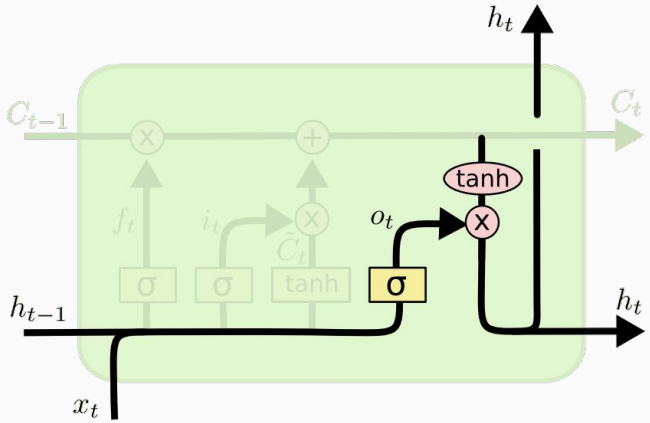


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$



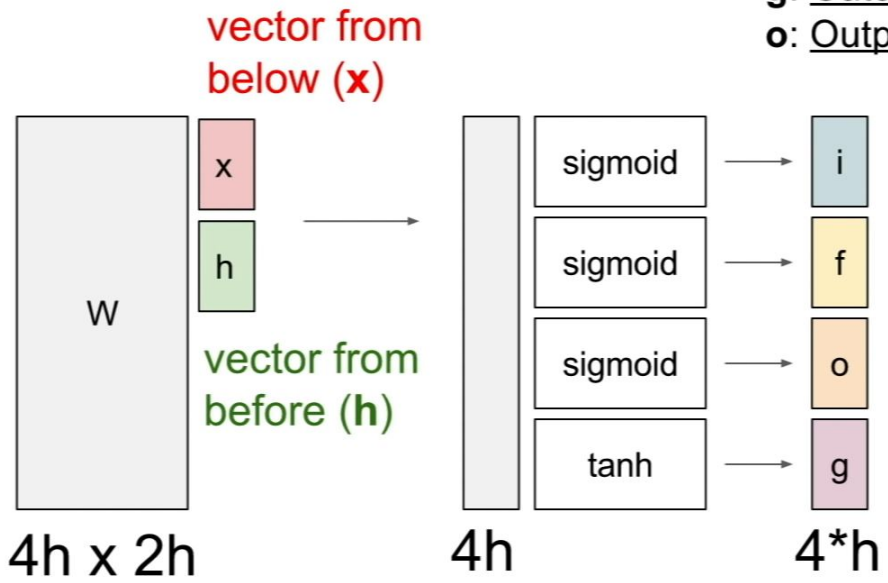
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]

f: Forget gate, Whether to erase cell
i: Input gate, whether to write to cell
g: Gate gate (?), How much to write to cell
o: Output gate, How much to reveal cell

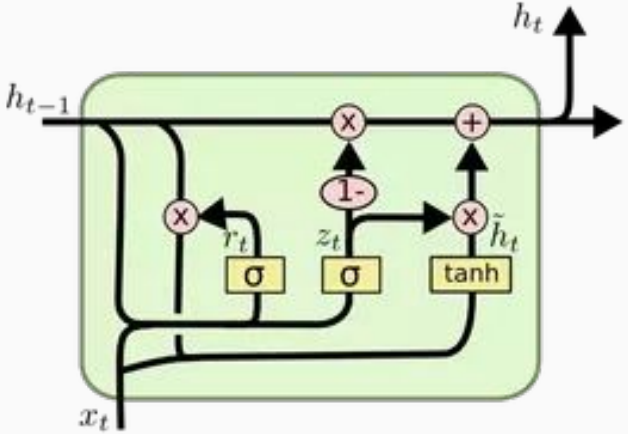


$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Gated recurrent unit



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

GRU vs LSTM

Argumenty za LSTM:

- Większa moc wyrazu (choć nieznacznie)

Argumenty za GRU:

- Lżejsze w uczeniu
- Łatwiej uogólnia się

Dziękuję za uwagę

Bibliografia

- http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/
- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- https://en.wikipedia.org/wiki/Backpropagation_through_time
- https://en.wikipedia.org/wiki/Recurrent_neural_network
- https://en.wikipedia.org/wiki/Long_short-term_memory
- https://brohrer.github.io/how_rnns_lstm_work.html
- https://www.youtube.com/watch?v=6niqTuYFZLQ