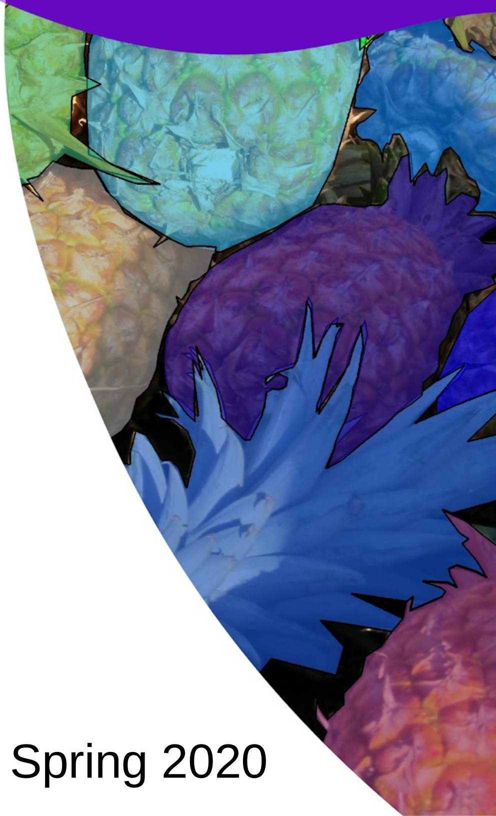
NEW YORK UNIVERSITY

Deep Learning Architectures

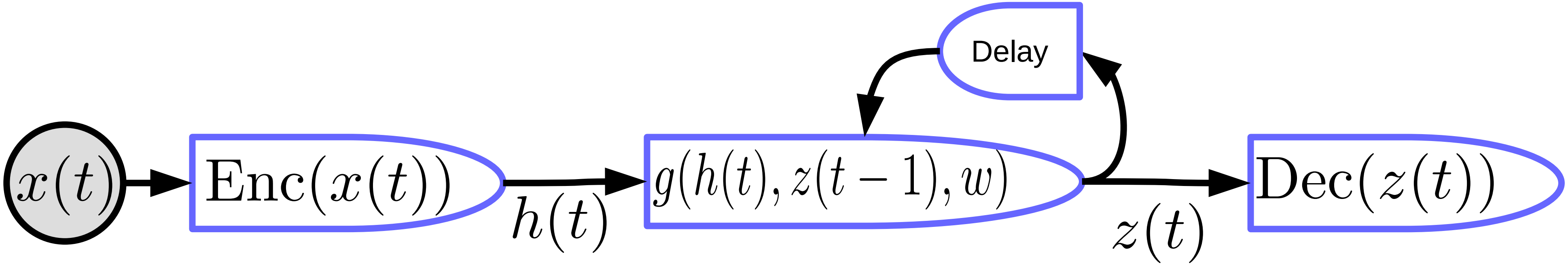
http://bit.ly/DLSP20

Yann LeCun
NYU - Courant Institute & Center for Data Science
Facebook AI Research
http://yann.lecun.com
TAs: Alfredo Canziani, Mark Goldstein



Recurrent Networks

- **Networks with loops. For backprop, unroll the loop.**



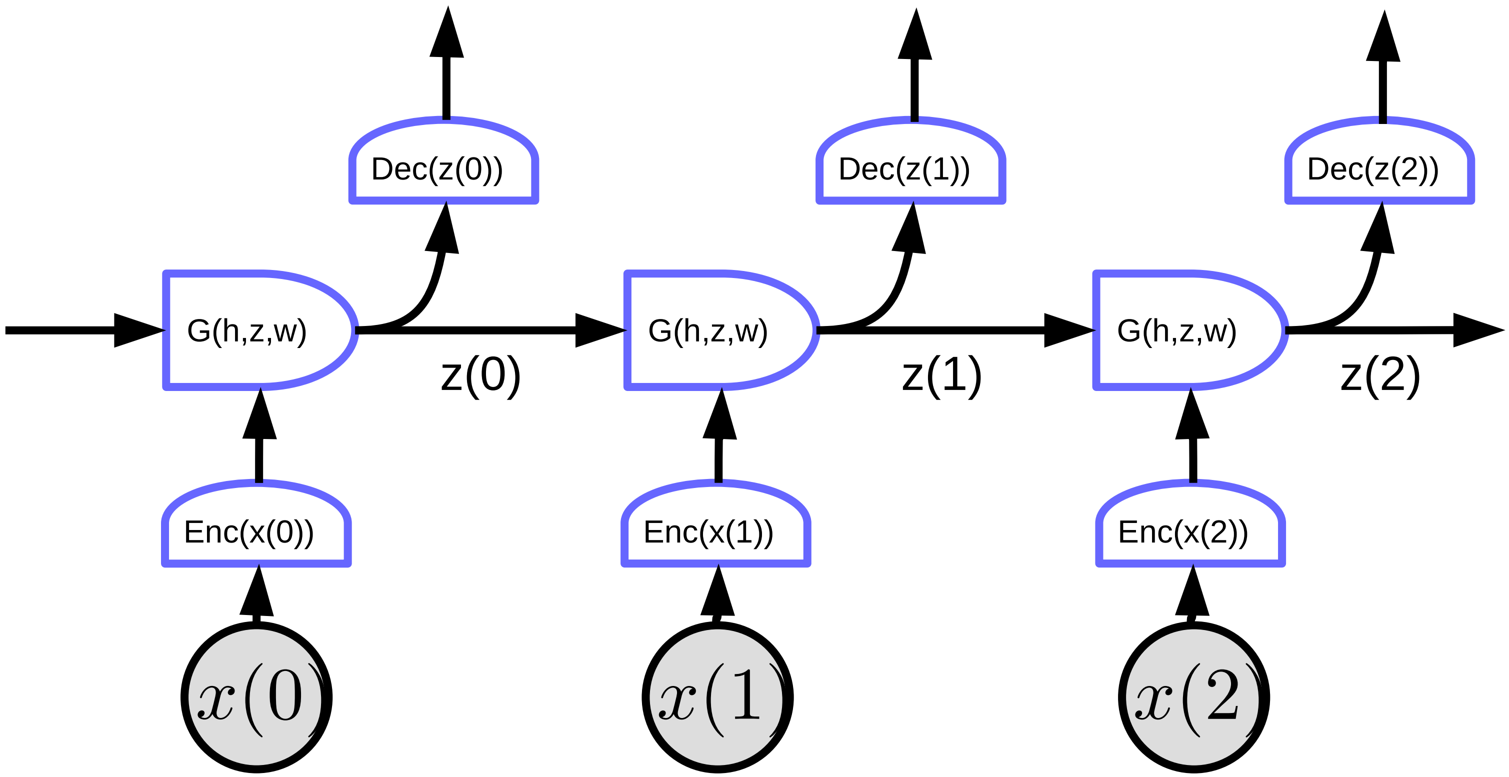
$$h(t) = \mathrm{Enc}(x(t))$$

$$z(t) = g(h(t), z(t-1), w)$$

$$y(t) = \mathrm{Dec}(z(t))$$

Recurrent Networks

- **Networks with loops. For backprop, unroll the loop.**



RNN tricks

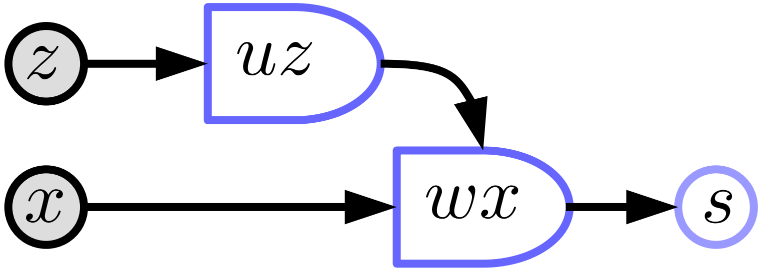
- [Pascanu, Mikolov, Bengio, ICML 2013; Bengio, Boulanger & Pascanu, ICASSP 2013]

- Clipping gradients (avoid exploding gradients)
- Leaky integration (propagate long-term dependencies)
- Momentum (cheap 2nd order)
- Initialization (start in right ballpark avoids exploding/vanishing)
- Sparse Gradients (symmetry breaking)
- Gradient propagation regularizer (avoid vanishing gradient)
- LSTM self-loops (avoid vanishing gradient)

Multplicative Modules

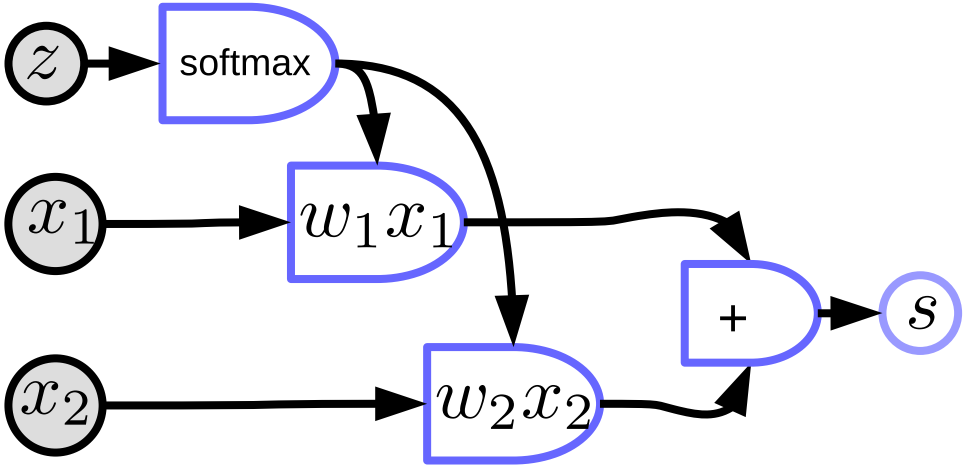
- **Quadratic layer, product units, Sigma-Pi units**

$$s_i = \sum_j w_{ij} x_j \ \text{with} : w_{ij} = \sum_k u_{ijk} z_k \ ; \text{equiv} : s_i = \sum_{jk} u_{ijk} z_k x_j$$



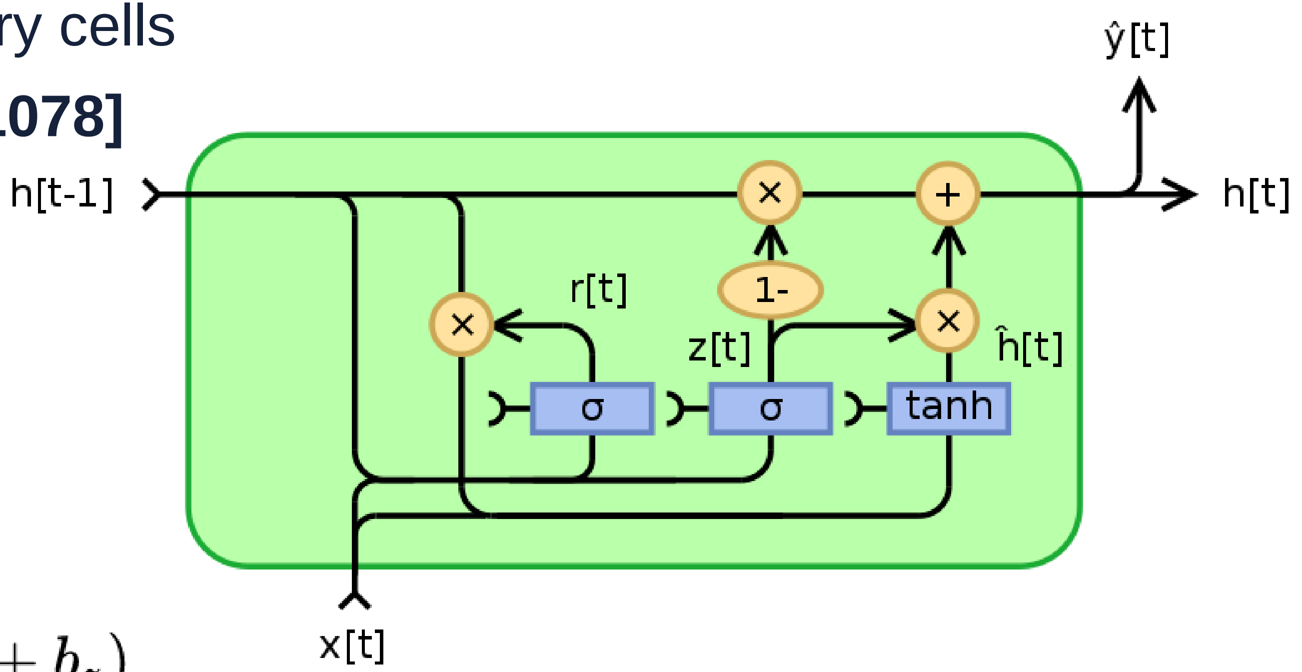
- **Attention module**

$$s_i = \sum_j w_j x_{ij} \qquad w_j = \frac{e^{z_j}}{\sum_k e^{z_k}}$$



GRU (Gated Recurrent Units)

- **Recurrent nets quickly "forget" their state**
  - Solution: explicit memory cells
- **GRU [Cho arXiv:1406.1078]**



x_t: input vector

h_t: output vector

z_t: update gate vector

r_t: reset gate vector

W, U and b: parameter

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z)$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r)$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \phi_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$

LSTM (Long Short-Term Memory)

- **Recurrent nets quickly "forget" their state**
  - Solution: explicit memory cells
- **LSTM [Hochreiter & Schmidhuber 97]**

$x_t \in \mathbb{R}^d$: input vector to the LSTM unit

$f_t \in \mathbb{R}^h$: forget gate's activation vector

$i_t \in \mathbb{R}^h$: input/update gate's activation vector

$o_t \in \mathbb{R}^h$: output gate's activation vector

$h_t \in \mathbb{R}^h$: hidden state vector also known as output

$c_t \in \mathbb{R}^h$: cell state vector

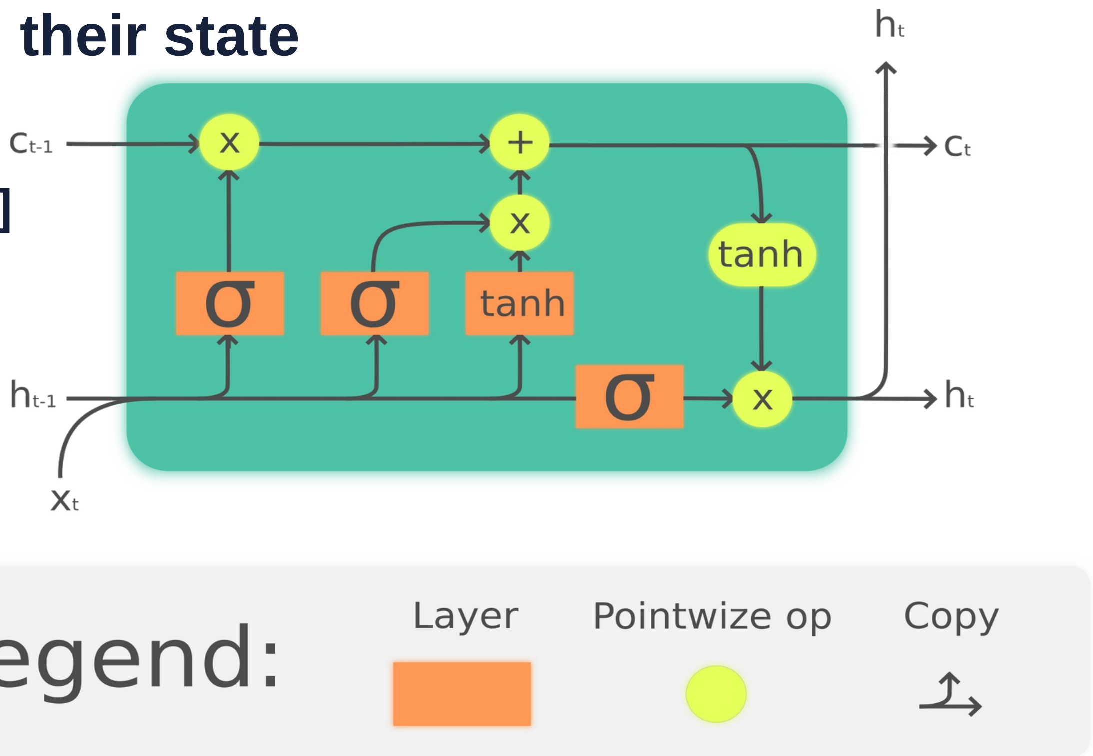
$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$

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

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

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

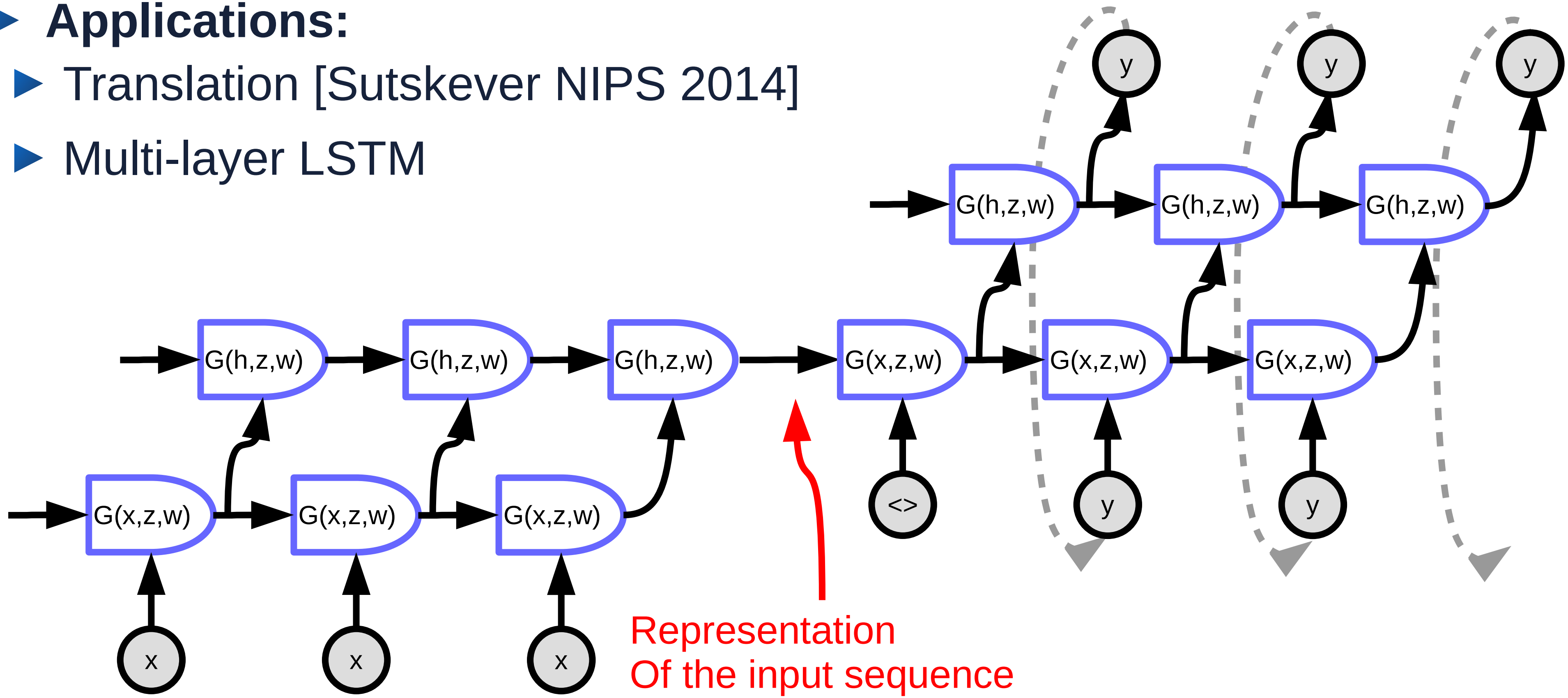
$$h_t = o_t \circ \sigma_h(c_t)$$



Guillaume Chevalier https://commons.wikimedia.org/w/index.php?curid=71836793

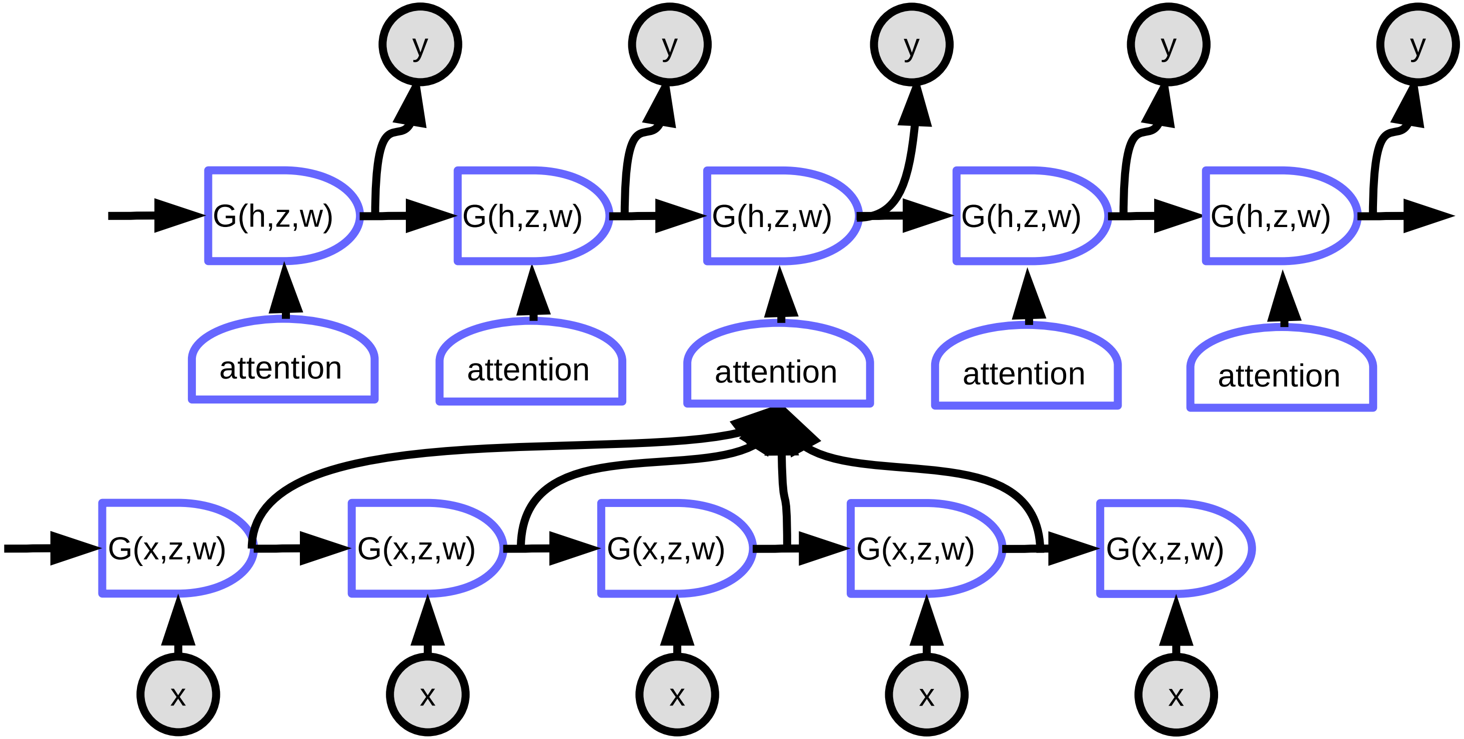
Sequence to Sequence

- **Sequence encoder → sequence decoder**
- **Applications:**
  - Translation [Sutskever NIPS 2014]
  - Multi-layer LSTM



Sequence to Sequence with Attention

- **Sequence encoder → sequence decoder with attention**
- **Applications:**
  - Translation [Bahdanau, Cho, Bengio ArXiv:1409.0473]



Memory Network

- **Short-term associative memory**

Transformer

- **Each group of unit is a memory network**