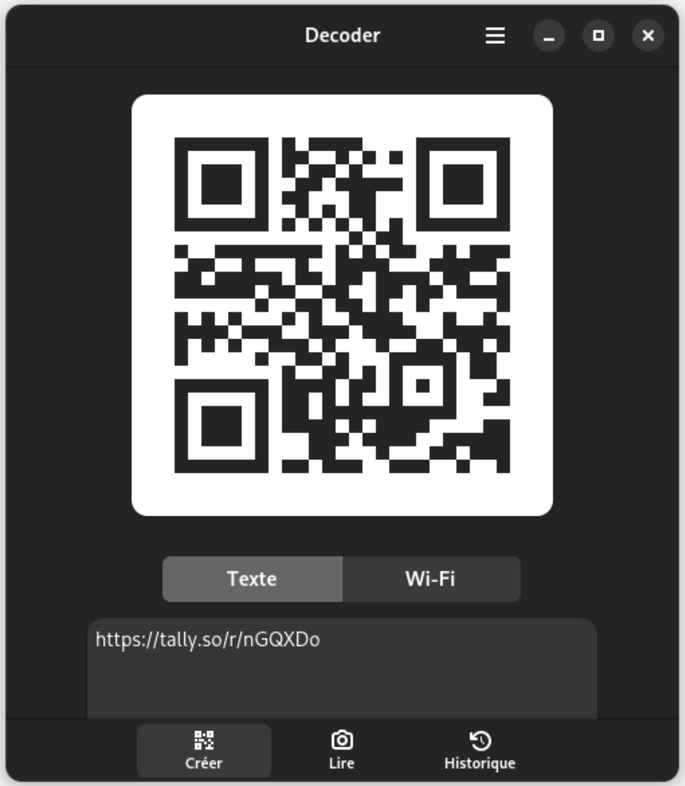
Recurrent Neural Networks

Antoine Bosselut

EPFL



🚀 Accelerate Your Career in AI Safety!

We're launching a **12-week program** designed to introduce you to diverse opportunities in **AI safety**. Here's what you can expect:
✅ **Exclusive Cohort** – Only **10 to 20 participants**, selected based on motivation and experience.

✅ **Personalized Mentorship** – Each participant will be matched with a dedicated mentor.

✅ **Engaging Workshops & Talks** – Every **two weeks on Monday evenings**, featuring experts in the field.

✅ **Research Access** – Gain facilitated entry into **HAIDI**, a leading AI safety research group in collaboration with the **Swiss AI Initiative**.

✅ **Certificate of Completion** – Earn official recognition from **Safe AI Lausanne**.

Applications will be reviewed on a rolling-basis. We will start latest on the 3rd of March.

If you have any questions, feel free to ask here: https://go.epfl.ch/aisafety-telegram

Let's start with your personal details

Gender

First Name *

Last Name *

Email address *

Phone number *

What is your degree? *

the sustainable innovation challenge

a student-led initiative organizing a sustainability-focused **challenge** and **conference**

13th & 14th of March - Rolex Learning Center @EPFL

epfl-sic.ch

Contact: sic@epfl.ch
Instagram: @epflsic
LinkedIn: EPFL Sustainable Innovation Challenge

EPFL

CARBON REMOVAL BOOSTER

ETH STUDENT PROJECT HOUSE

Transversal Skills and Career Center

EPFL Sustainability

EPFL RoboCup Team

EPFL
CREATE Lab

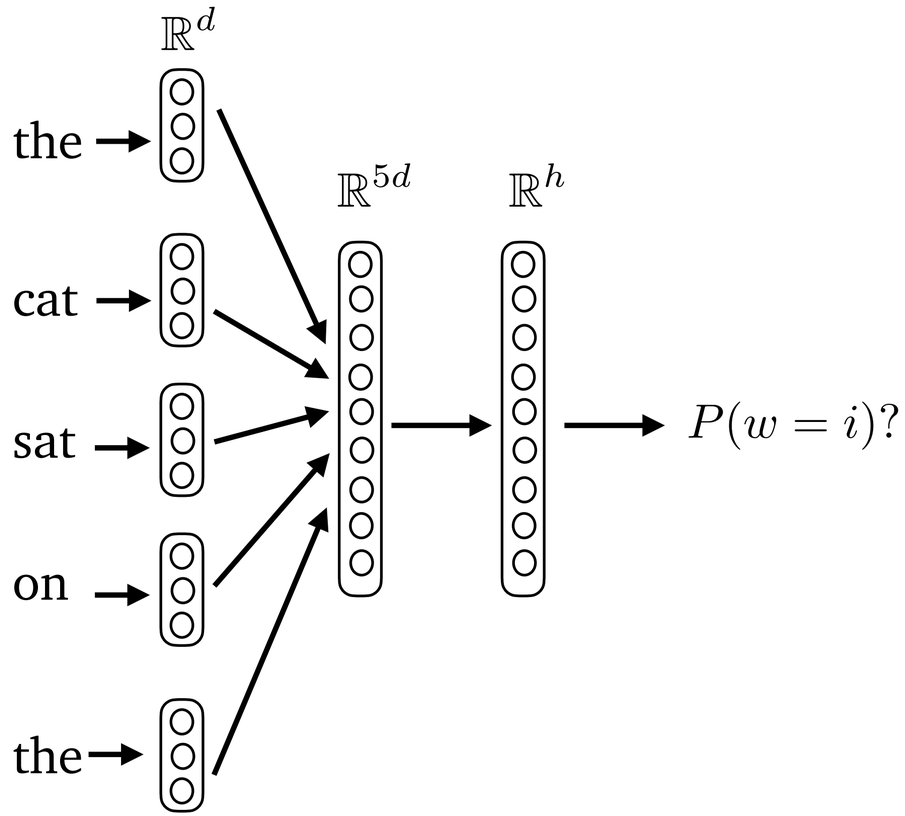
RoboCup

Section Outline

- **Fixing the context bottleneck:** recurrent neural networks
- **Training recurrent neural networks:** backpropagation through time
- **Training Challenges:** Vanishing Gradients
- **Mitigations:** LSTMs, GRUs

Fixed-context Neural Language Models

- P(mat | the cat sat on the) = ?



- Input layer (n = 5):

$$\mathbf{x} = [\mathbf{e}_{\text{the}}; \mathbf{e}_{\text{cat}}; \mathbf{e}_{\text{sat}}; \mathbf{e}_{\text{on}}; \mathbf{e}_{\text{the}}] \in \mathbb{R}^{dn}$$

- Hidden layer

$$\mathbf{h} = \tanh(\mathbf{W}\mathbf{x} + \mathbf{b}) \in \mathbb{R}^h$$

- Output layer (softmax)

$$\mathbf{z} = \mathbf{U}\mathbf{h} \in \mathbb{R}^{|V|}$$

$$P(w = i \mid \text{the cat sat on the})$$

$$= \text{softmax}_i(\mathbf{z}) = \frac{e^{z_i}}{\sum_k e^{z_k}}$$

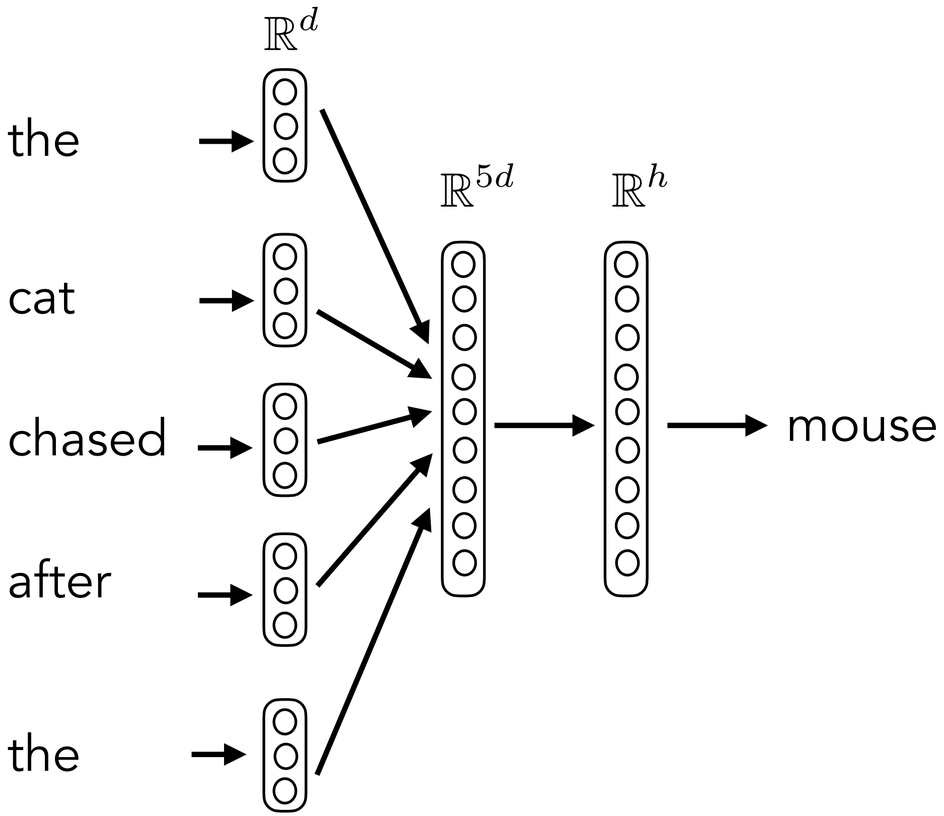
(Bengio et al., 2003): A Neural Probabilistic Language Model

Advantages vs. Disadvantages

- No more sparsity problem
  - All sequences can be estimated with non-zero probability ! **Why?**
- Model size is much smaller!
  - Depends on number of weights in model, not number of sequences!

- Fixed windows are still too small to encode **long-range dependencies**

Fixed-context Neural Language Models



P(mouse | the cat chased the) ✓

P(mouse | the starving cat chased the) ✓

P(mouse | starving cat chased after the) ✓

P(mouse | cat fanatically chased after the) ✓

P(mouse | fanatically chased after the elusive) ✗

(Bengio et al., 2003): A Neural Probabilistic Language Model

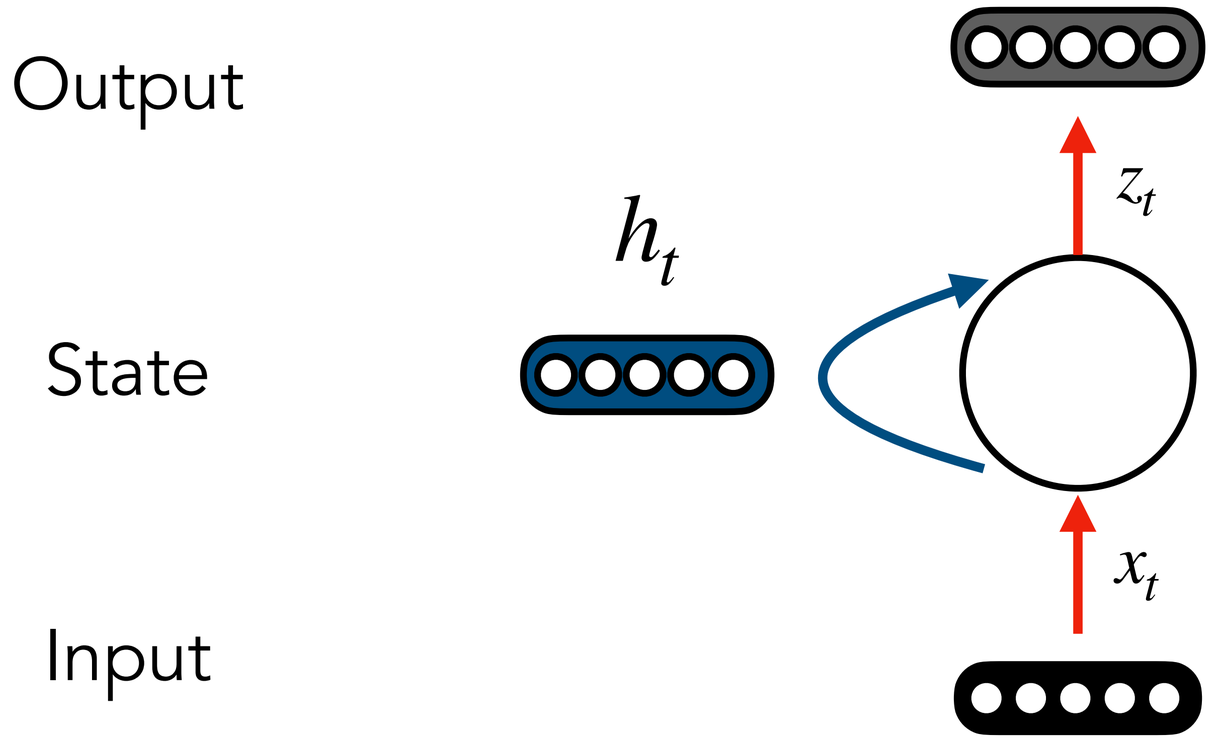
Advantages vs. Disadvantages

- No more sparsity problem
  - All sequences can be estimated with non-zero probability ! **Why?**
- Model size is much smaller!
  - Depends on number of weights in model, not number of sequences!

- Fixed windows are still too small to encode **long-range dependencies**
- Weights in **W** aren't shared across embeddings in the window (computationally inefficient!)
- Enlarging the window size makes the weight matrix **W** larger (more computationally expensive!)

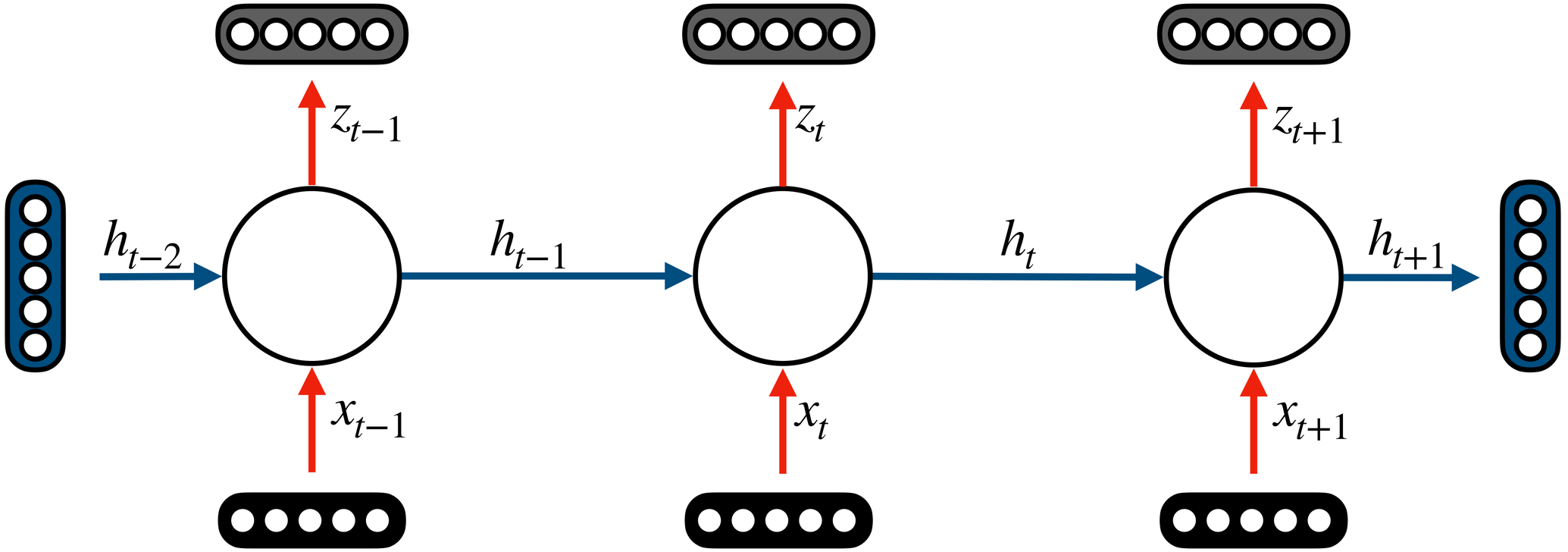
Recurrent Neural Networks

- **Solution:** Recurrent neural networks — NNs with feedback loops



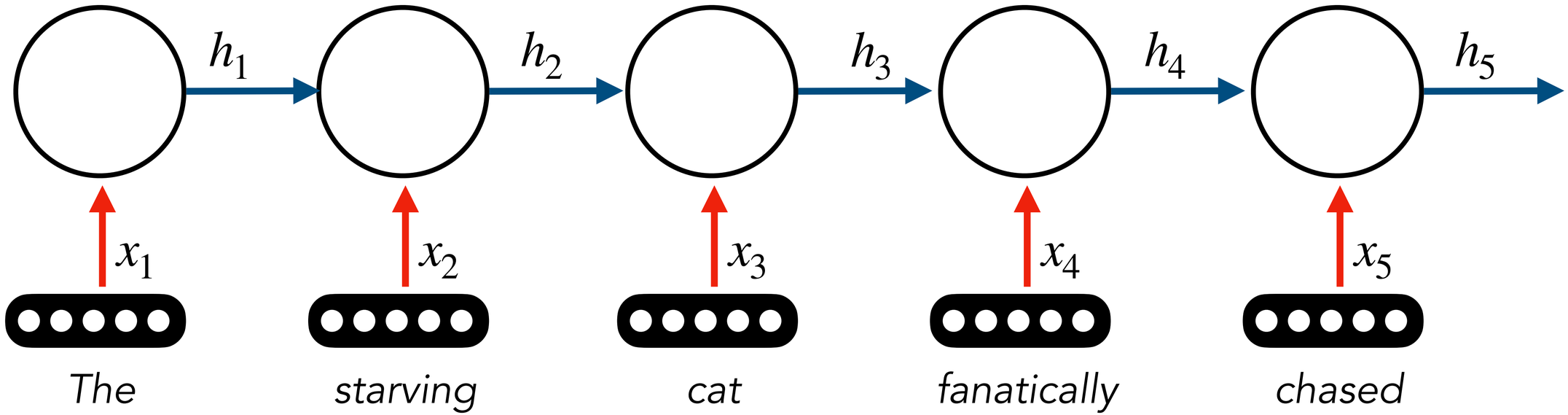
Recurrent Neural Networks

Unrolling the RNN across all time steps gives full computation graph

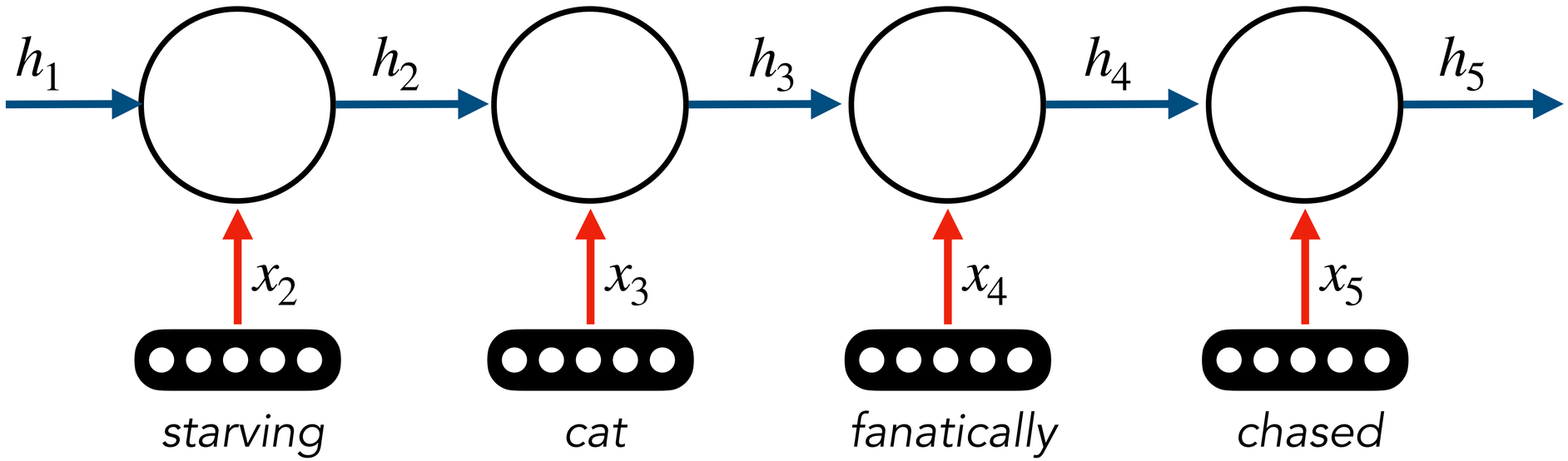


Allows for learning from entire sequence history, regardless of length

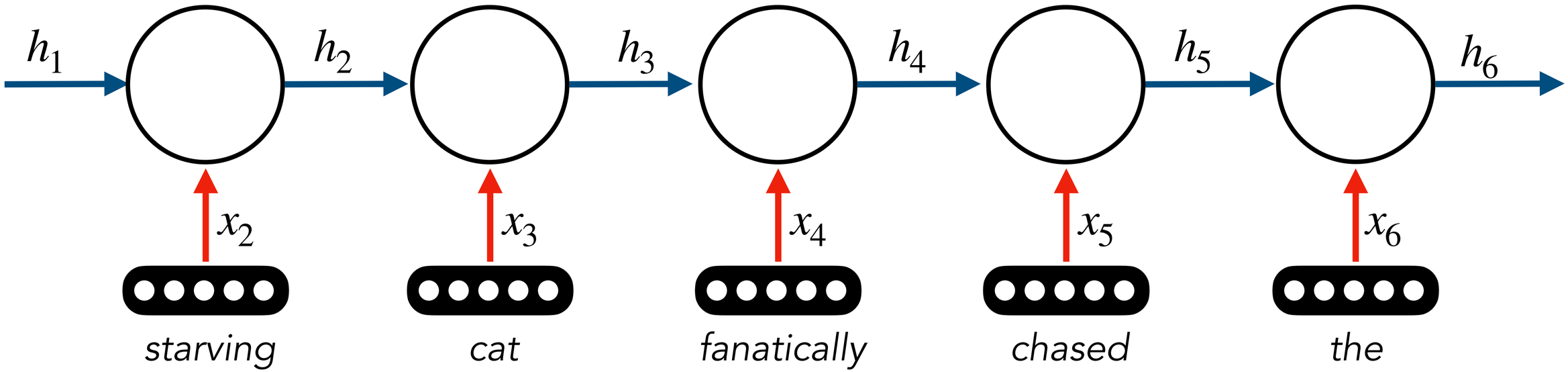
Unrolling the RNN



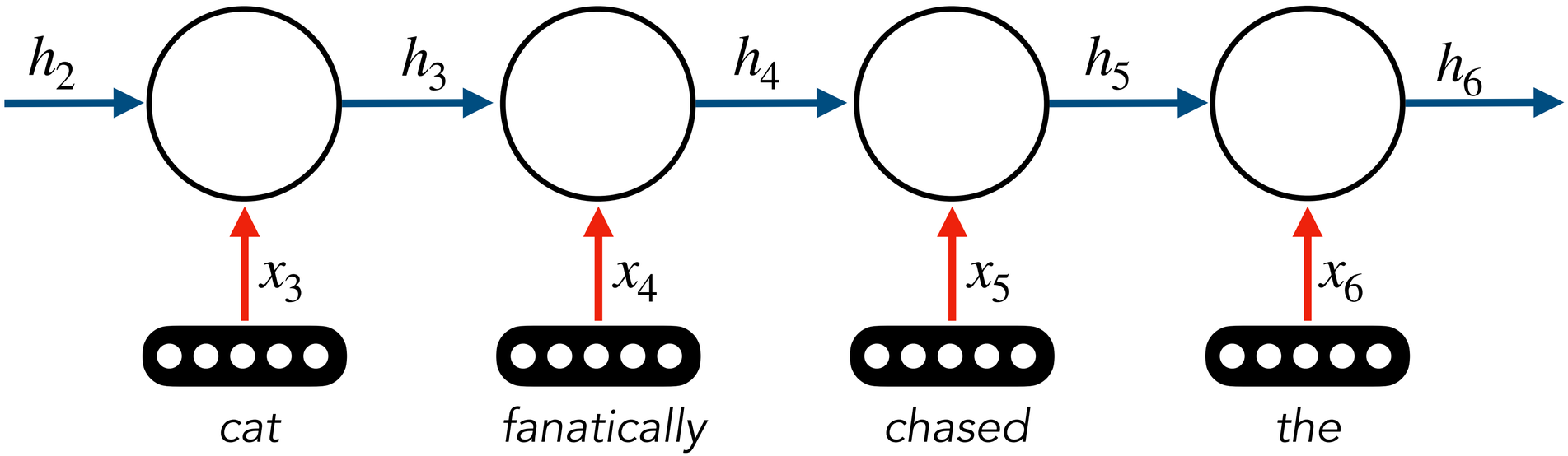
Unrolling the RNN



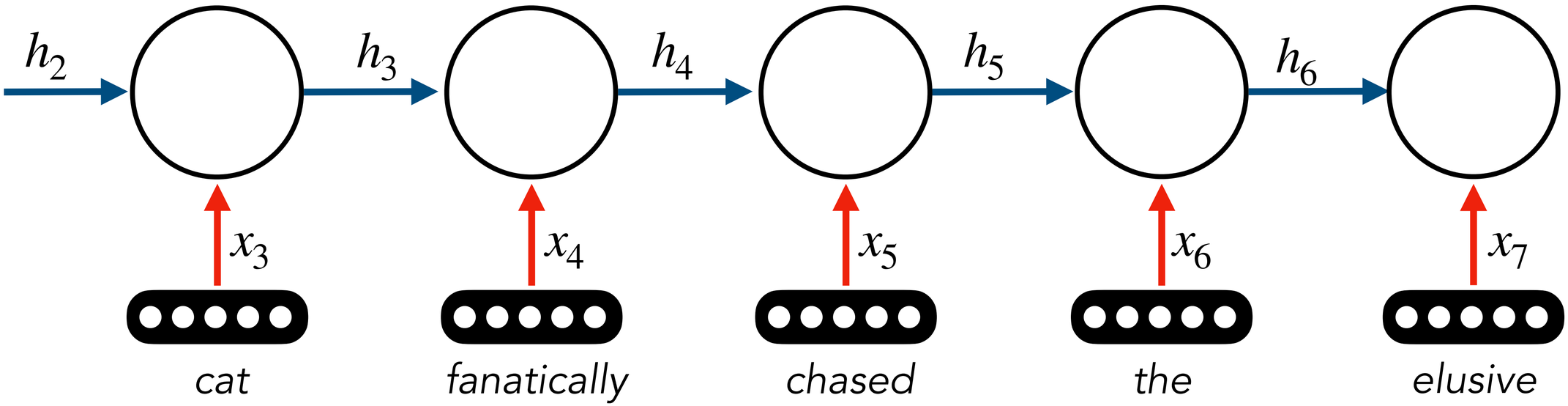
Unrolling the RNN



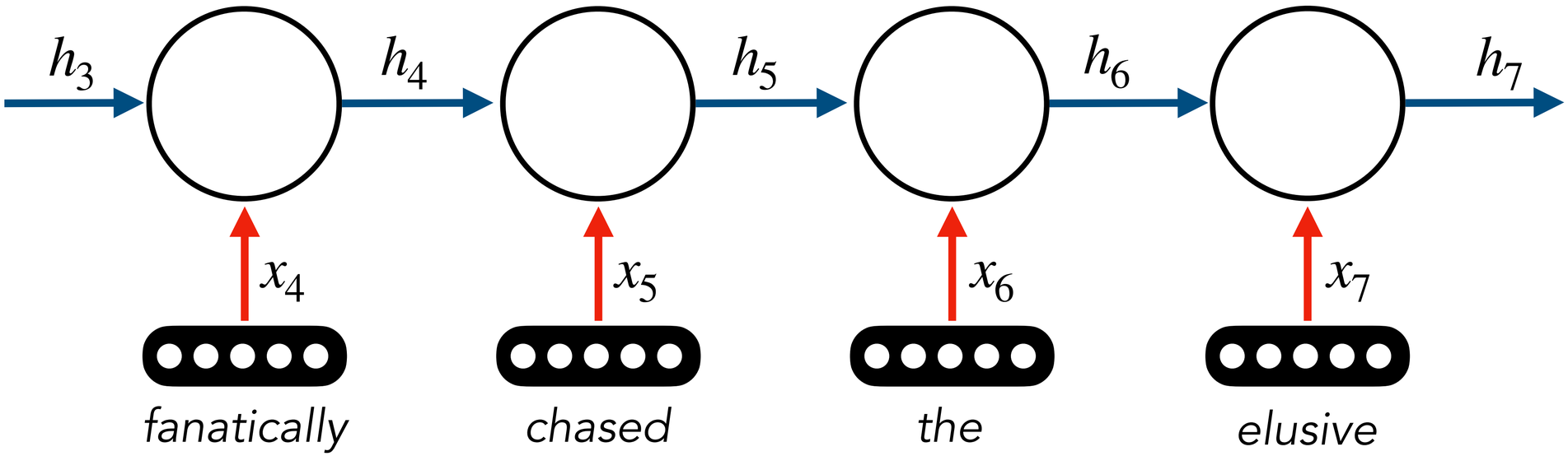
Unrolling the RNN



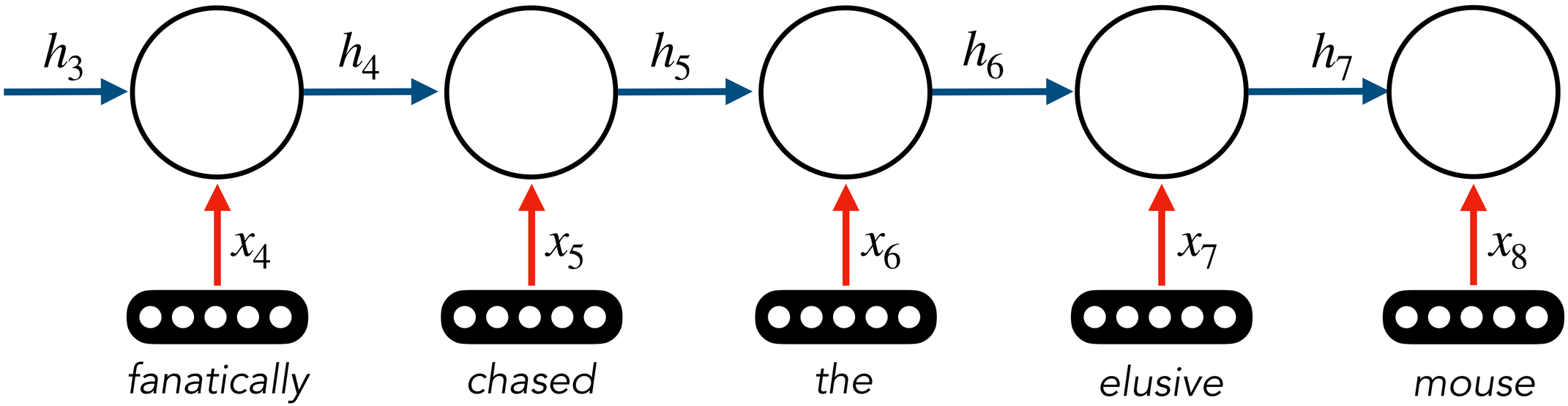
Unrolling the RNN



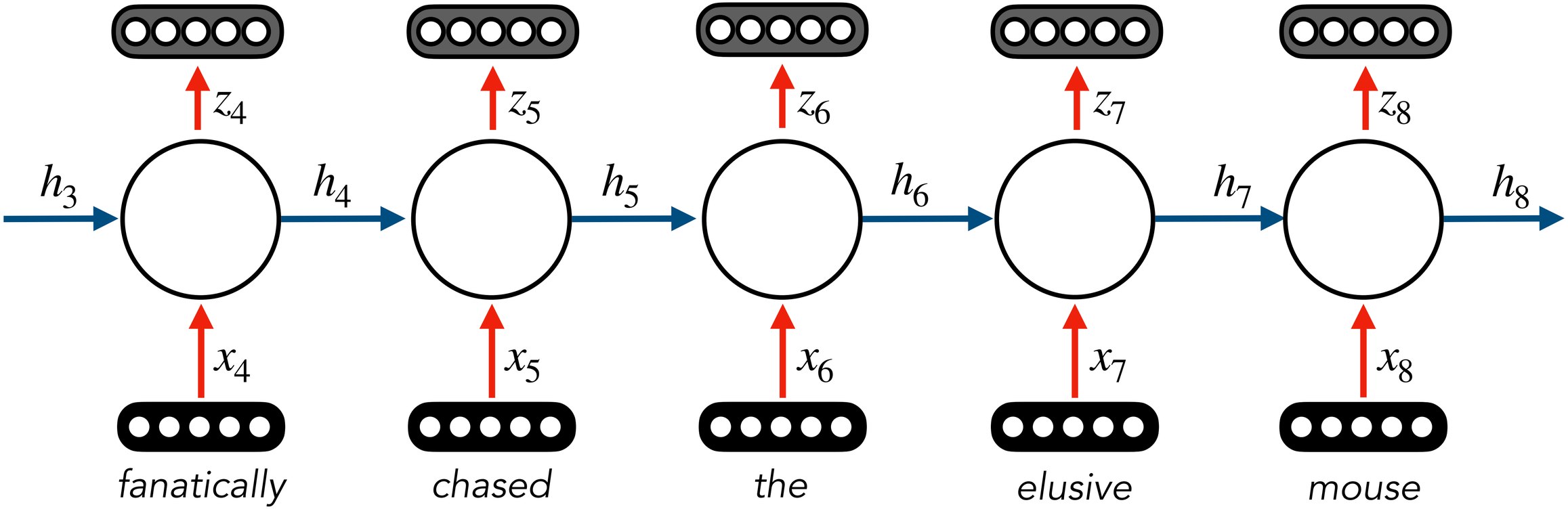
Unrolling the RNN



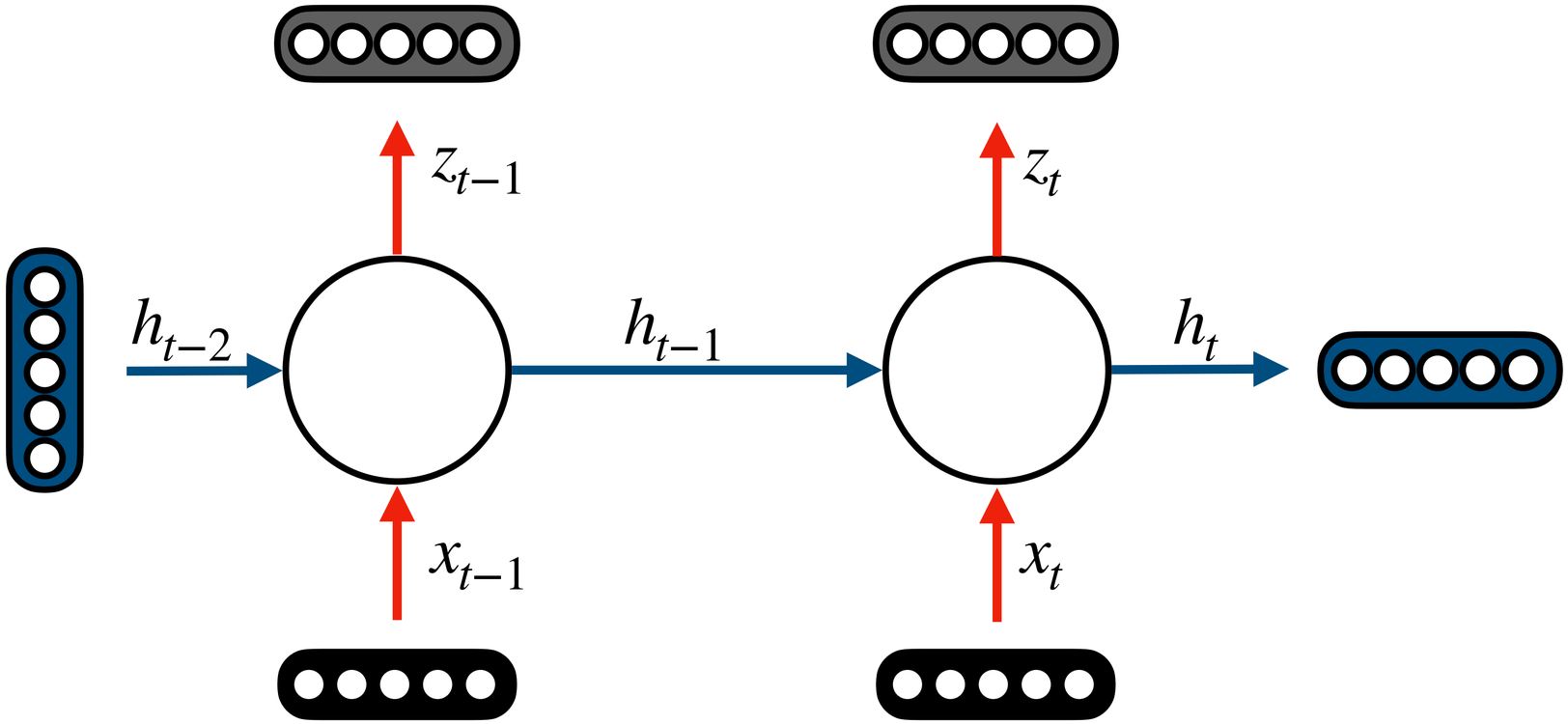
Unrolling the RNN



Unrolling the RNN

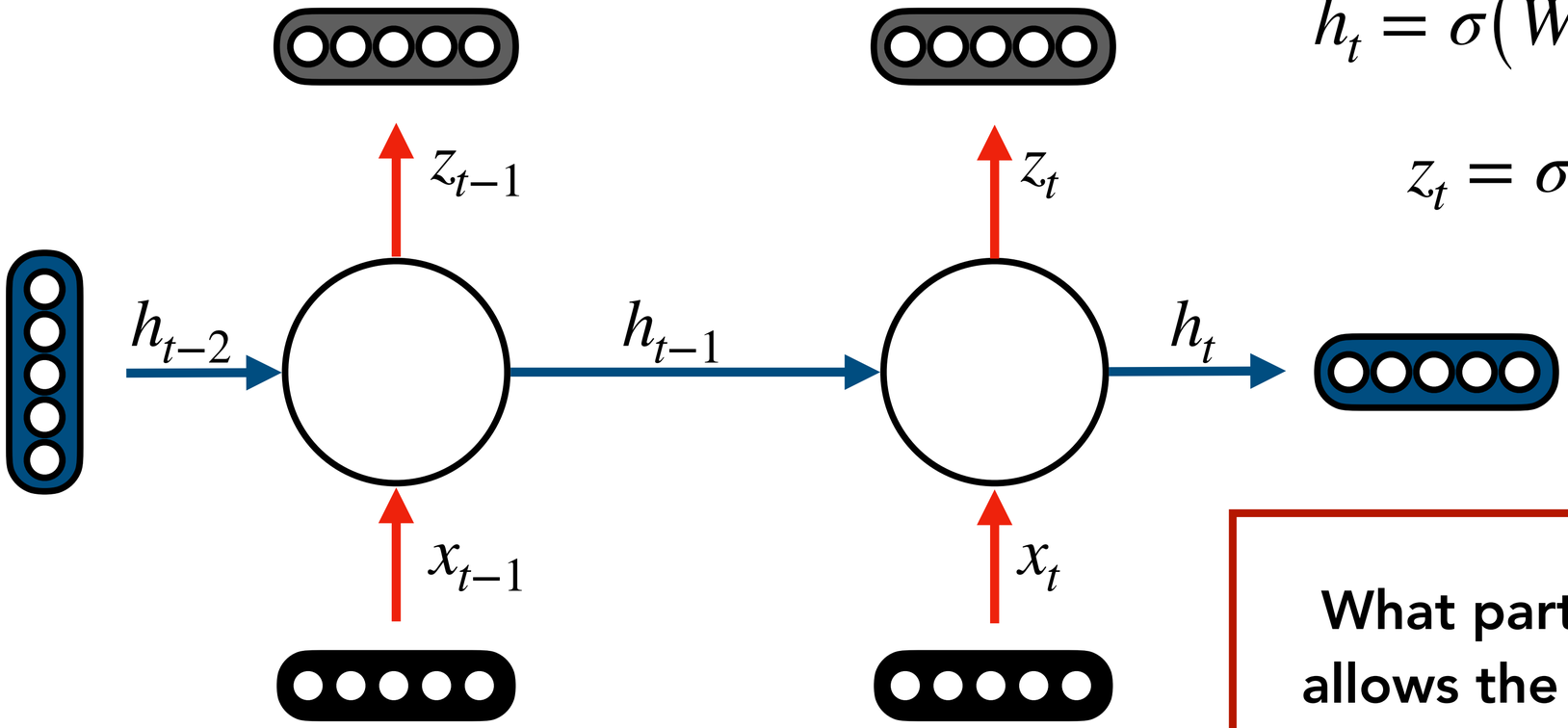


Classical RNN: Elman Network



(Elman, 1990)

Classical RNN: Elman Network



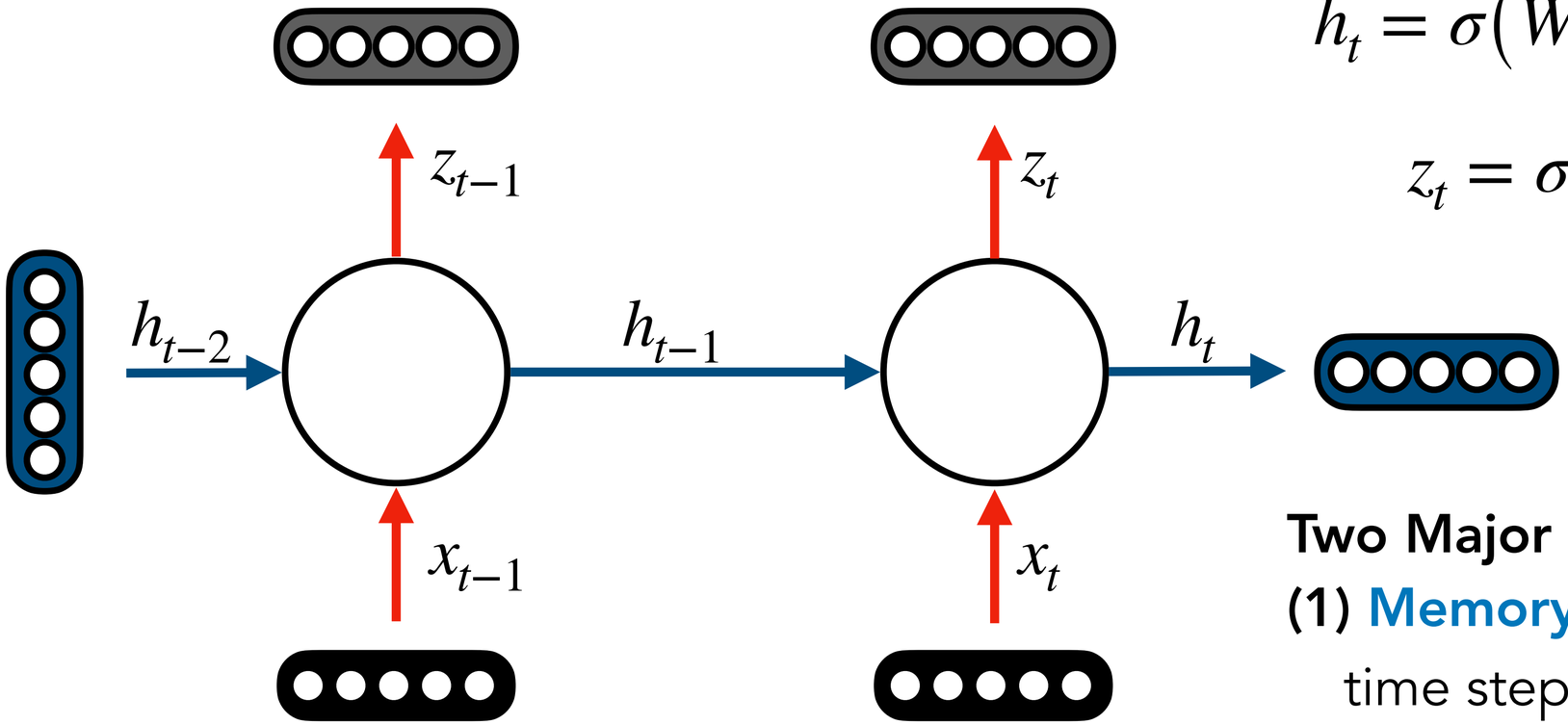
$$h_t = \sigma\left(W_{hx}x_t + W_{hh}h_{t-1} + b_h\right)$$

$$z_t = \sigma\left(W_{zh}h_t + b_z\right)$$

What part of this formulation allows the model to maintain a state of previously input tokens ?

(Elman, 1990)

Classical RNN: Elman Network



$$h_t = \sigma\left(W_{hx}x_t + W_{hh}h_{t-1} + b_h\right)$$

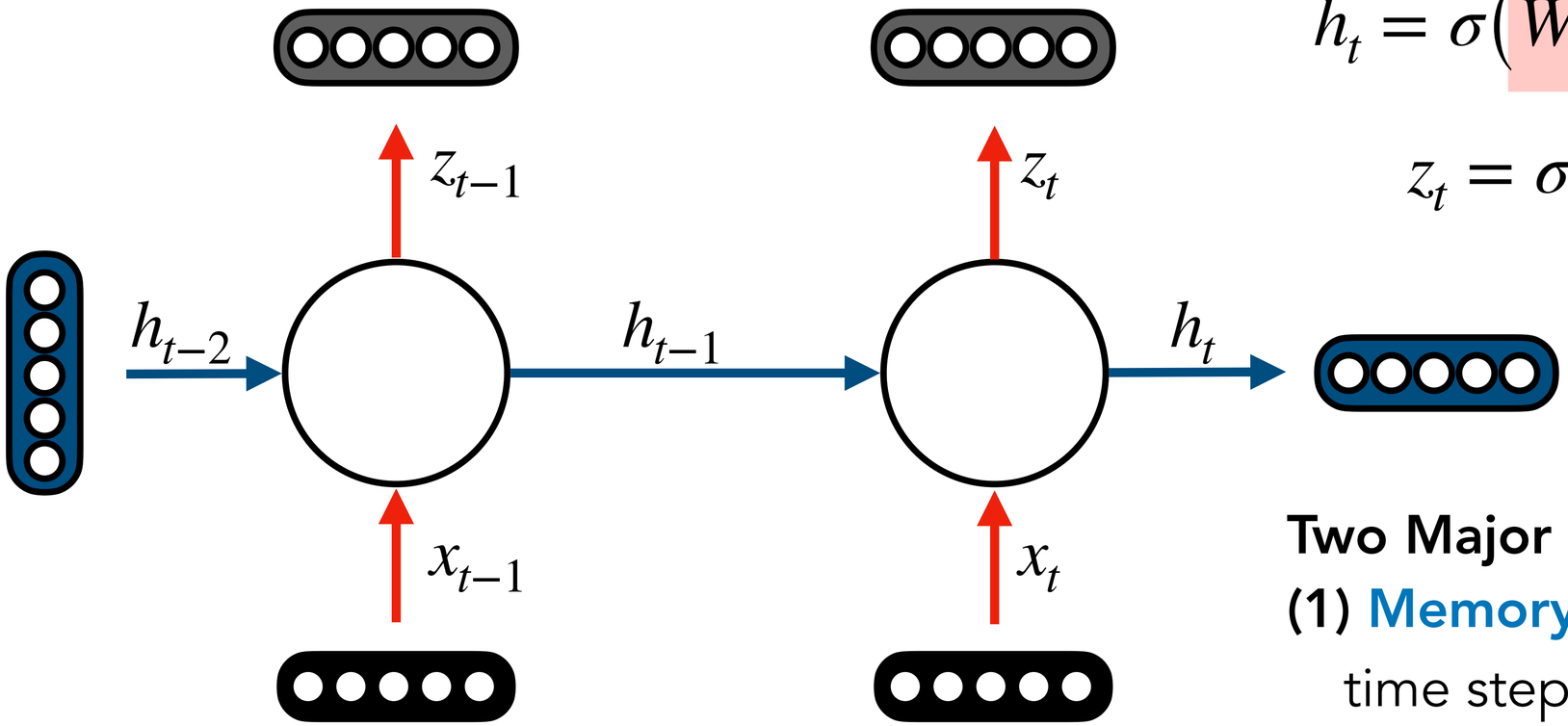
$$z_t = \sigma\left(W_{zh}h_t + b_z\right)$$

Two Major Components:

(1) **Memory** is maintained across time steps by the state h.

(Elman, 1990)

Classical RNN: Elman Network



$$h_t = \sigma\left(W_{hx}x_t + W_{hh}h_{t-1} + b_h\right)$$

$$z_t = \sigma\left(W_{zh}h_t + b_z\right)$$

Two Major Components:

(1) **Memory** is maintained across time steps by the state h.

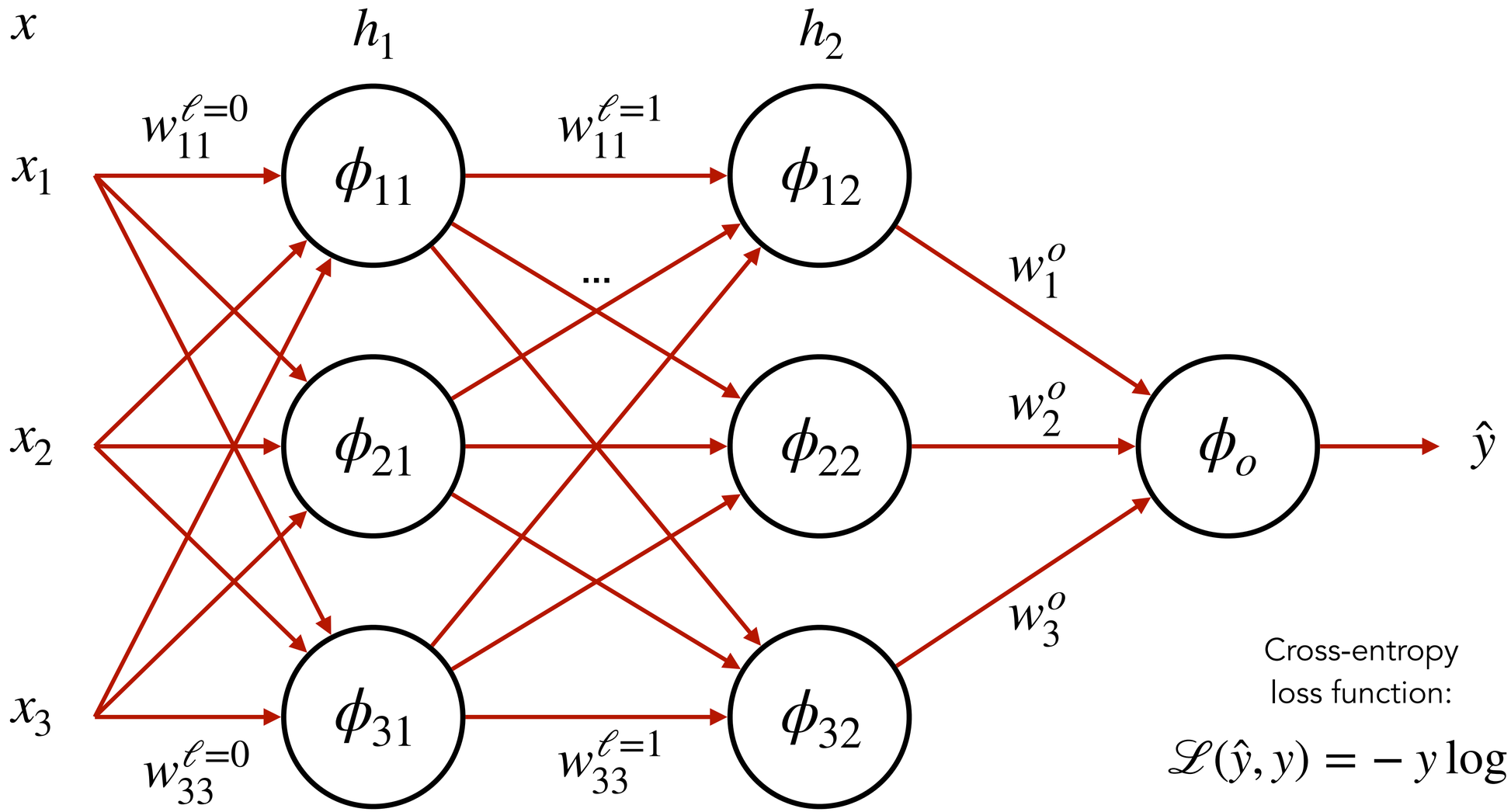
(2) **Parameter matrices** are shared across all time steps.

(Elman, 1990)

Objective

Learn parameter matrices W_{hh}, W_{hx}, W_{zh} in the Elman network such that the RNN can represent textual sequences

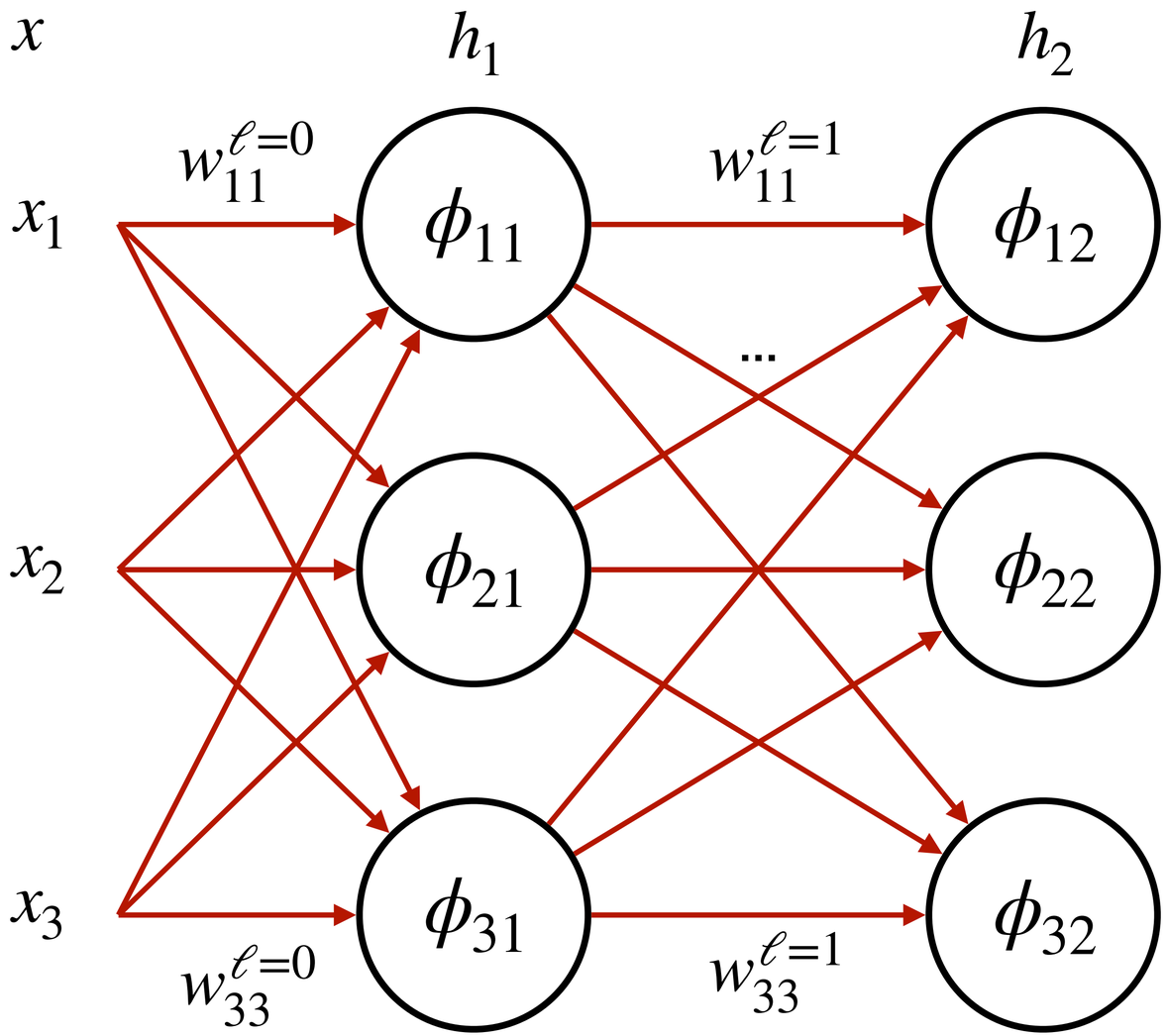
Backpropagation Review: FFNs



Cross-entropy loss function:

$$\mathcal{L}(\hat{y}, y) = -\, y \log P(\hat{y})$$

Backpropagation Review: FFNs



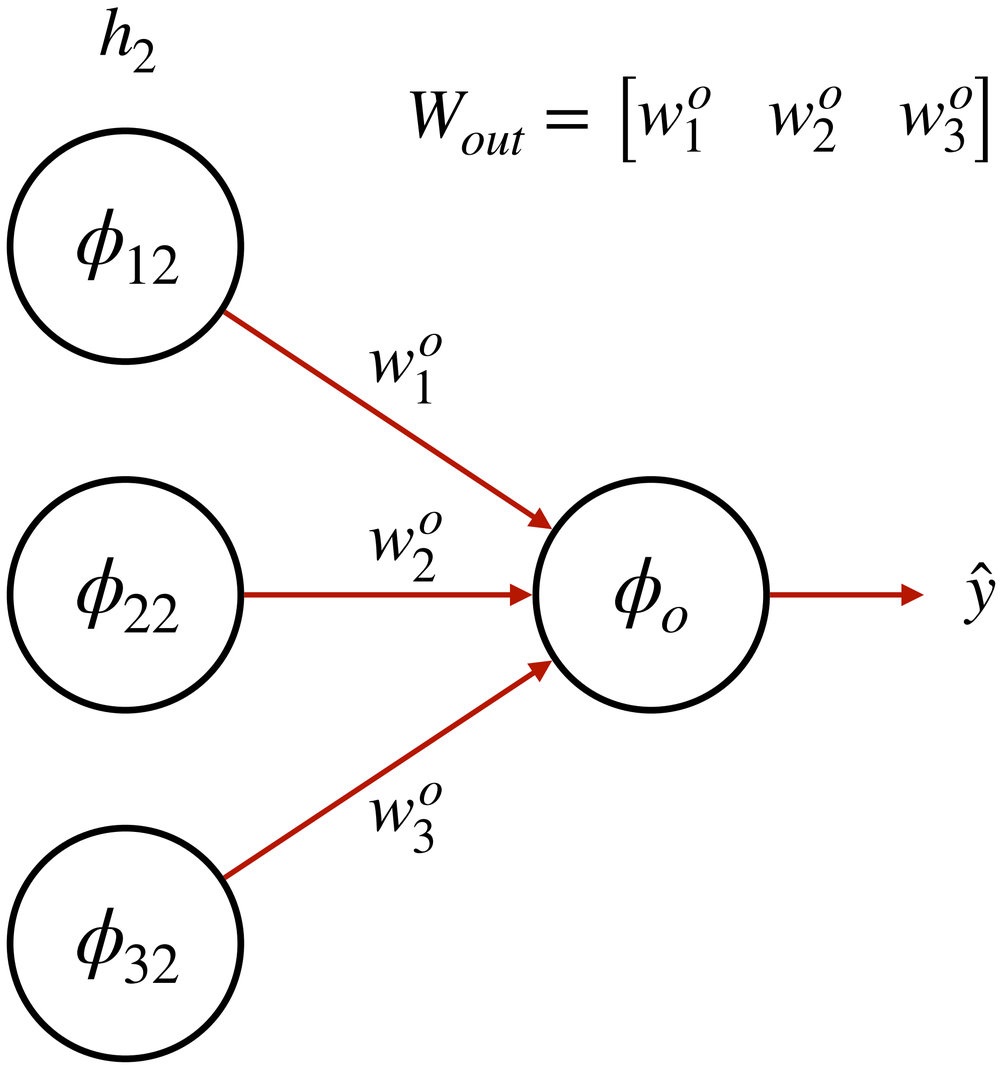
$$W_0 = \begin{bmatrix} w_{11}^{\ell=0} & w_{21}^{\ell=0} & w_{31}^{\ell=0} \\ w_{12}^{\ell=0} & w_{22}^{\ell=0} & w_{32}^{\ell=0} \\ w_{13}^{\ell=0} & w_{23}^{\ell=0} & w_{33}^{\ell=0} \end{bmatrix}$$

$$W_1 = \begin{bmatrix} w_{11}^{\ell=1} & w_{21}^{\ell=1} & w_{31}^{\ell=1} \\ w_{12}^{\ell=1} & w_{22}^{\ell=1} & w_{32}^{\ell=1} \\ w_{13}^{\ell=1} & w_{23}^{\ell=1} & w_{33}^{\ell=1} \end{bmatrix}$$

$$h_1 = \phi_1(W_0 x)$$

$$h_2 = \phi_2(W_1 h_1)$$

Backpropagation Review: FFNs



$$W_{out} = \begin{bmatrix} w_1^o & w_2^o & w_3^o \end{bmatrix}$$

$$\mathscr{L}(\hat{y}, y) = -\, y \log P(\hat{y})$$

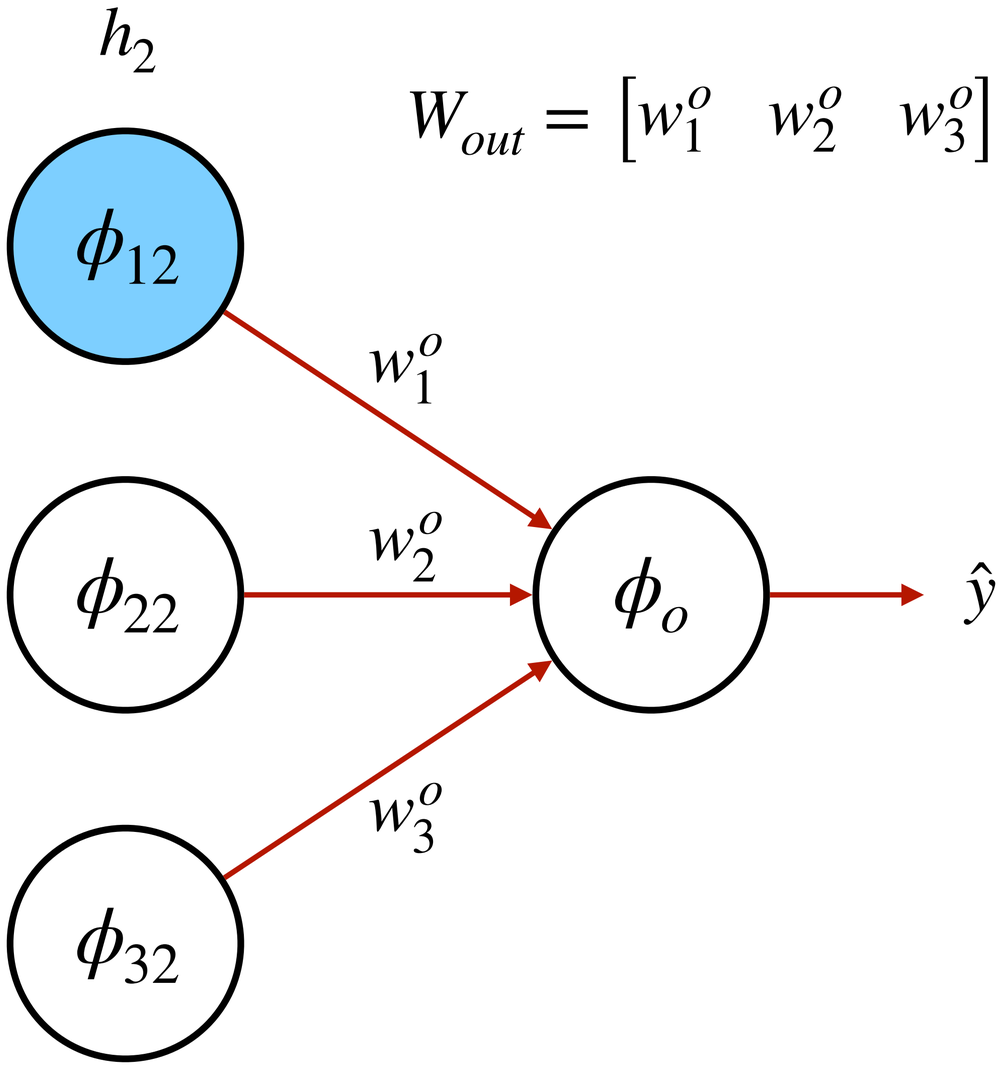
$$P(\hat{y}) = \phi_o(u)$$

$$u = W_{out} h_2 = w_1^o \times \phi_{12}(\,.\,) + w_2^o \times \phi_{22}(\,.\,) + w_3^o \times \phi_{32}(\,.\,)$$

$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{12}(\,.\,)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(\,.\,)}$$

$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

Backpropagation Review: FFNs



$$W_{out} = \begin{bmatrix} w_1^o & w_2^o & w_3^o \end{bmatrix}$$

$$\mathscr{L}(\hat{y}, y) = -\, y \log P(\hat{y})$$

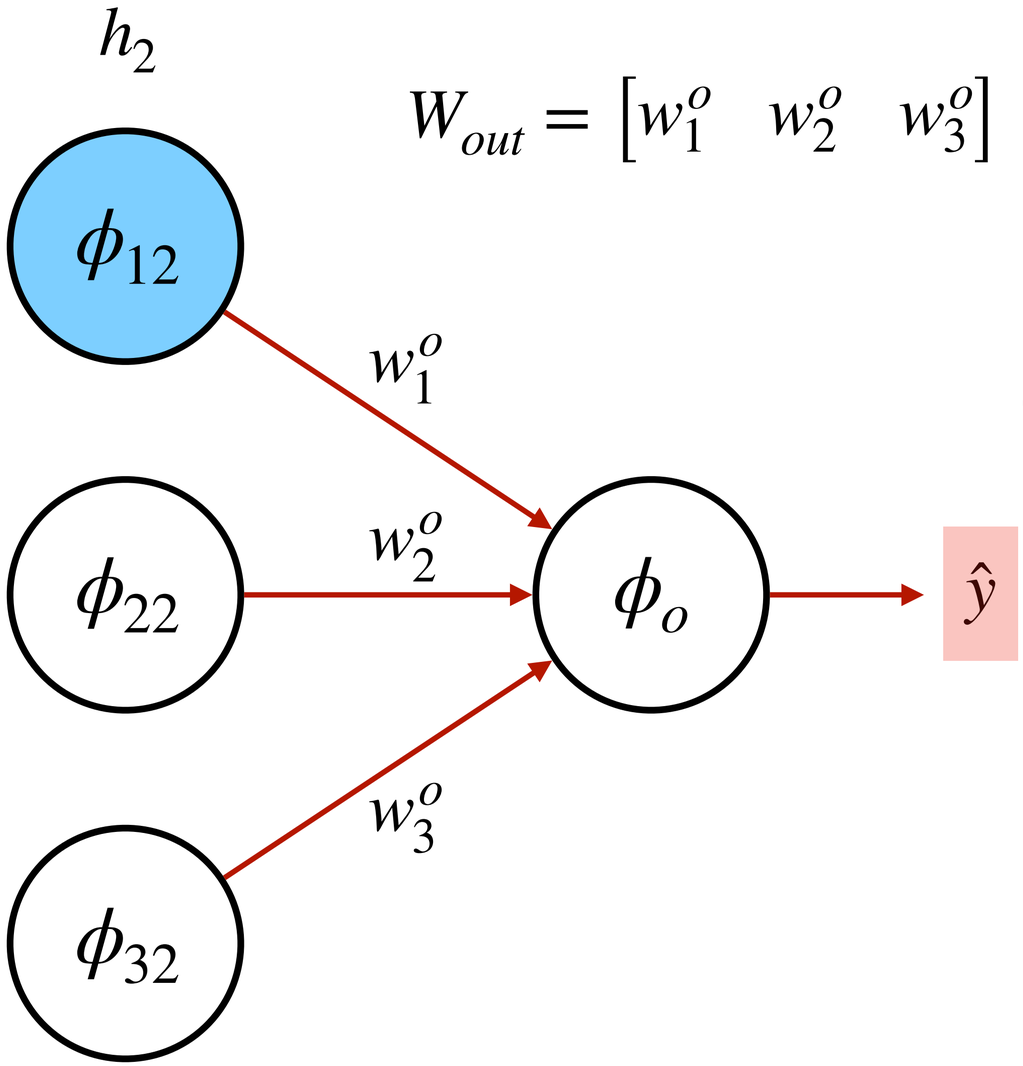
$$P(\hat{y}) = \phi_o(u)$$

$$u = W_{out} h_2 = w_1^o \times \phi_{12}(\,.\,) + w_2^o \times \phi_{22}(\,.\,) + w_3^o \times \phi_{32}(\,.\,)$$

$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{12}(\,.\,)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(\,.\,)}$$

$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

Backpropagation Review: FFNs



$$W_{out} = \begin{bmatrix} w_1^o & w_2^o & w_3^o \end{bmatrix}$$

$$\mathscr{L}(\hat{y}, y) = -\, y \log P(\hat{y})$$

$$P(\hat{y}) = \phi_o(u)$$

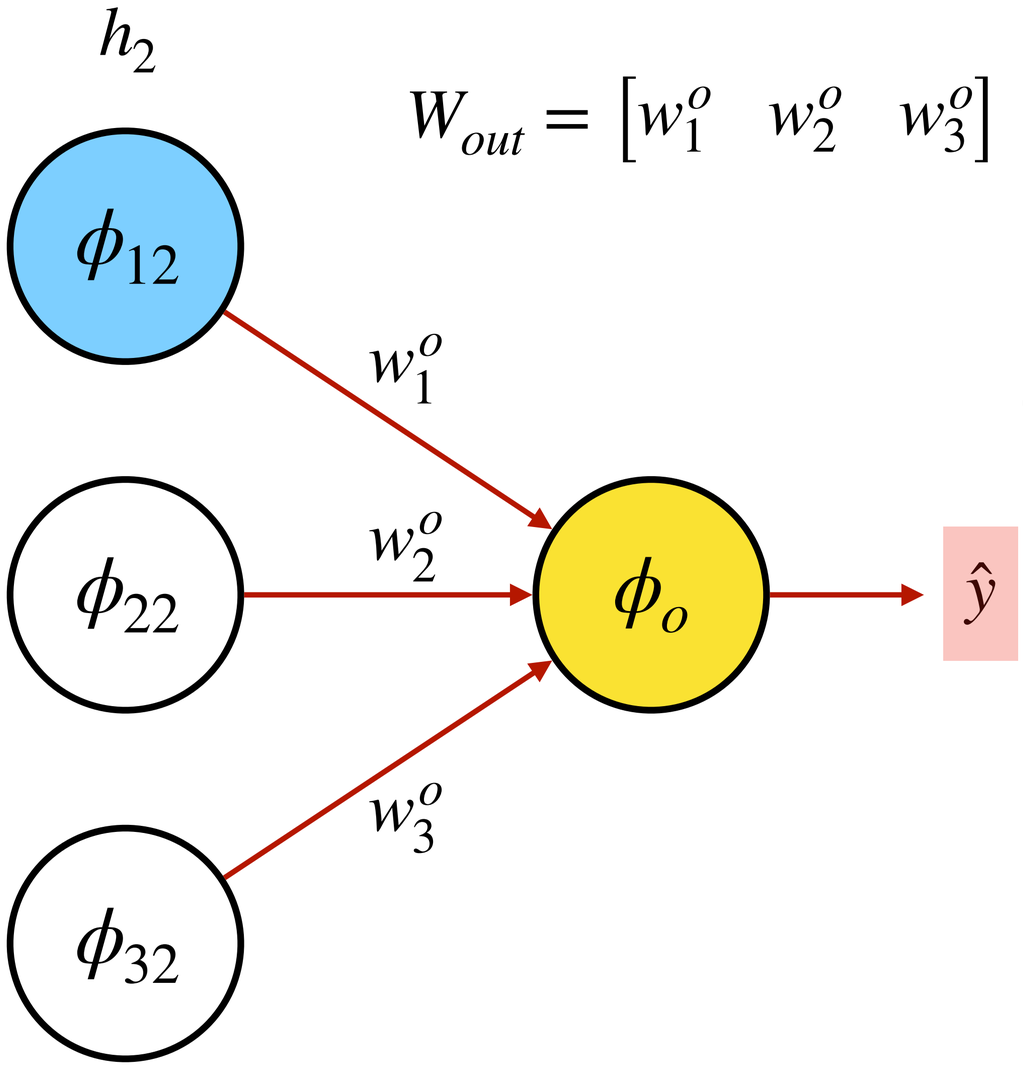
$$u = W_{out} h_2 = w_1^o \times \phi_{12}(\,.\,) + w_2^o \times \phi_{22}(\,.\,) + w_3^o \times \phi_{32}(\,.\,)$$

$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{12}(\,.\,)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(\,.\,)}$$

$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

Depends on label y

Backpropagation Review: FFNs



$$W_{out} = \begin{bmatrix} w_1^o & w_2^o & w_3^o \end{bmatrix}$$

$$\mathcal{L}(\hat{y}, y) = -\, y \log P(\hat{y})$$

$$P(\hat{y}) = \phi_o(u)$$

$$u = W_{out} h_2 = w_1^o \times \phi_{12}(\,.\,) + w_2^o \times \phi_{22}(\,.\,) + w_3^o \times \phi_{32}(\,.\,)$$

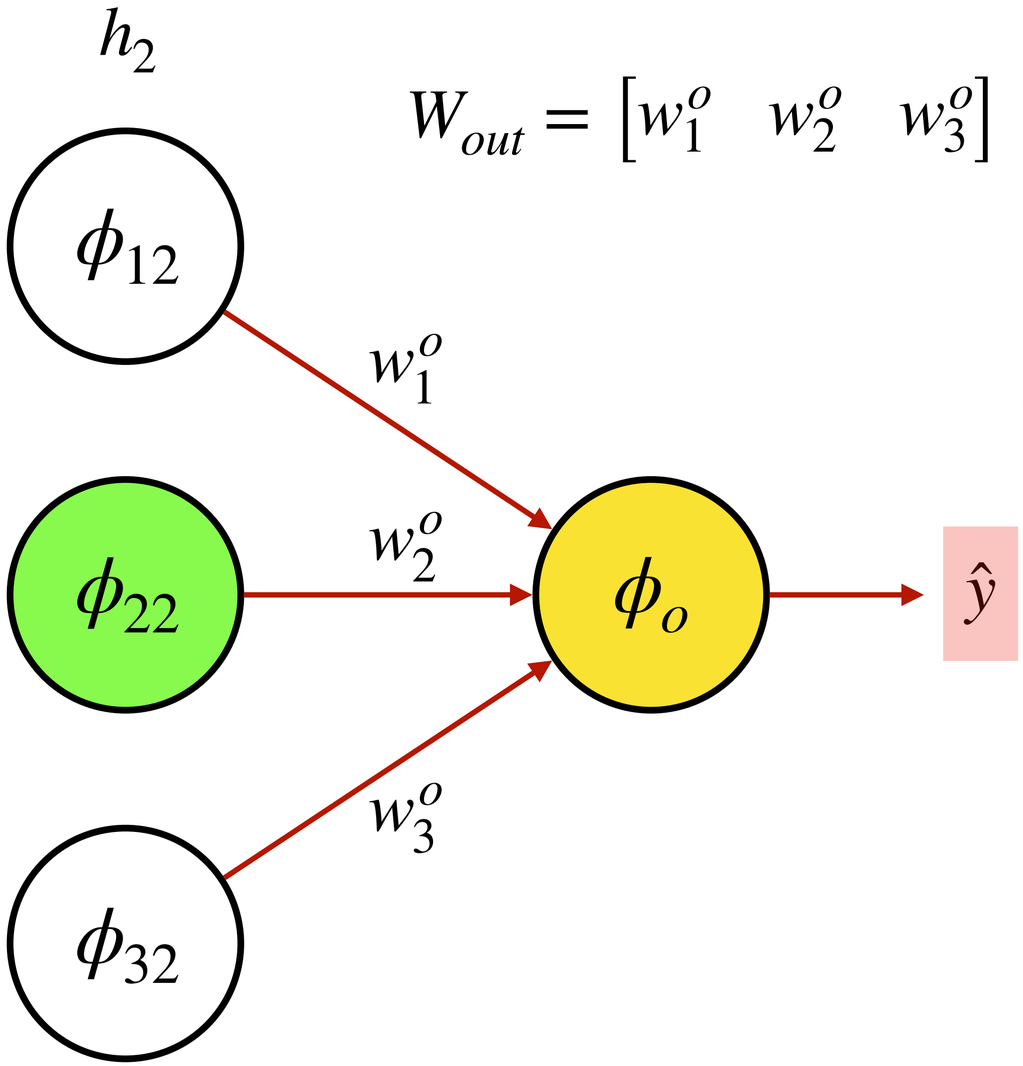
$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{12}(\,.\,)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(\,.\,)}$$

$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

Depends on label y

Depends on ϕ_o

Backpropagation Review: FFNs



$$W_{out} = \begin{bmatrix} w_1^o & w_2^o & w_3^o \end{bmatrix}$$

$$\mathscr{L}(\hat{y}, y) = -\, y \log P(\hat{y})$$

$$P(\hat{y}) = \phi_o(u)$$

$$u = W_{out} h_2 = w_1^o \times \phi_{12}(\,.\,) + w_2^o \times \phi_{22}(\,.\,) + w_3^o \times \phi_{32}(\,.\,)$$

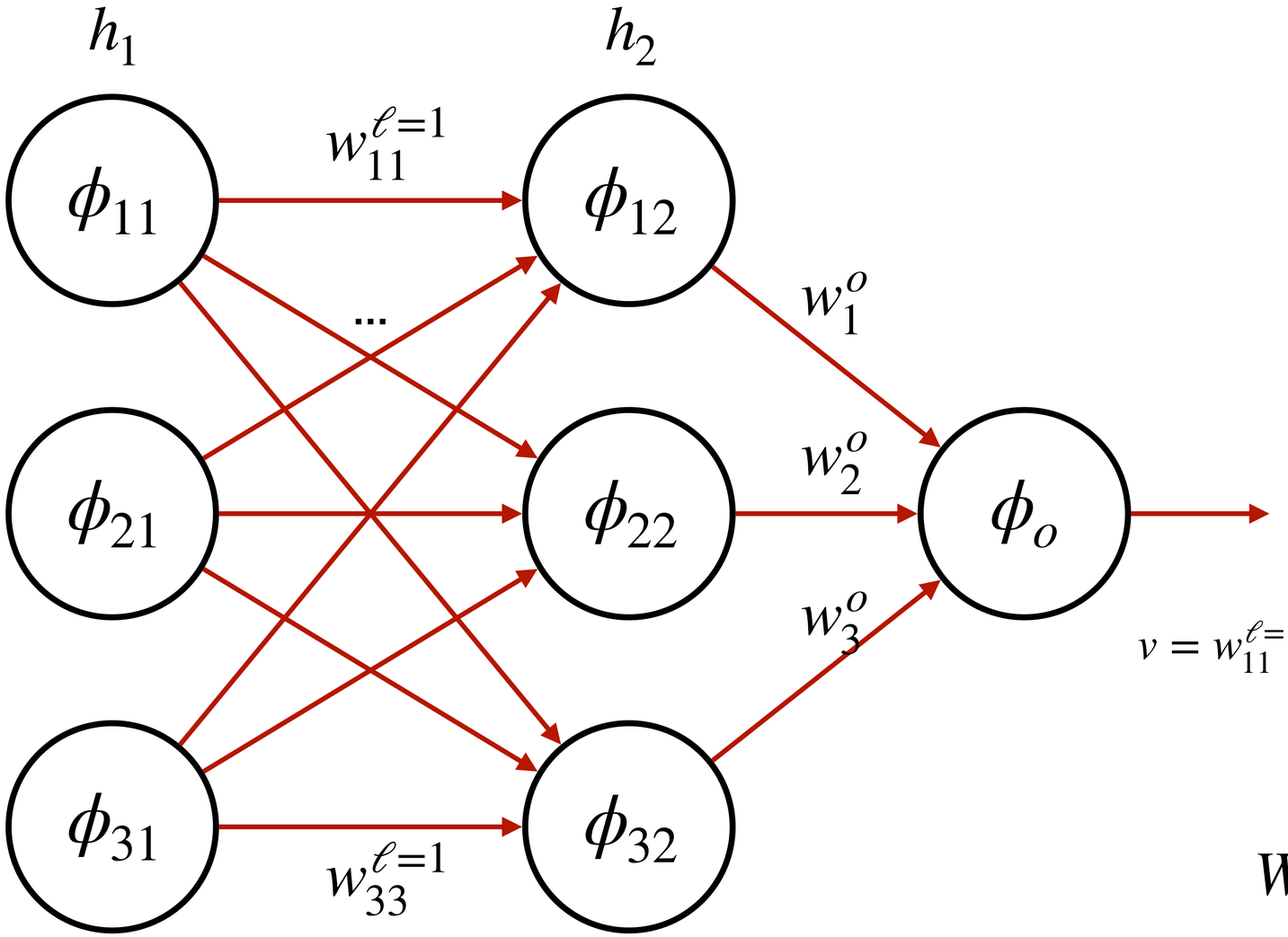
$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{22}(\,.\,)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{22}(\,.\,)}$$

$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_2^0$$

Depends on label y

Depends on ϕ_o

Backpropagation Review: FFNs



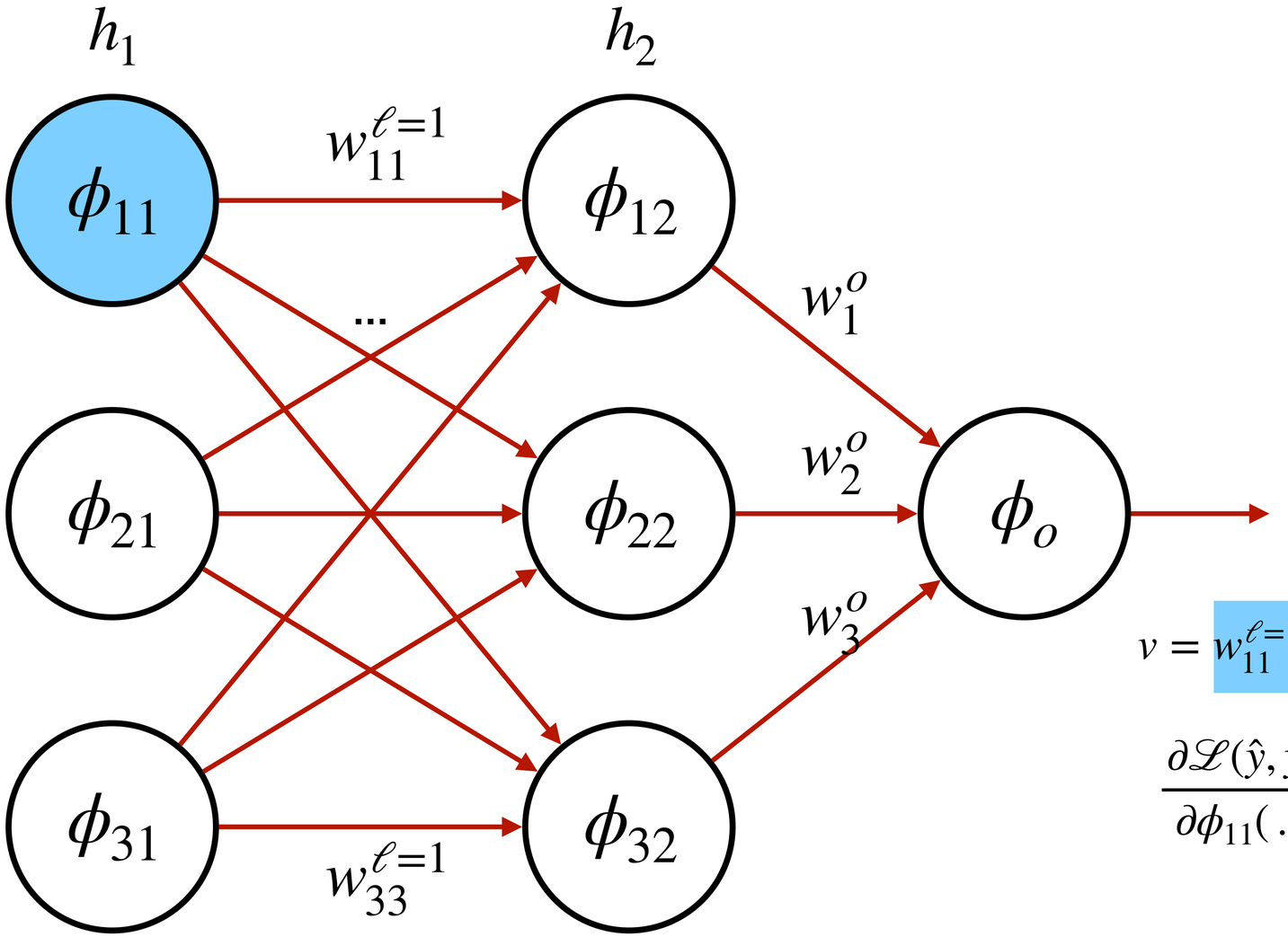
$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{12}(.)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(.)}$$

$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

$$v = w_{11}^{\ell=1} \times \phi_{11}(.) + w_{21}^{\ell=1} \times \phi_{21}(.) + w_{31}^{\ell=1} \times \phi_{31}(.)$$

$$W_1 = \begin{bmatrix} w_{11}^{\ell=1} & w_{21}^{\ell=1} & w_{31}^{\ell=1} \\ w_{12}^{\ell=1} & w_{22}^{\ell=1} & w_{32}^{\ell=1} \\ w_{13}^{\ell=1} & w_{23}^{\ell=1} & w_{33}^{\ell=1} \end{bmatrix}$$

Backpropagation Review: FFNs



$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{12}(.)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(.)}$$

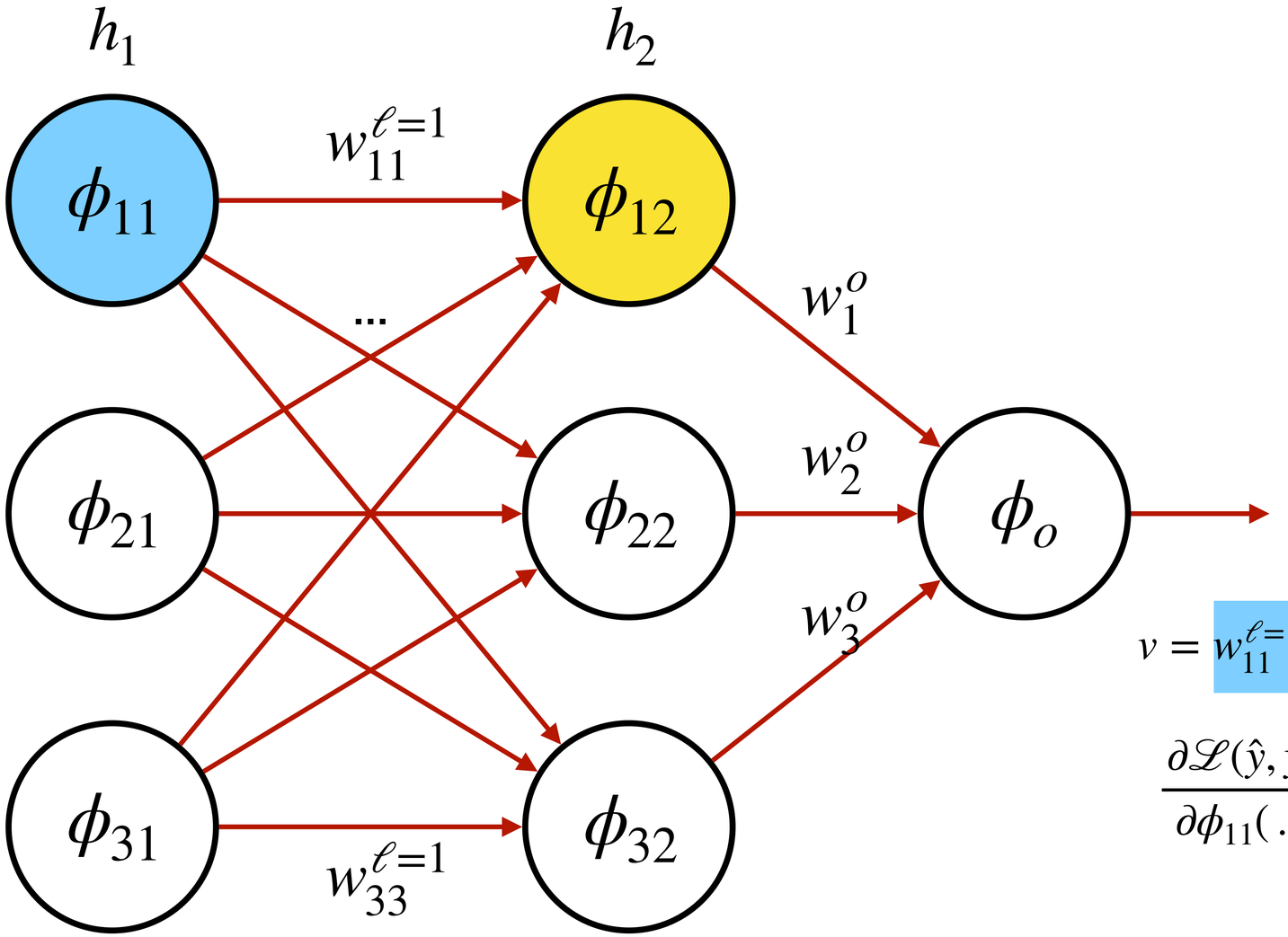
$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

$$v = w_{11}^{\ell=1} \times \phi_{11}(.) + w_{21}^{\ell=1} \times \phi_{21}(.) + w_{31}^{\ell=1} \times \phi_{31}(.)$$

$$\frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \phi_{11}(.)} = \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(v)} \frac{\partial \phi_{12}(v)}{\partial v} \frac{\partial v}{\partial \phi_{11}(.)}$$

$$= \frac{\partial \mathscr{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0 \frac{\partial \phi_{12}(v)}{\partial v} w_{11}^{\ell=1}$$

Backpropagation Review: FFNs



$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{12}(.)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(.)}$$

$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

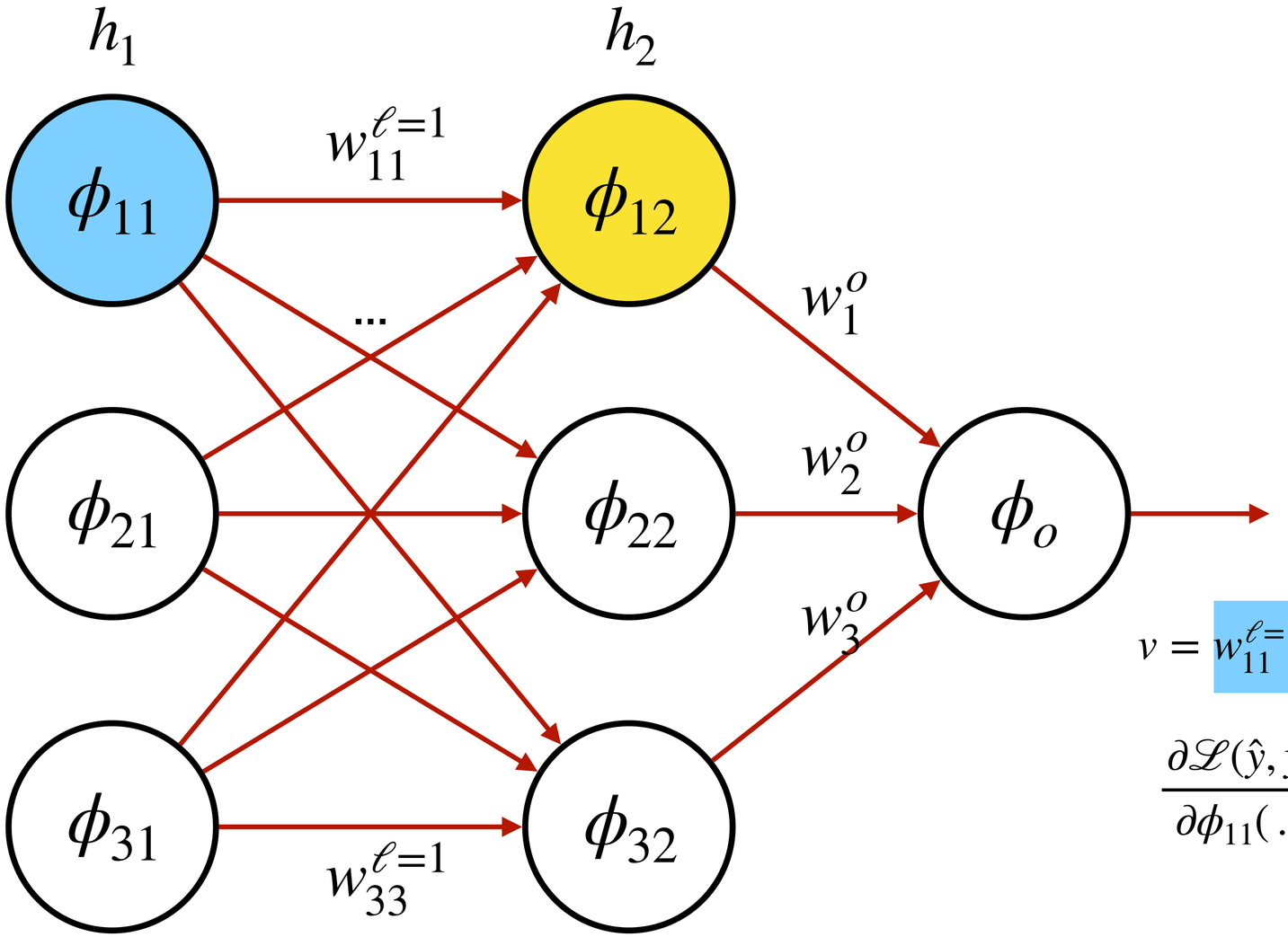
Depends on ϕ_{12}

$$v = w_{11}^{\ell=1} \times \phi_{11}(.) + w_{21}^{\ell=1} \times \phi_{21}(.) + w_{31}^{\ell=1} \times \phi_{31}(.)$$

$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{11}(.)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(v)} \frac{\partial \phi_{12}(v)}{\partial v} \frac{\partial v}{\partial \phi_{11}(.)}$$

$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0 \frac{\partial \phi_{12}(v)}{\partial v} w_{11}^{\ell=1}$$

Backpropagation Review: FFNs



$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{12}(.)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(.)}$$

$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

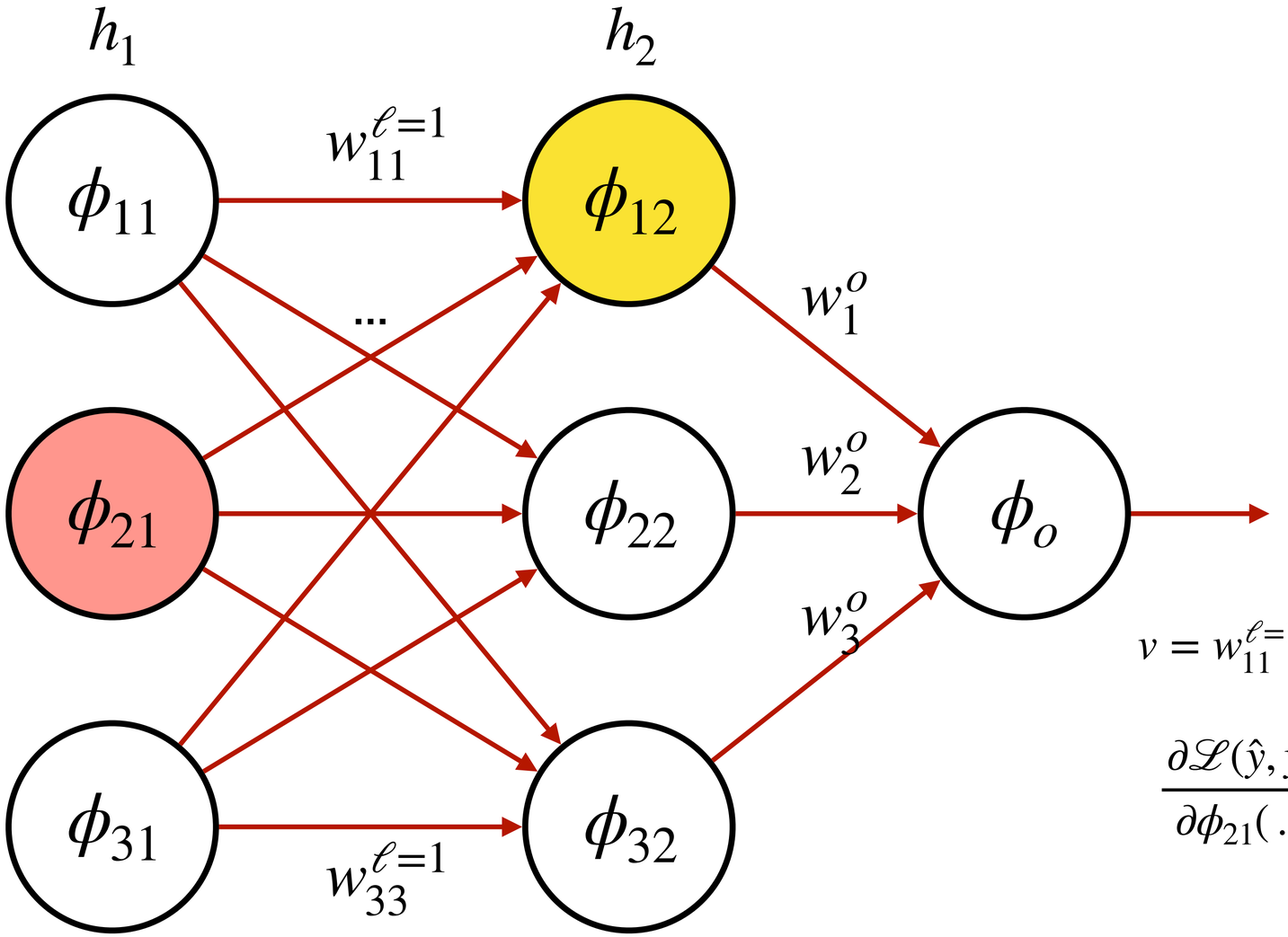
Depends on ϕ_{12}

$$v = w_{11}^{\ell=1} \times \phi_{11}(.) + w_{21}^{\ell=1} \times \phi_{21}(.) + w_{31}^{\ell=1} \times \phi_{31}(.)$$

$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{11}(.)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(v)} \frac{\partial \phi_{12}(v)}{\partial v} \frac{\partial v}{\partial \phi_{11}(.)}$$

$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0 \frac{\partial \phi_{12}(v)}{\partial v} w_{11}^{\ell=1}$$

Backpropagation Review: FFNs



$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{12}(.)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(.)}$$

$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0$$

Depends on ϕ_{12}

$$v = w_{11}^{\ell=1} \times \phi_{11}(.) + w_{21}^{\ell=1} \times \phi_{21}(.) + w_{31}^{\ell=1} \times \phi_{31}(.)$$

$$\frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \phi_{21}(.)} = \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial u} \frac{\partial u}{\partial \phi_{12}(v)} \frac{\partial \phi_{12}(v)}{\partial v} \frac{\partial v}{\partial \phi_{21}(.)}$$

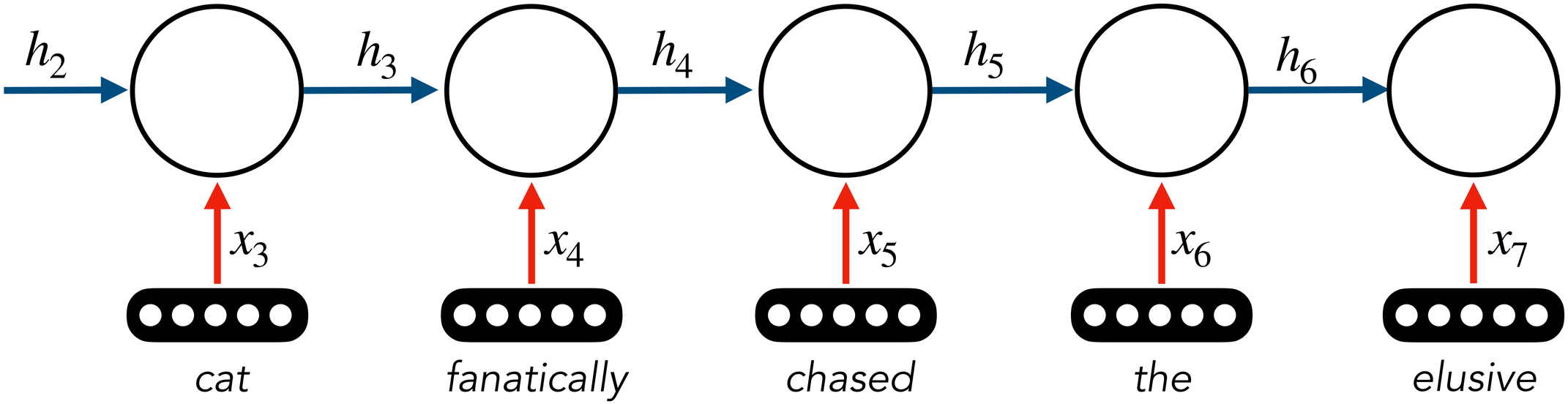
$$= \frac{\partial \mathcal{L}(\hat{y}, y)}{\partial \hat{y}} \frac{\partial \phi_o(u)}{\partial u} w_1^0 \frac{\partial \phi_{12}(v)}{\partial v} w_{21}^{\ell=1}$$

Question

How would we extend backpropagation to a recurrent neural network?

Recall

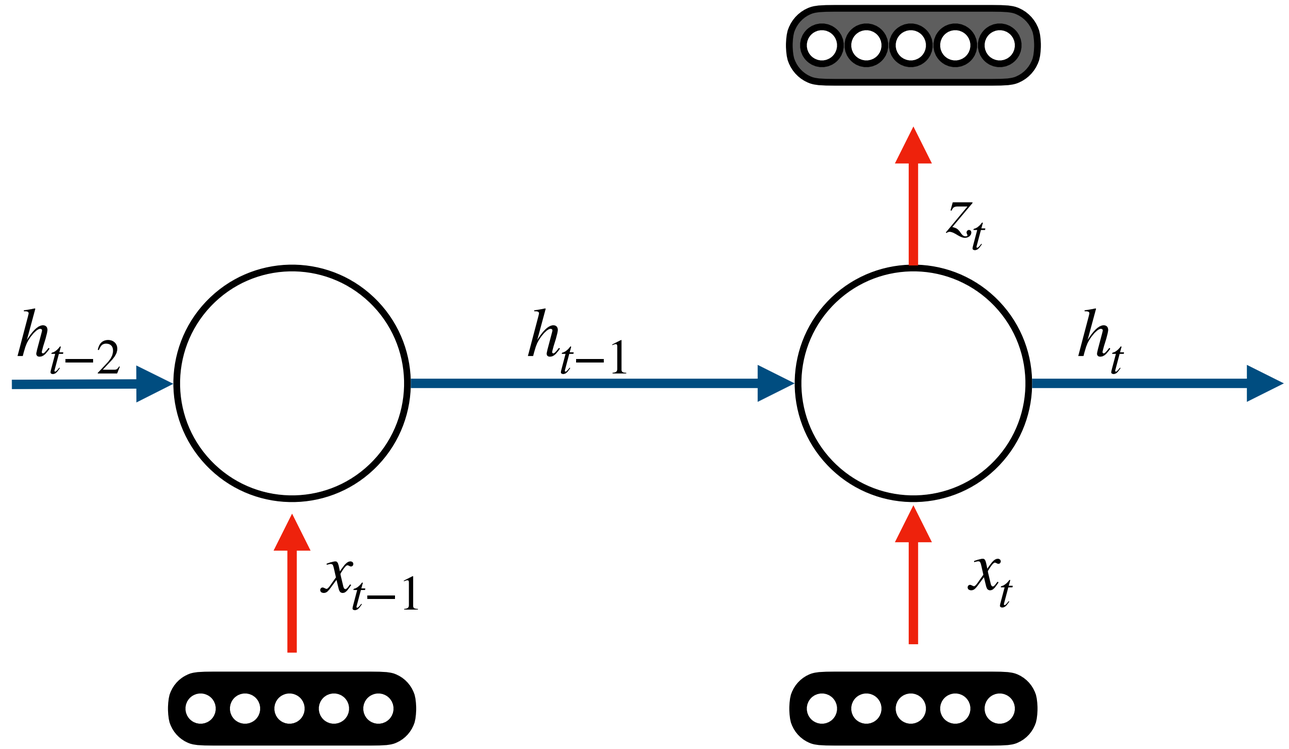
- RNN can be unrolled to a feedforward neural network
- Depth of feedforward neural network depends on length of the sequence



Backpropagation through Time

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$



Backpropagation through Time

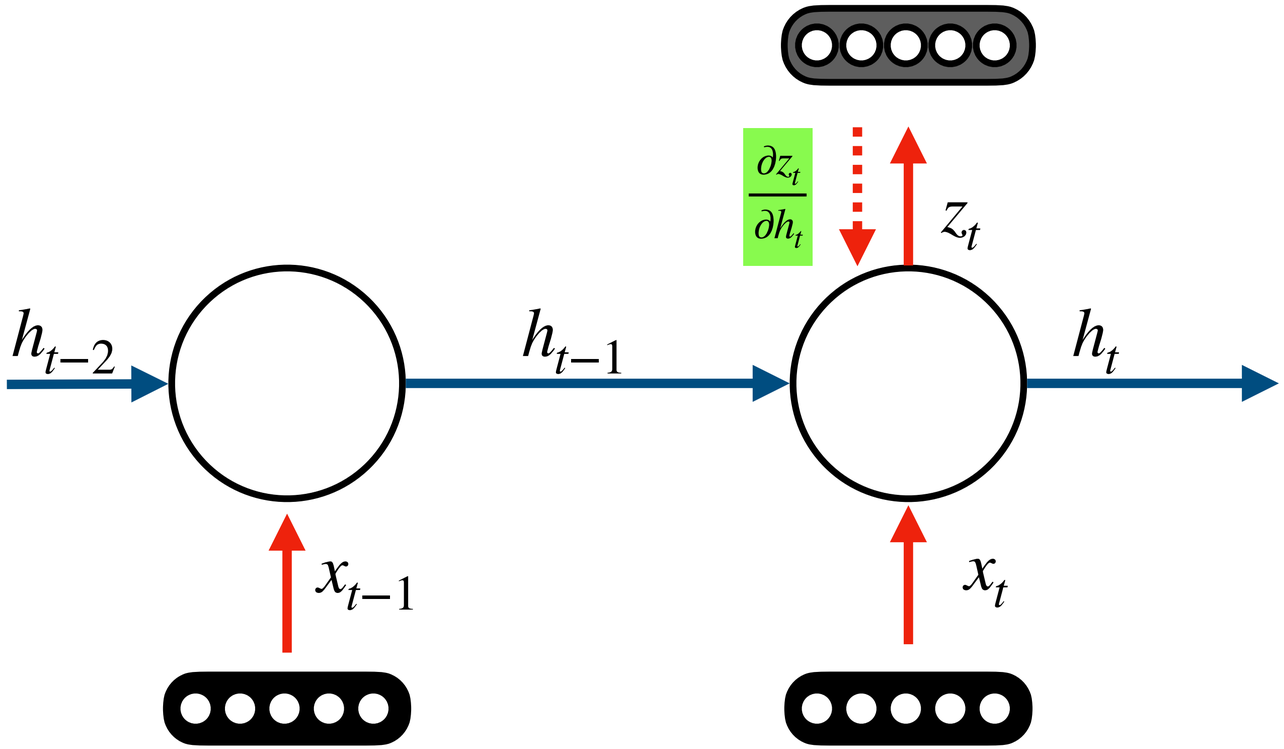
$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

$$v = W_{zh} h_t + b_z \qquad z_t = \sigma(v)$$

$$u = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} \frac{\partial v}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} W_{zh}$$



Backpropagation through Time

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

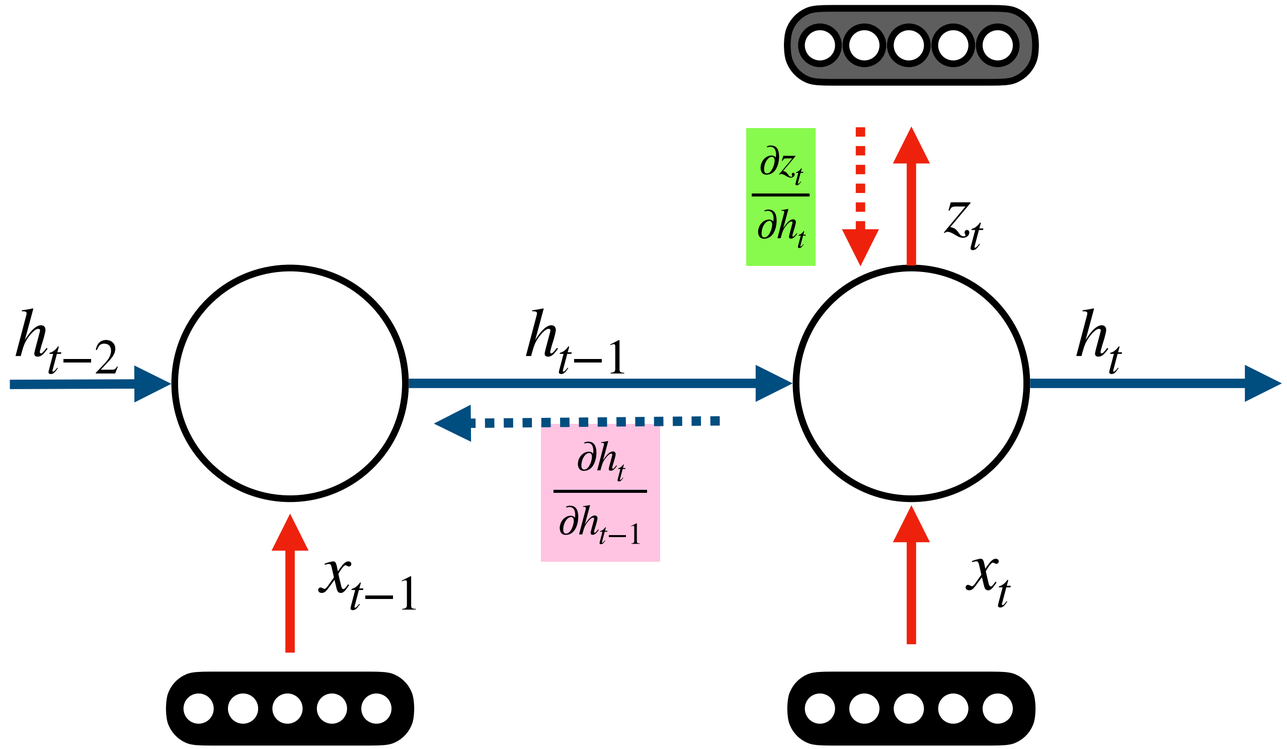
$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

$$v = W_{zh} h_t + b_z \qquad z_t = \sigma(v)$$

$$u = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} \frac{\partial v}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u} \frac{\partial u}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u} W_{hh}$$



Backpropagation through Time

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

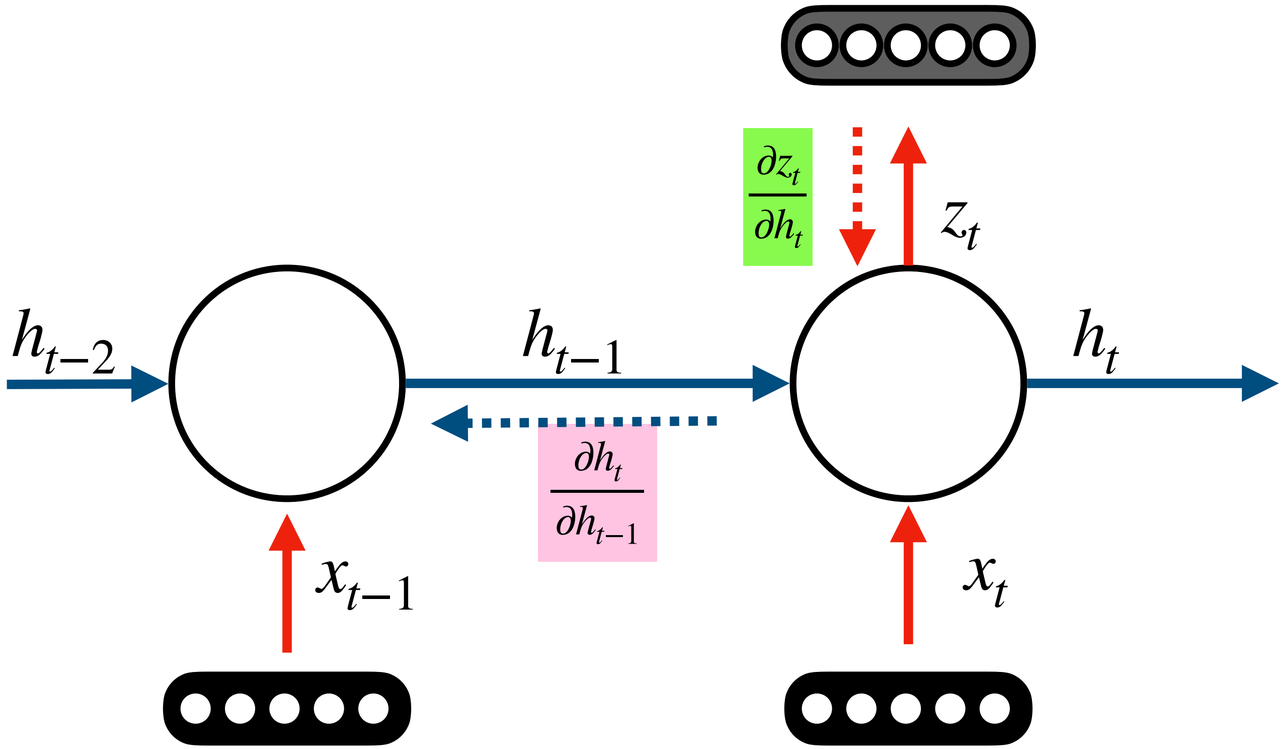
$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

$$v = W_{zh} h_t + b_z \qquad z_t = \sigma(v)$$

$$u = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} \frac{\partial v}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u} \frac{\partial u}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u} W_{hh}$$



$$\frac{\partial z_t}{\partial h_{t-1}} = \frac{\partial z_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t-1}}$$

Backpropagation through Time

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

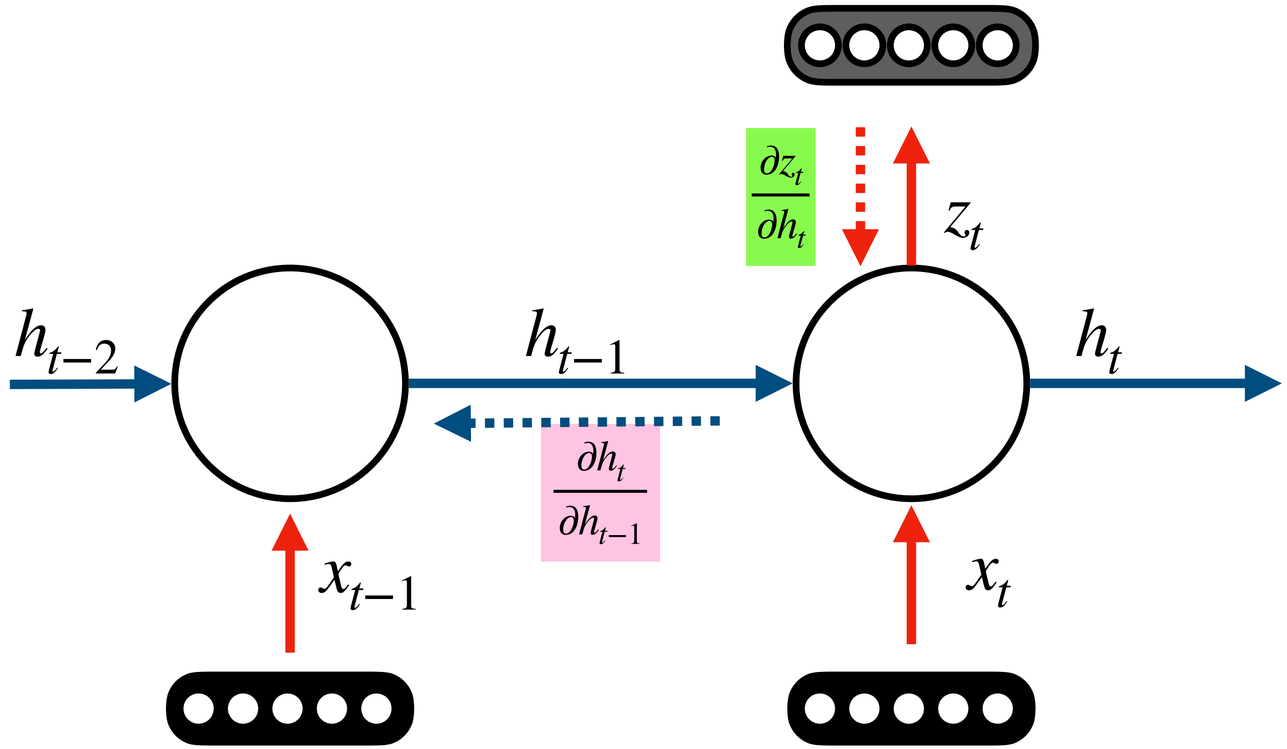
$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

$$v = W_{zh} h_t + b_z \qquad z_t = \sigma(v)$$

$$u = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} \frac{\partial v}{\partial h_t} = \frac{\partial \sigma(v)}{\partial v} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u} \frac{\partial u}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u} W_{hh}$$



$$\frac{\partial z_t}{\partial h_{t-1}} = \frac{\partial z_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(v)}{\partial v} \frac{\partial v}{\partial h_t} \frac{\partial \sigma(u)}{\partial u} \frac{\partial u}{\partial h_{t-1}} = \frac{\partial \sigma(v)}{\partial v} W_{zh} \frac{\partial \sigma(u)}{\partial u} W_{hh}$$

Backpropagation through Time

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

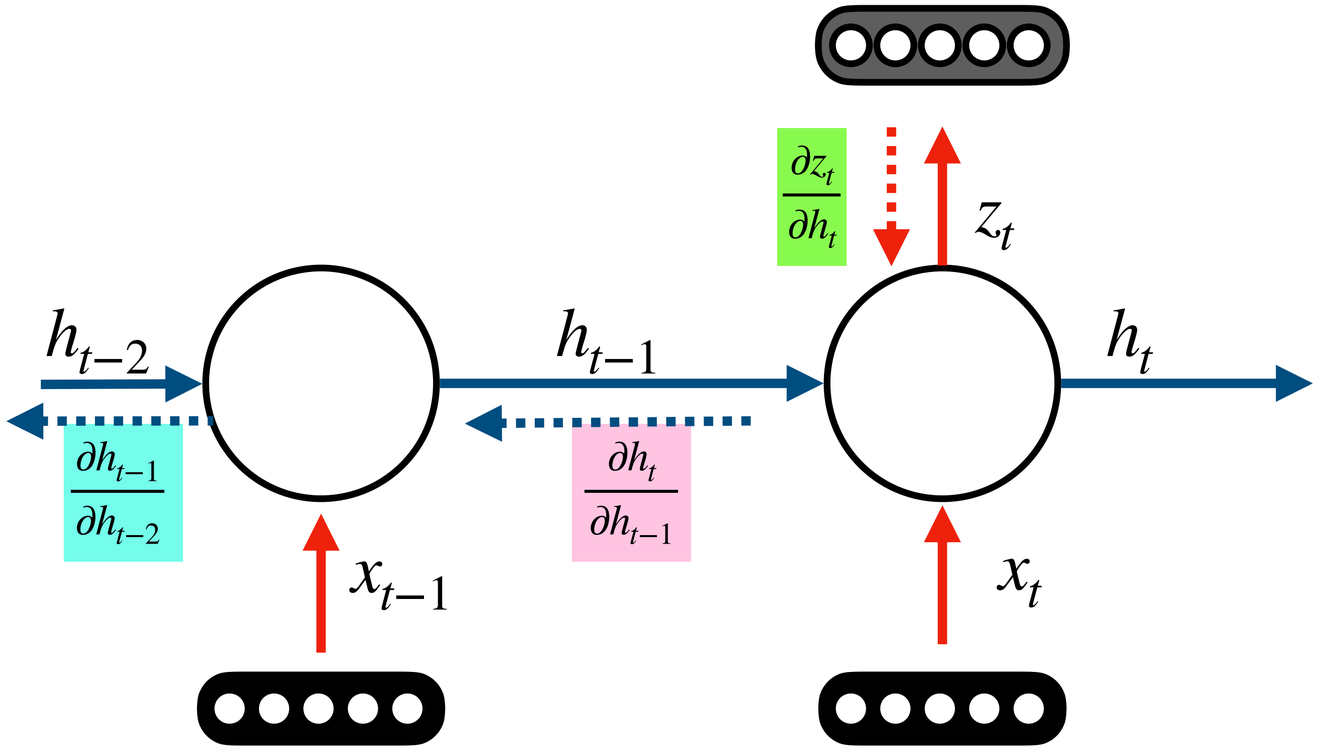
$$v_t = W_{zh} h_t + b_z \qquad z_t = \sigma(v_t)$$

$$u_t = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u_t)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t} \frac{\partial v_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t} \frac{\partial u_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh}$$

$$\frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} \frac{\partial u_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$



$$\frac{\partial z_t}{\partial h_{t-1}} = \frac{\partial z_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh} \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh} \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$

Backpropagation through Time

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

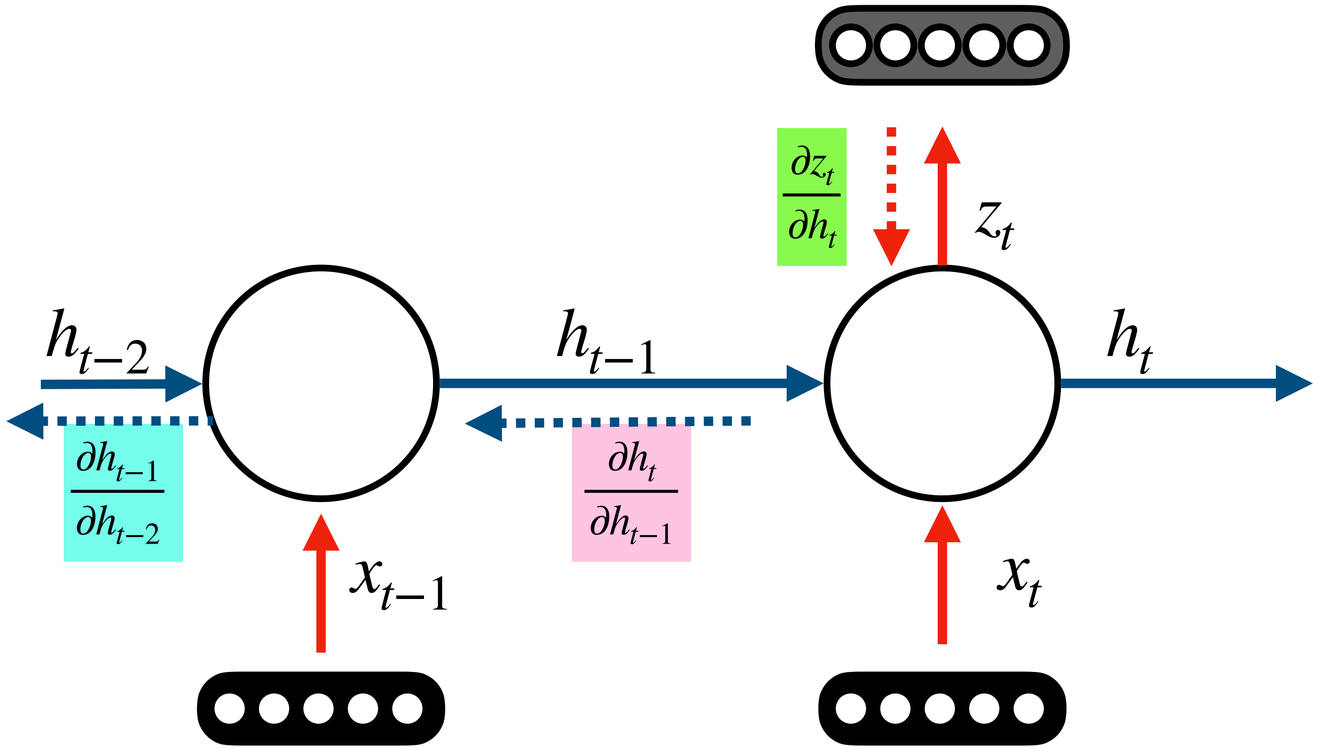
$$v_t = W_{zh} h_t + b_z \qquad z_t = \sigma(v_t)$$

$$u_t = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u_t)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t} \frac{\partial v_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t} \frac{\partial u_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh}$$

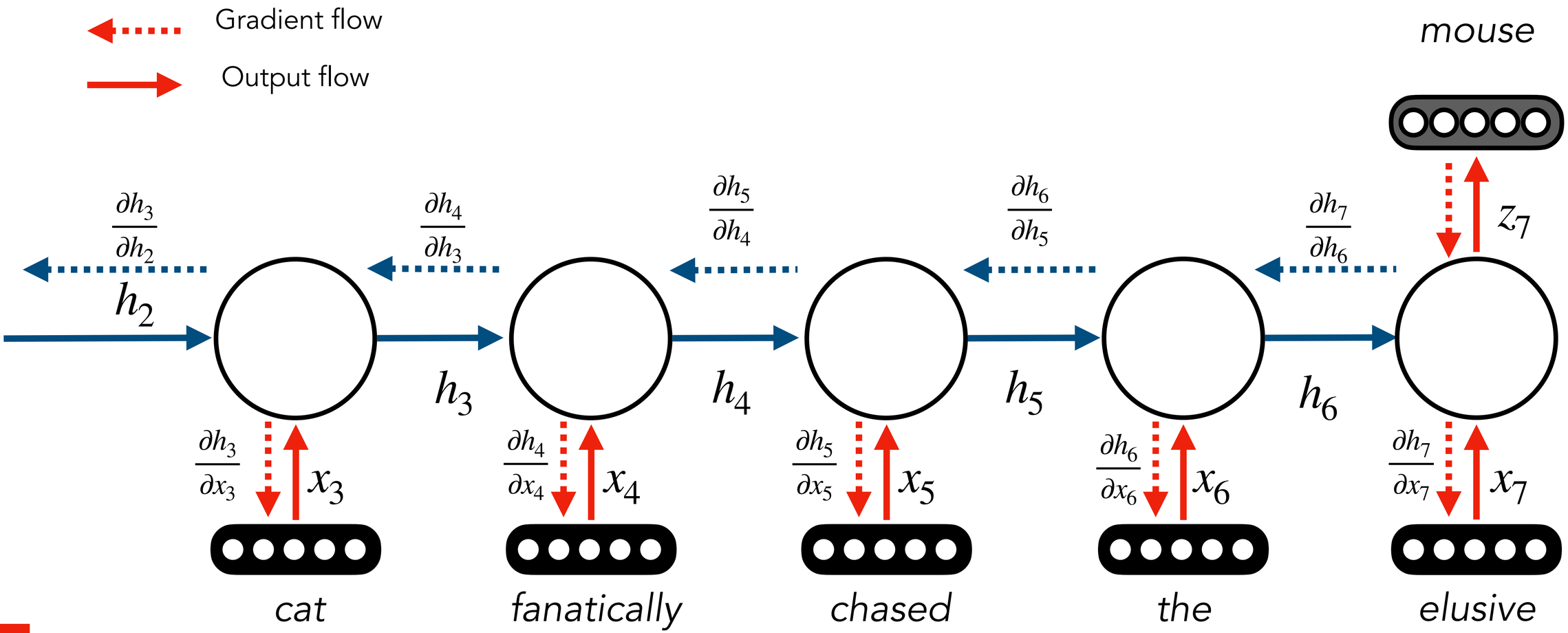
$$\frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} \frac{\partial u_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$



$$\frac{\partial z_t}{\partial h_{t-1}} = \frac{\partial z_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh} \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh} \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$

Note that these are actually the same matrix

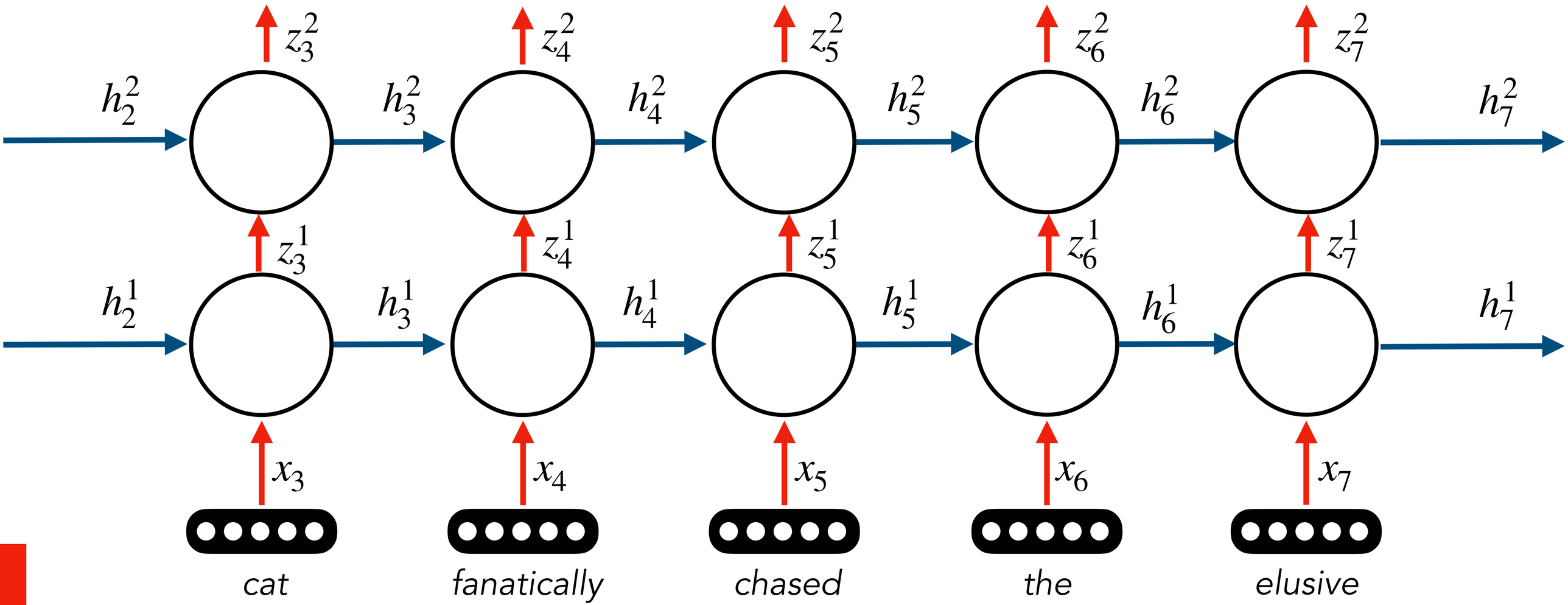
Backpropagation through time

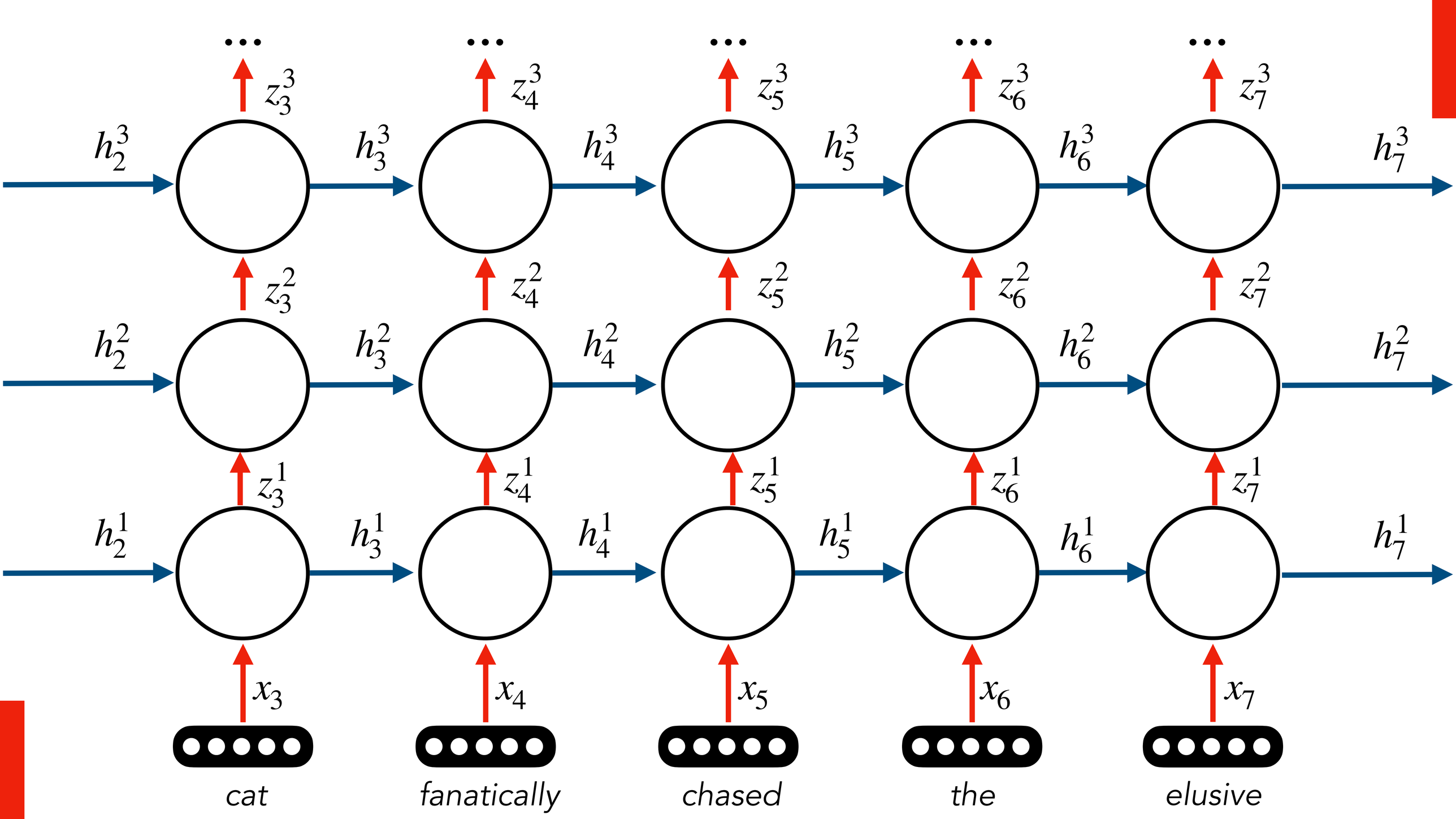


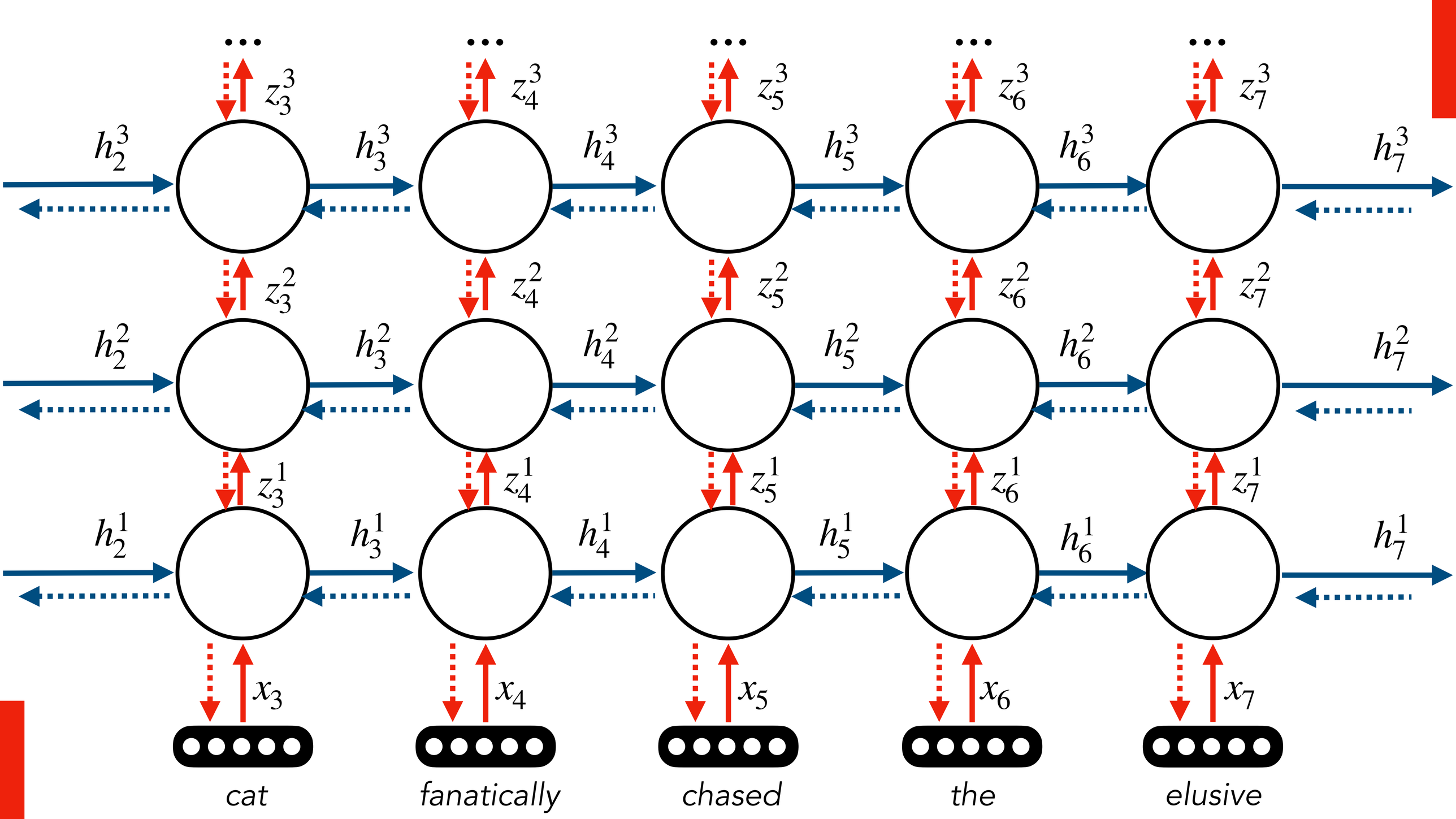
RNNs In Practice

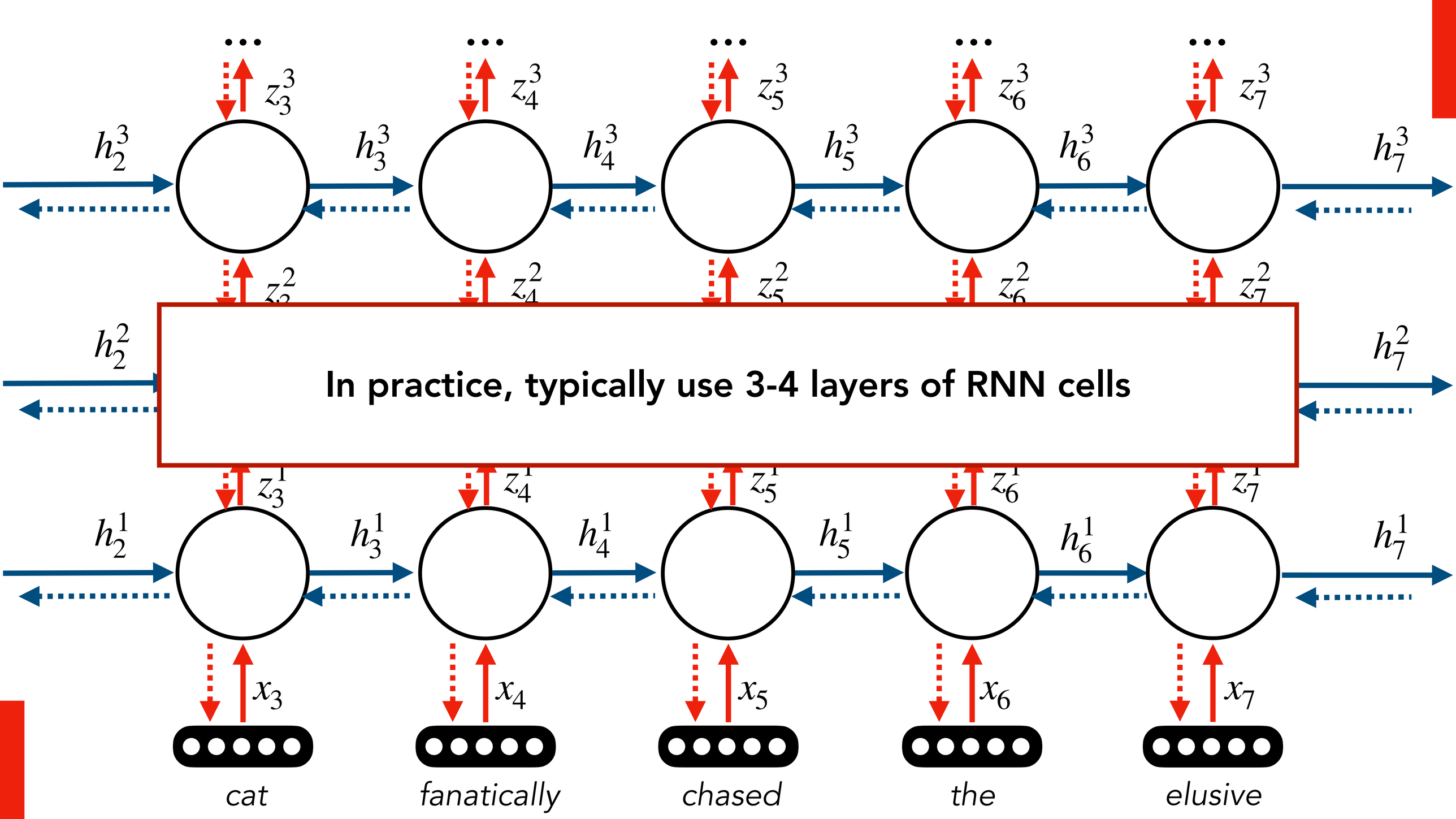
- Computing gradients by hand is hard!
  - Though potentially a good way to sanity check whether you network behaves the way you expect
- Most modern software packages for deep learning use automatic differentiation to compute gradients automatically from the forward pass
- Only need to define the forward pass of your model (much easier!)
- You'll use PyTorch in this class! You won't have to compute gradients by hand :)

Multiple Layers

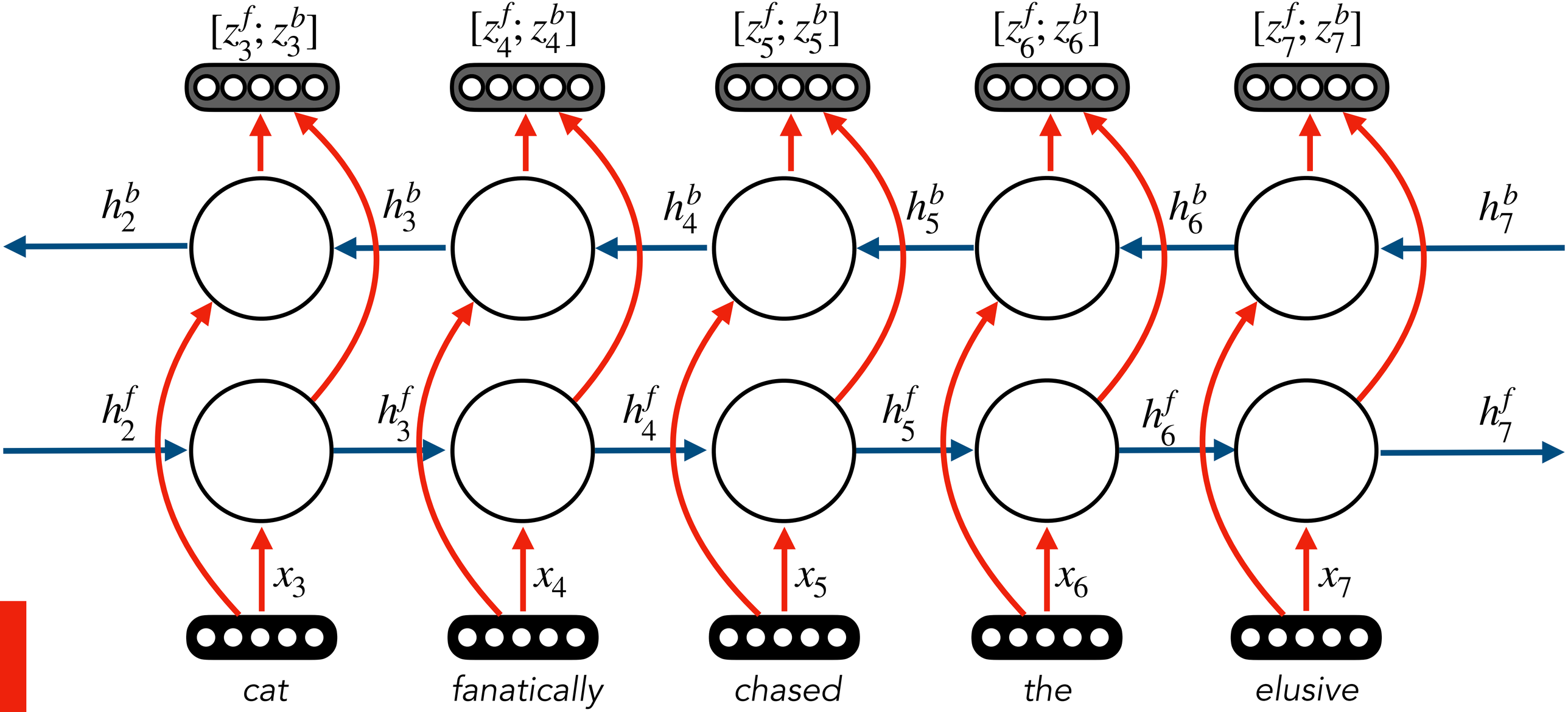



z_3^3 z_4^3 z_5^3 z_6^3 z_7^3
h_2^3 h_3^3 h_4^3 h_5^3 h_6^3 h_7^3
z_3^2 z_4^2 z_5^2 z_6^2 z_7^2
h_2^2 h_3^2 h_4^2 h_5^2 h_6^2 h_7^2
z_3^1 z_4^1 z_5^1 z_6^1 z_7^1
h_2^1 h_3^1 h_4^1 h_5^1 h_6^1 h_7^1
x_3 x_4 x_5 x_6 x_7
cat
fanatically
chased
the
elusive

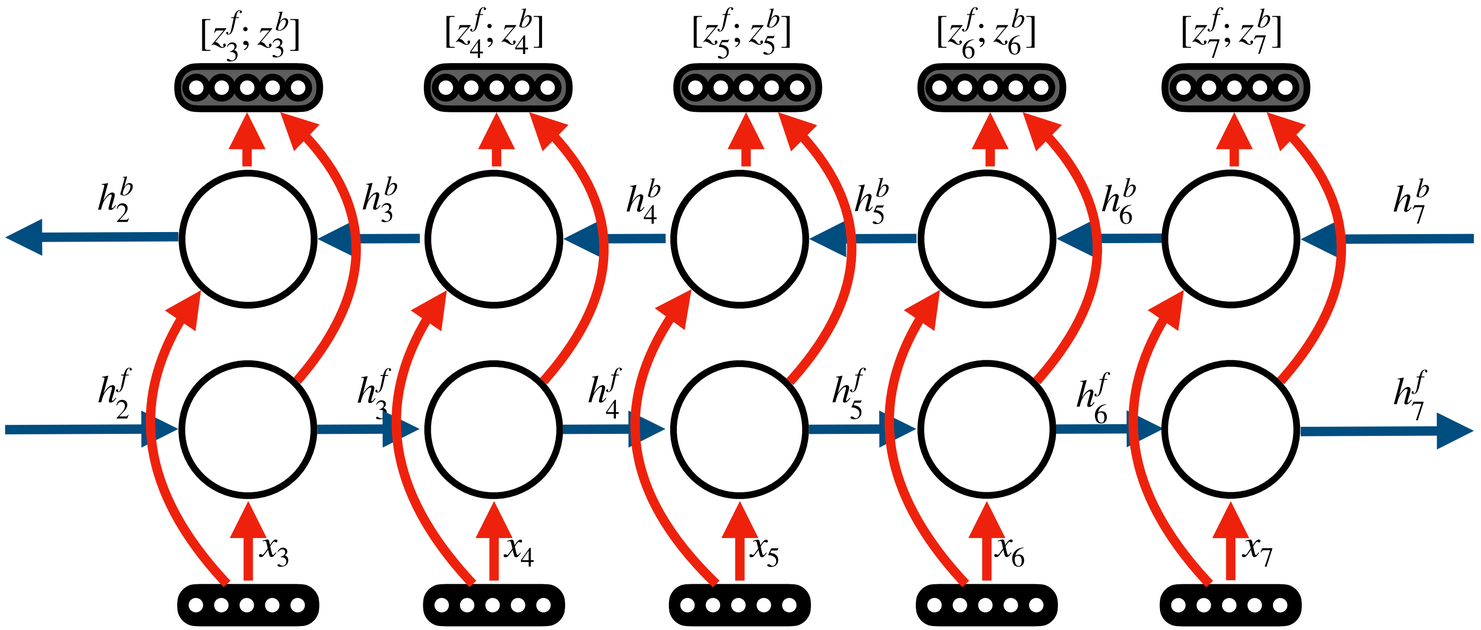

z_3^3 z_4^3 z_5^3 z_6^3 z_7^3
h_2^3 h_3^3 h_4^3 h_5^3 h_6^3 h_7^3
z_3^2 z_4^2 z_5^2 z_6^2 z_7^2
h_2^2 h_3^2 h_4^2 h_5^2 h_6^2 h_7^2
z_3^1 z_4^1 z_5^1 z_6^1 z_7^1
h_2^1 h_3^1 h_4^1 h_5^1 h_6^1 h_7^1
x_3 x_4 x_5 x_6 x_7
cat
fanatically
chased
the
elusive


In practice, typically use 3-4 layers of RNN cells
cat
fanatically
chased
the
elusive

Bidirectionality



Bidirectionality



- **Concatenate** the output states for final representation at each step
  - Can also do mean / max
- Separate parameters for forward and backward RNNs
- If you can use the future text for the task, then you should use **bidirectionality**

Question

For which of the following task types can we use a bidirectional RNN?

(a) Classification

(b) Sequence labelling

(c) Text Generation

Recap

- Neural language models allow us to *share information* among similar sequences by learning neural representations that similarly represent them
- **Problem:** Fixed context language models can only process a limited window of the word history at a time
- **Solution:** recurrent neural networks can **theoretically** learn to model an **unbounded context length**

Issue with Recurrent Models

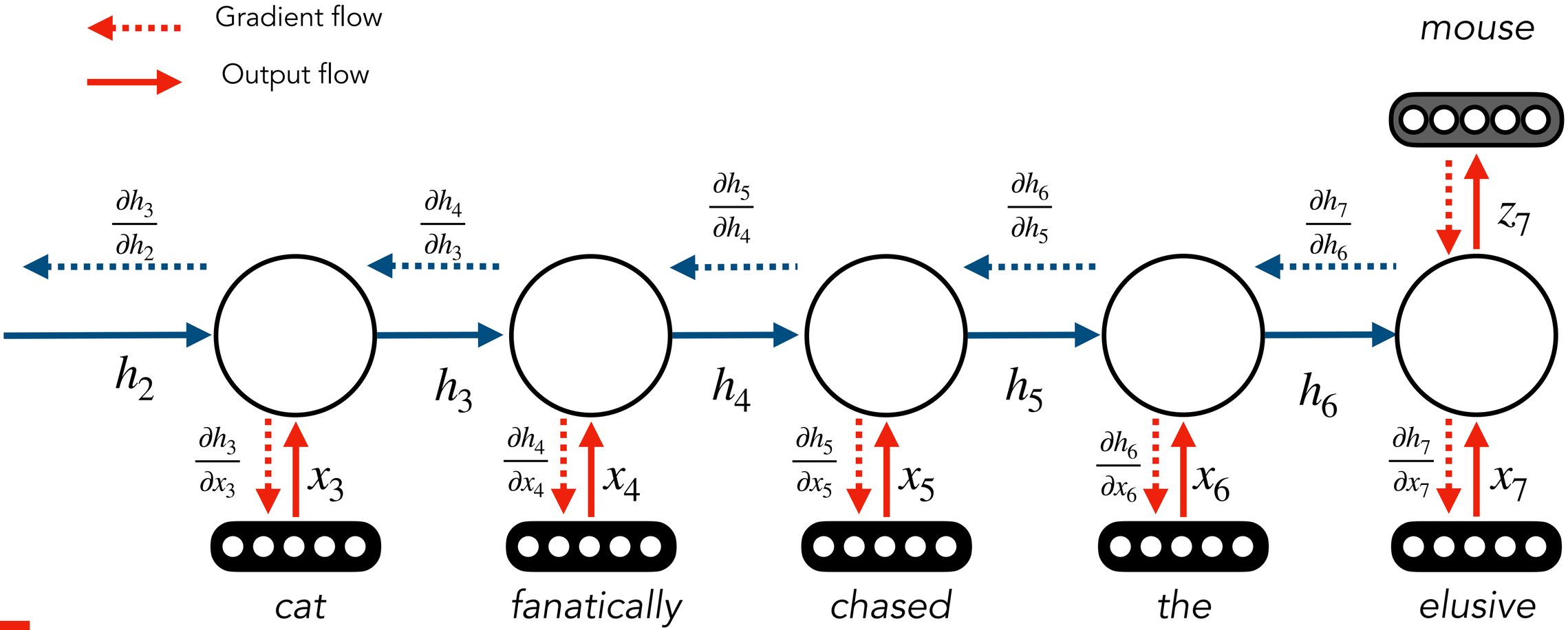
- Multiple steps of state overwriting makes it challenging to learn long-range dependencies.

> *They tuned, discussed for a moment, then struck up a lively* ***jig****. Everyone joined in, turning the courtyard into an even more chaotic scene, people now* ***dancing*** *in circles,* ***swinging*** *and* ***spinning*** *in circles, everyone making up their own* ***dance steps****. I felt my feet tapping, my body wanting to move. Aside from writing, I 've always loved* **dancing** *.*

- Nearby words should affect each other more than farther ones, but RNNs make it challenging to learn **any** long-range interactions

LAMBADA dataset, 2016

Backpropagation through time



Vanishing Gradients

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

$$v_t = W_{zh} h_t + b_z \qquad z_t = \sigma(v_t)$$

$$u_t = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u_t)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t}\frac{\partial v_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t}\frac{\partial u_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh}$$

$$\frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}}\frac{\partial u_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$

$$\frac{\partial z_t}{\partial h_{t-2}} = \frac{\partial z_t}{\partial h_t}\frac{\partial h_t}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh} \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh} \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$

Generalising this:

$$\frac{\partial h_t}{\partial h_{t-T}} = \prod_{i=t-T}^{i=t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=t-T}^{i=t} \frac{\partial \sigma(u_i)}{\partial u_i} W_{hh}$$

Vanishing Gradients

$$z_t = \sigma\left(W_{zh} h_t + b_z\right)$$

$$h_t = \sigma\left(W_{hx} x_t + W_{hh} h_{t-1} + b_h\right)$$

$$v_t = W_{zh} h_t + b_z \qquad z_t = \sigma(v_t)$$

$$u_t = W_{hx} x_t + W_{hh} h_{t-1} + b_h \qquad h_t = \sigma(u_t)$$

$$\frac{\partial z_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t}\frac{\partial v_t}{\partial h_t} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh}$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t}\frac{\partial u_t}{\partial h_{t-1}} = \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh}$$

$$\frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}}\frac{\partial u_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$

$$\frac{\partial z_t}{\partial h_{t-2}} = \frac{\partial z_t}{\partial h_t}\frac{\partial h_t}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial h_{t-2}} = \frac{\partial \sigma(v_t)}{\partial v_t} W_{zh} \frac{\partial \sigma(u_t)}{\partial u_t} W_{hh} \frac{\partial \sigma(u_{t-1})}{\partial u_{t-1}} W_{hh}$$

Generalising this:

$$\frac{\partial h_t}{\partial h_{t-T}} = \prod_{i=t-T}^{i=t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=t-T}^{i=t} \frac{\partial \sigma(u_i)}{\partial u_i} W_{hh}$$

< 1 for many activation fxns

Typically small (Regularisation)

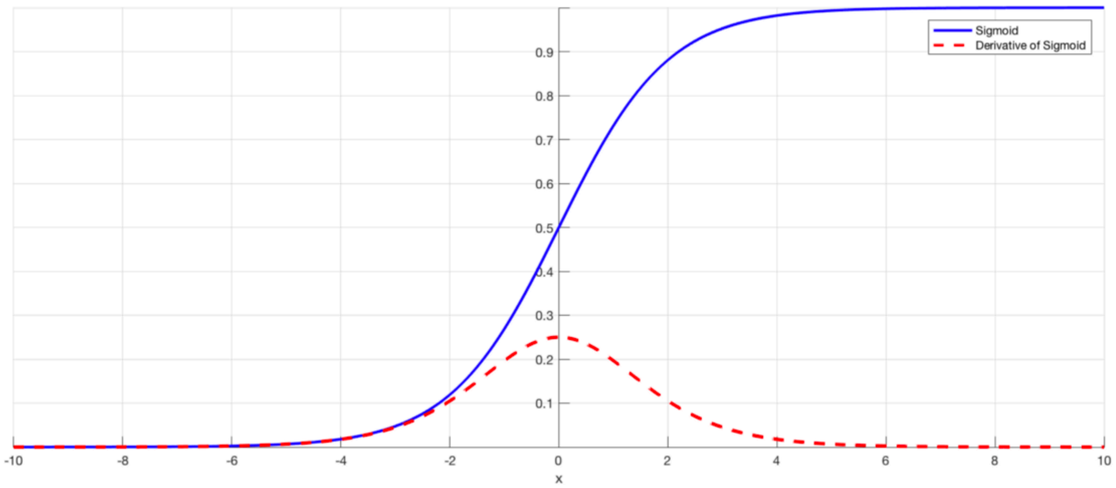
Vanishing Gradients

- **Learning Problem:** Long unrolled networks will crush gradients that backpropagate to earlier time steps

$$h_t = \sigma\left(W_{hx}x_t + W_{hh}h_{t-1} + b_h\right)$$

$$u = W_{hx}x_t + W_{hh}h_{t-1} + b_h$$

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u}\frac{\partial u}{\partial h_{t-1}} = W_{hh}\frac{\partial \sigma(u)}{\partial u}$$

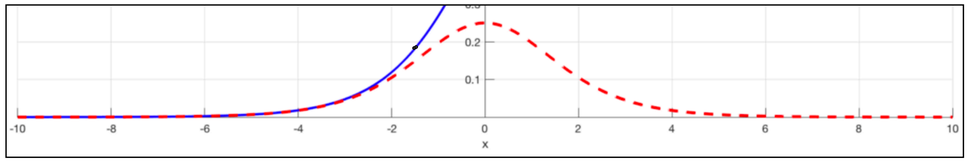


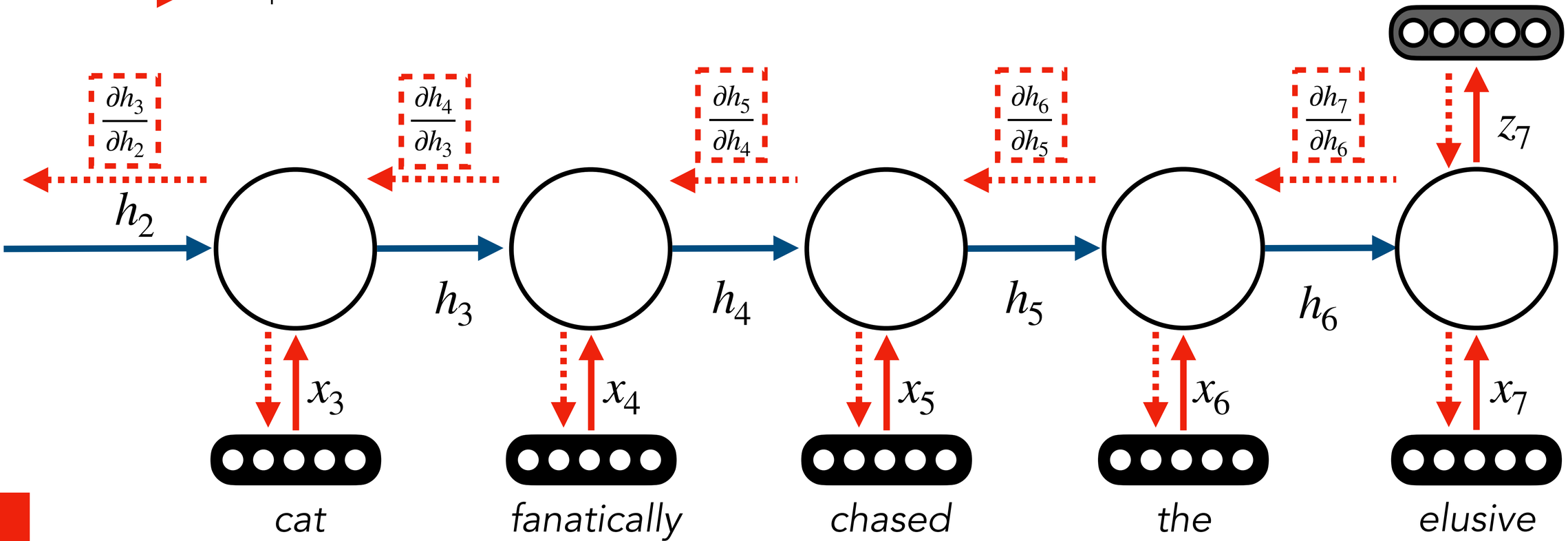
Vanishing Gradients

Gradient flow

Output flow

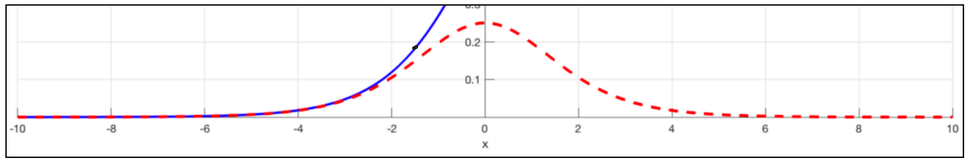
$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u}\frac{\partial u}{\partial h_{t-1}} = W_{hh}\frac{\partial \sigma(u)}{\partial u}$$





Vanishing Gradients

$$\frac{\partial h_t}{\partial h_{t-1}} = \frac{\partial \sigma(u)}{\partial u}\frac{\partial u}{\partial h_{t-1}} = W_{hh}\frac{\partial \sigma(u)}{\partial u}$$



Gradient flow

Output flow

Problem in many recurrent neural networks,

Especially pronounced in Elman networks (Vanilla RNNs) due to the sigmoid activation

$\frac{\partial h_3}{\partial h_2}$

h_2

z_7

x_3 x_4 x_5 x_6 x_7

cat *fanatically* *chased* *the* *elusive*

Question

How could we fix this vanishing gradient problem?

$$\frac{\partial h_t}{\partial h_{t-T}} = \prod_{i=t-T}^{i=t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=t-T}^{i=t} \frac{\partial \sigma(u_i)}{\partial u_i} W_{hh}$$

Typically less than one

Typically small (Regularisation)

Recap

- Early neural language models (and n-gram models) suffer from **fixed context windows**
- Recurrent neural networks can **theoretically** learn to model an **unbounded context length** using back propagation through time (BPTT)
- Practically, **vanishing gradients** can still stop many RNNs from learning **long-range dependencies**

References

- Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A Neural Probabilistic Language Model. *Journal of machine learning research*.
- Elman, J.L. (1990). Finding Structure in Time. *Cogn. Sci., 14*, 179-211.