Structured Neural Networks (I)

CS 690N, Spring 2018

Advanced Natural Language Processing

http://people.cs.umass.edu/~brenocon/anlp2018/

Brendan O'Connor

College of Information and Computer Sciences

University of Massachusetts Amherst

Structured neural networks?

- How to deal with arbitrary length inputs?
  - Documents, sentences, long-distance history
- Build structure directly into network architectures
  - Convolutional
  - Recurrent
- Dynamic autodiff frameworks make training easy(-ish) (PyTorch, DyNet)

“Averaging” network

- Continuous Bag-of-Words

$$\mathrm{CBOW}(f_1, ..., f_k) = \frac{1}{k}\sum_{i=1}^{k} v(f_i)$$

- Use averaged representation for e.g. softmax classifier
- Example: *FastText* doc classifier (Joulin et al. 2016)
  - Pre-trained word embeddings
  - Bag-of-words, Bag-of-ngrams
  - Hashing (ngram embeddings randomly shared)
  - Hierarchical softmax speed trick
  - With >100k sentiment labeled training docs, performs better than explicit feature logistic regression

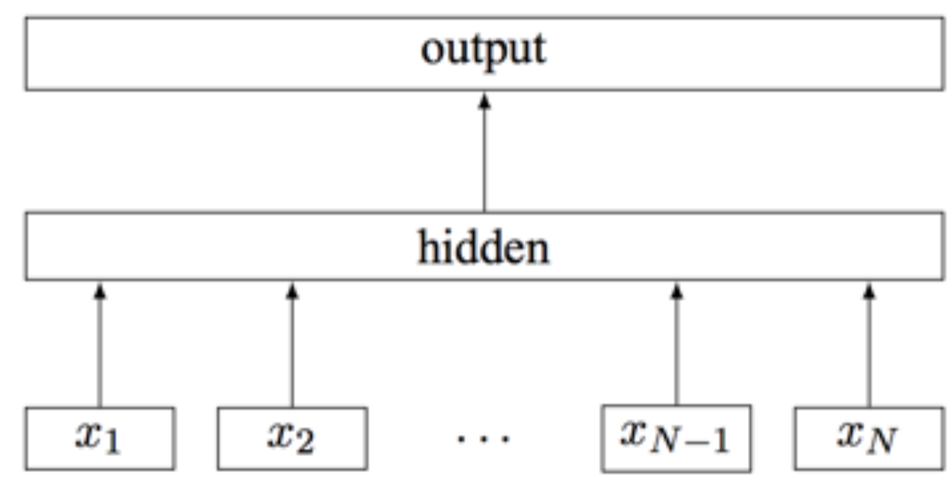
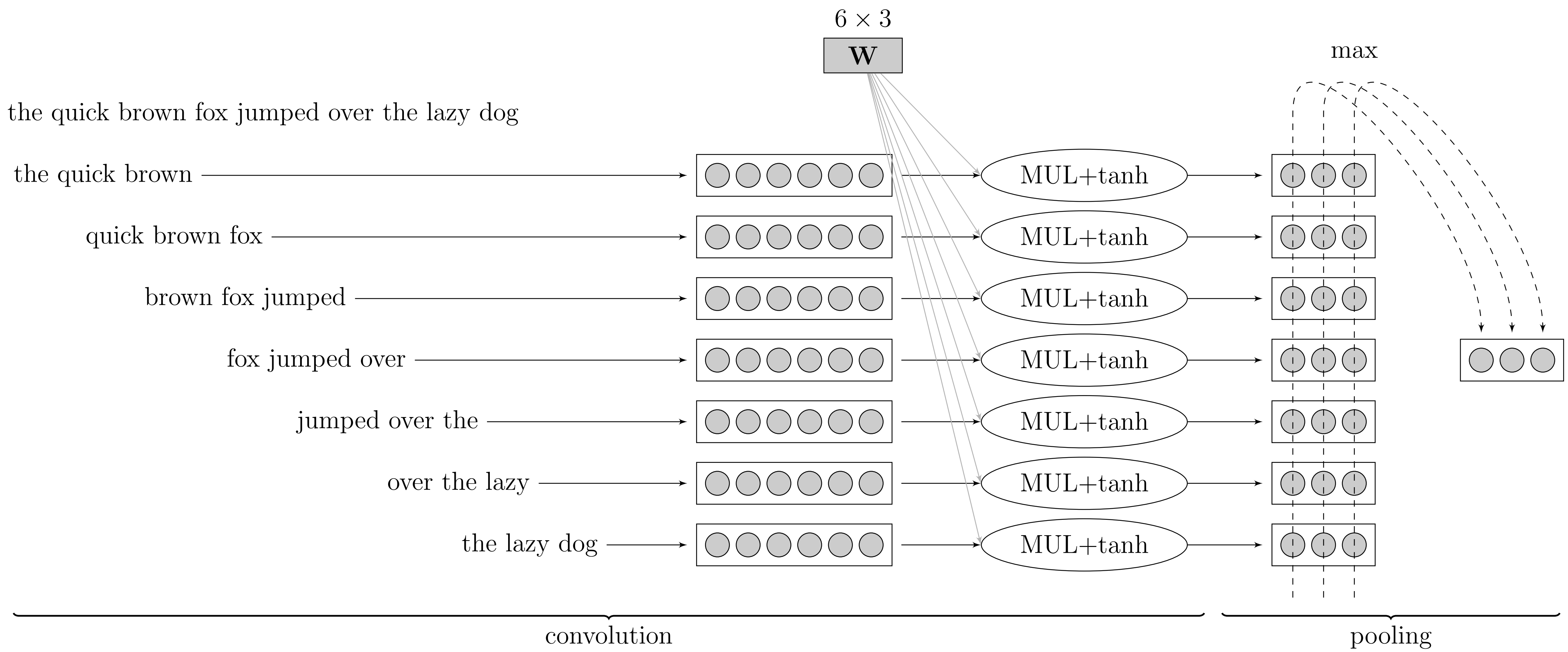


Figure 1: Model architecture of `fastText` for a sentence with N ngram features $x_1, \ldots, x_N$. The features are embedded and averaged to form the hidden variable.

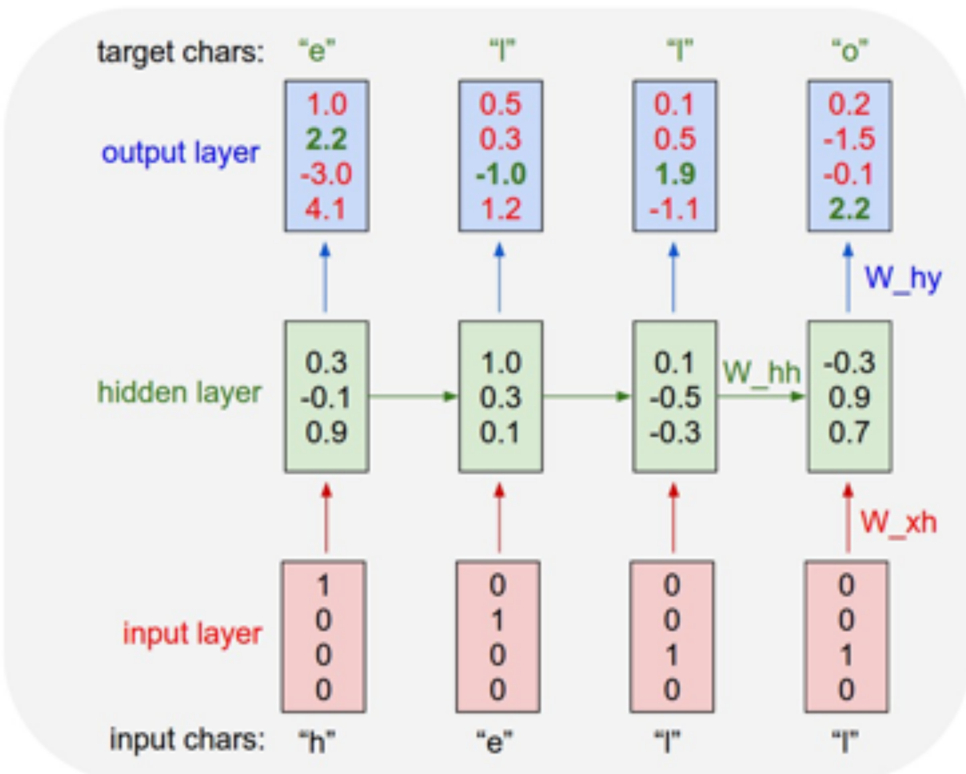
Convolutional NN



- Sentence representation independent of sentence length:
  - Sliding window of concatenated word embeddings
  - Feedforward transform then elementwise max across positions
- Final sentence representation could be used in various ways: e.g. classification (Kim 2014). Use joint training.
- Only learns local dependencies (like n-grams)

[Diagram: Yoav Goldberg]

Recurrent NN



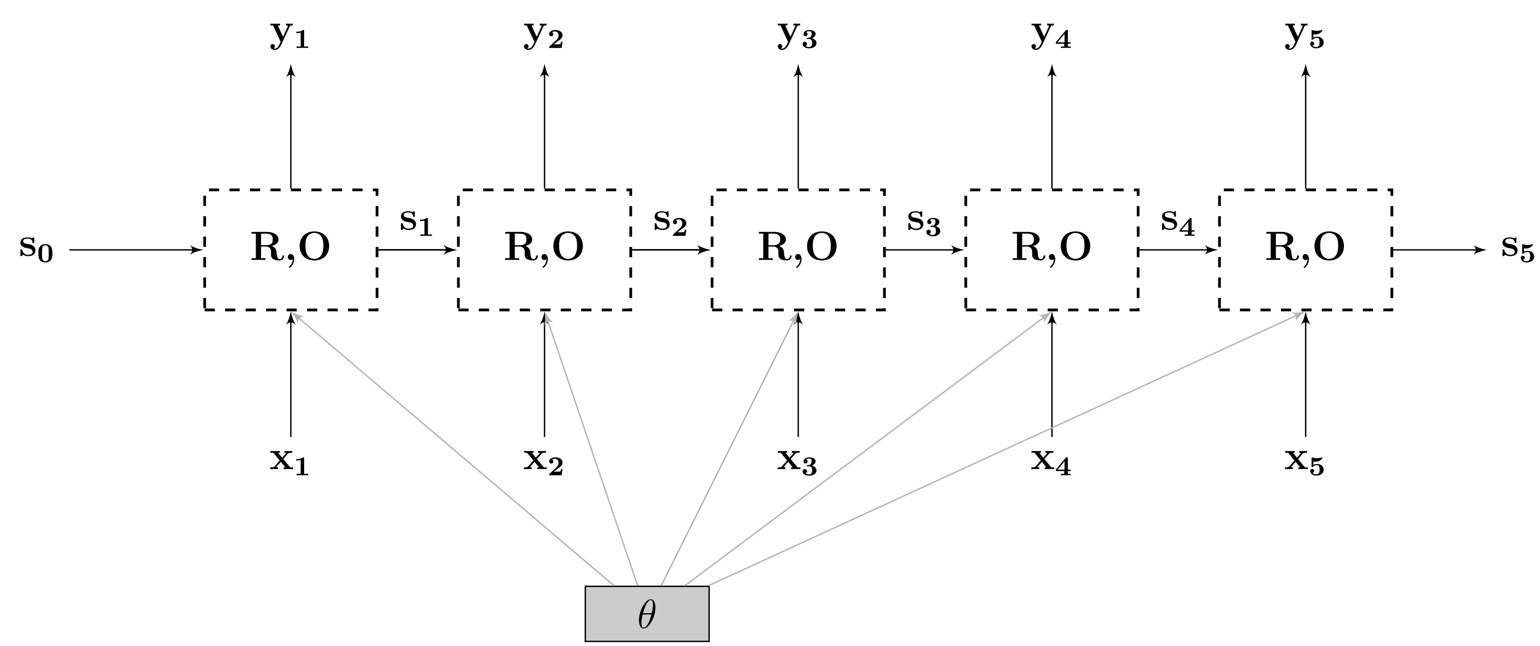


Figure 6: Graphical representation of an RNN (unrolled).

- Simple ("vanilla") RNN (Elman 1990)

$$\mathbf{s_i} = R_{\text{SRNN}}(\mathbf{s_{i-1}}, \mathbf{x_i}) = g(\mathbf{x_i}\mathbf{W^x} + \mathbf{s_{i-1}}\mathbf{W^s} + \mathbf{b})$$
$$\mathbf{y_i} = O_{\text{SRNN}}(\mathbf{s_i}) = \mathbf{s_i}$$

$$\mathbf{s_i}, \mathbf{y_i} \in \mathbb{R}^{d_s}, \ \ \mathbf{x_i} \in \mathbb{R}^{d_x}, \ \ \mathbf{W^x} \in \mathbb{R}^{d_x \times d_s}, \ \ \mathbf{W^s} \in \mathbb{R}^{d_s \times d_s}, \ \ \mathbf{b} \in \mathbb{R}^{d_s}$$

- Other local models: LSTM and GRU

RNN Uses

- Acceptor

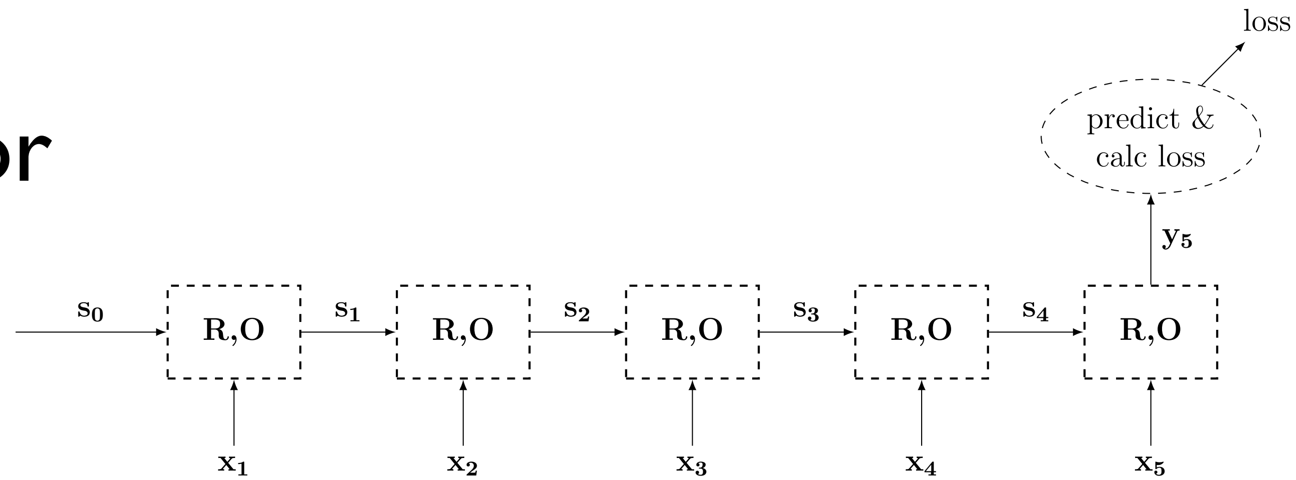
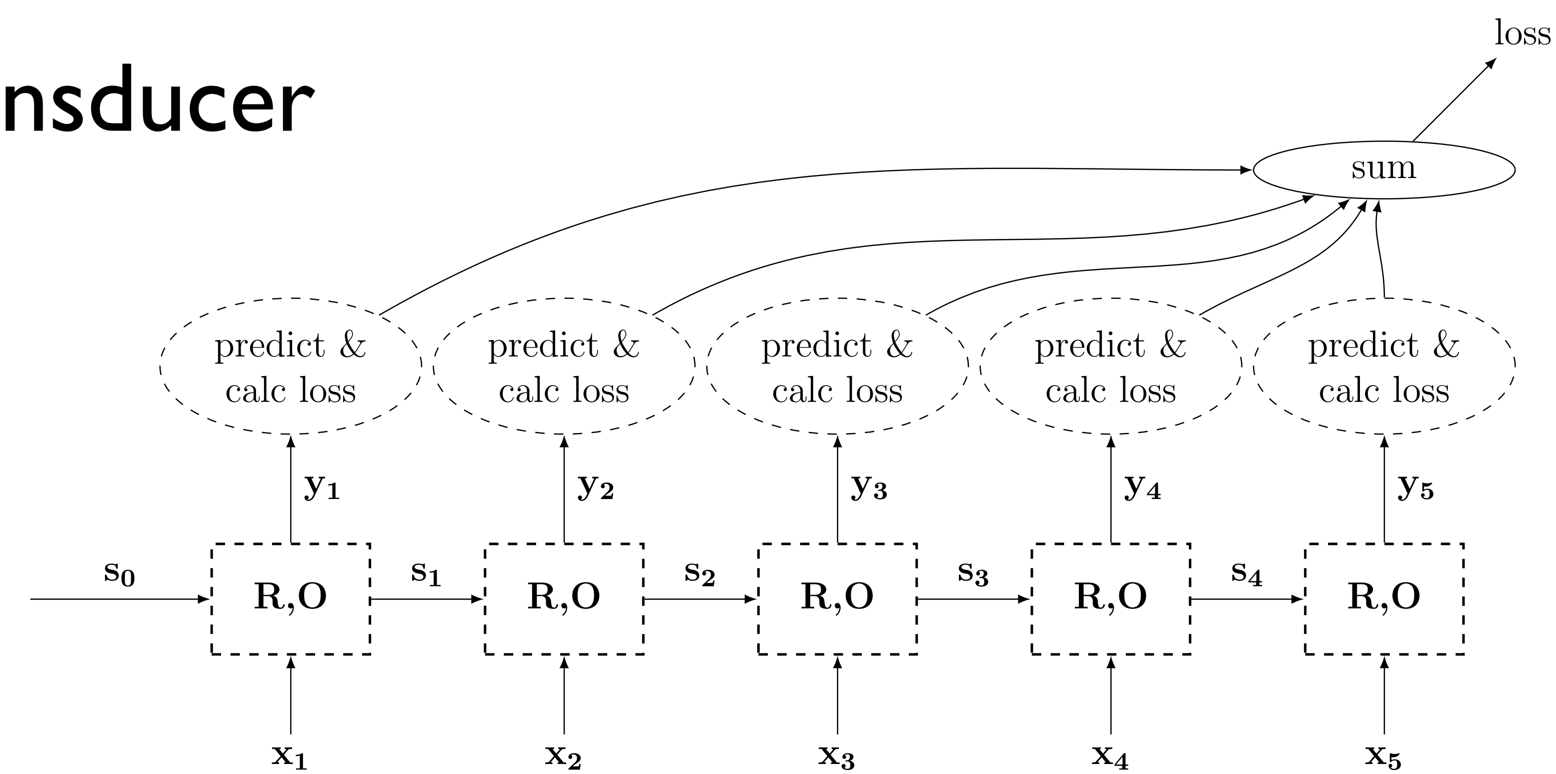


Figure 7: Acceptor RNN Training Graph.

- Transducer



[Diagram: Yoav Goldberg]

RNN Uses

- Encoder-decoder

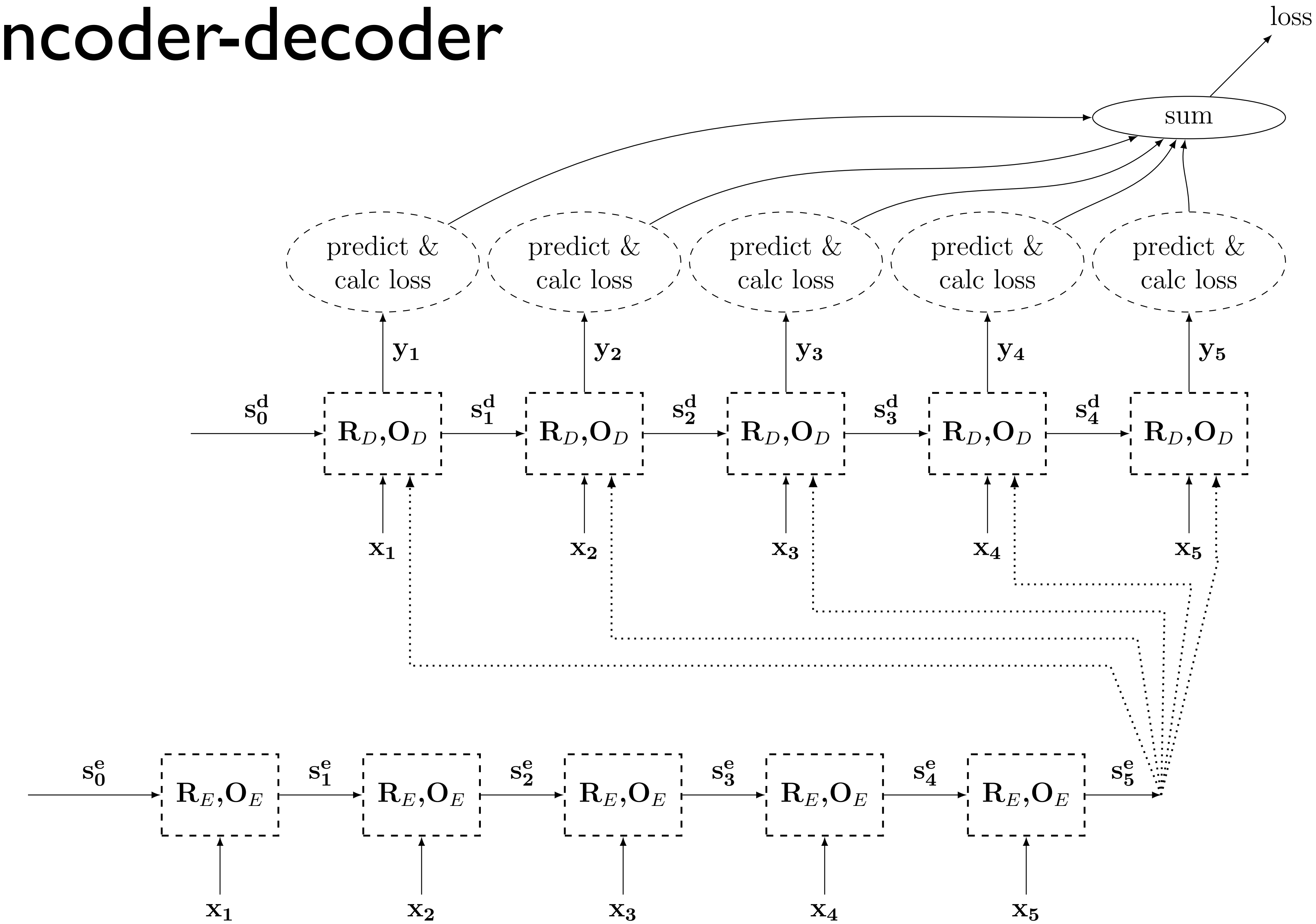


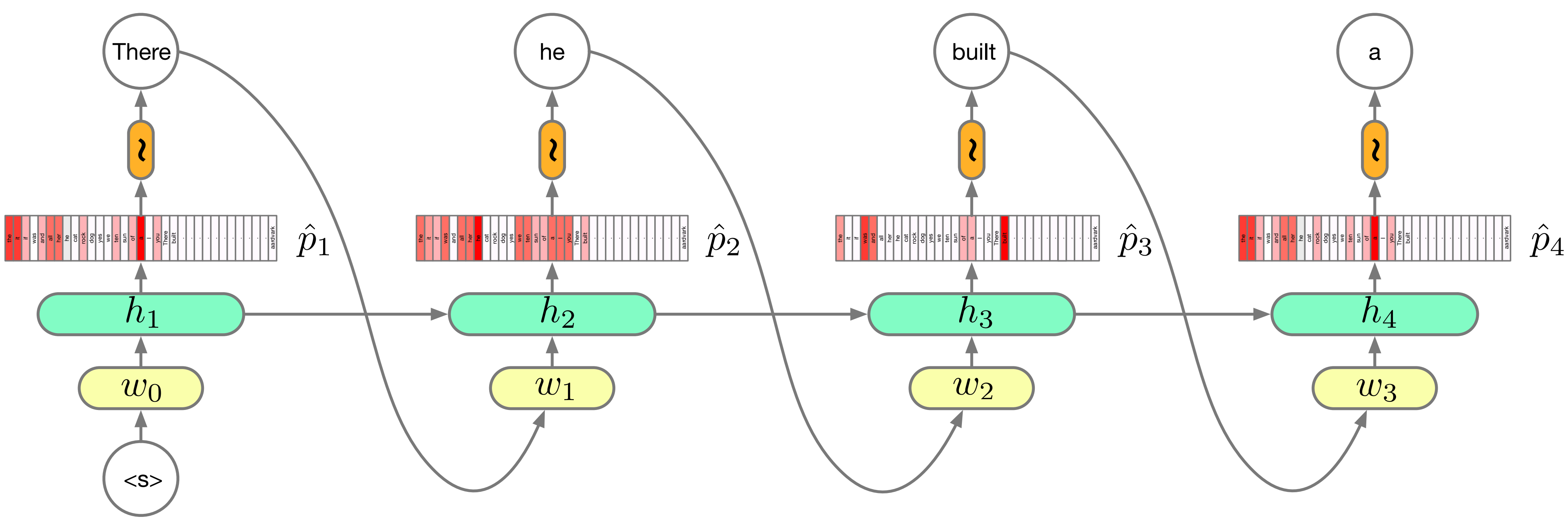
Figure 9: Encoder-Decoder RNN Training Graph.

[Diagram: Yoav Goldberg]

Language Modelling: Review

Language models aim to represent the history of observed text $(w_1, \ldots, w_{t-1})$ succinctly in order to predict the next word (w_t):

- With count based n-gram LMs we approximate the history with just the previous n words.
- Neural n-gram LMs embed the same fixed n-gram history in a continues space and thus capture correlations between histories.
- With Recurrent Neural Network LMs we drop the fixed n-gram history and compress the entire history in a fixed length vector, enabling long range correlations to be captured.



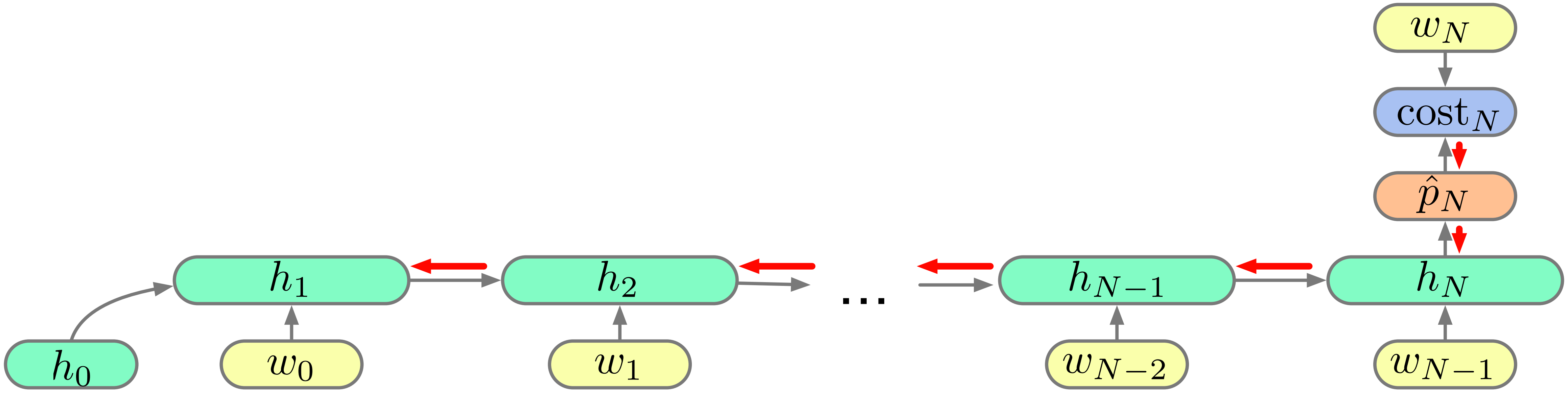
[Slide: Phil Blunsom]

Capturing Long Range Dependencies

If an RNN Language Model is to outperform an n-gram model it must discover and represent long range dependencies:

p(sandcastle | Alice went to the beach. There she built a)

While a simple RNN LM can represent such dependencies in theory, can it learn them?

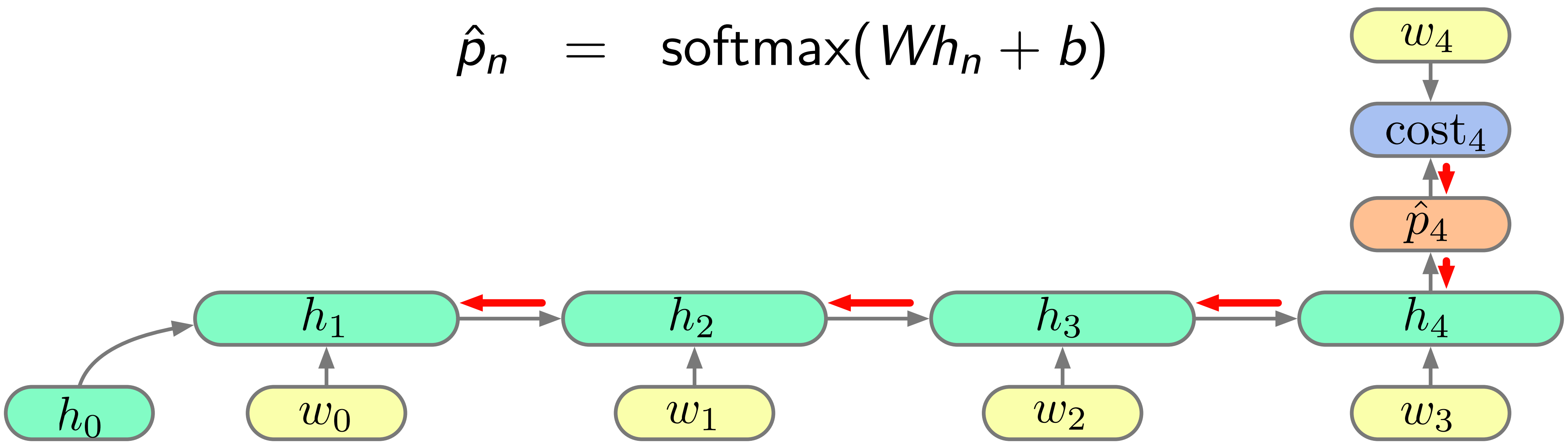


[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_4 to changes in h_1:

$$h_n = g(V[x_n; h_{n-1}] + c)$$
$$\hat{p}_n = \text{softmax}(Wh_n + b)$$



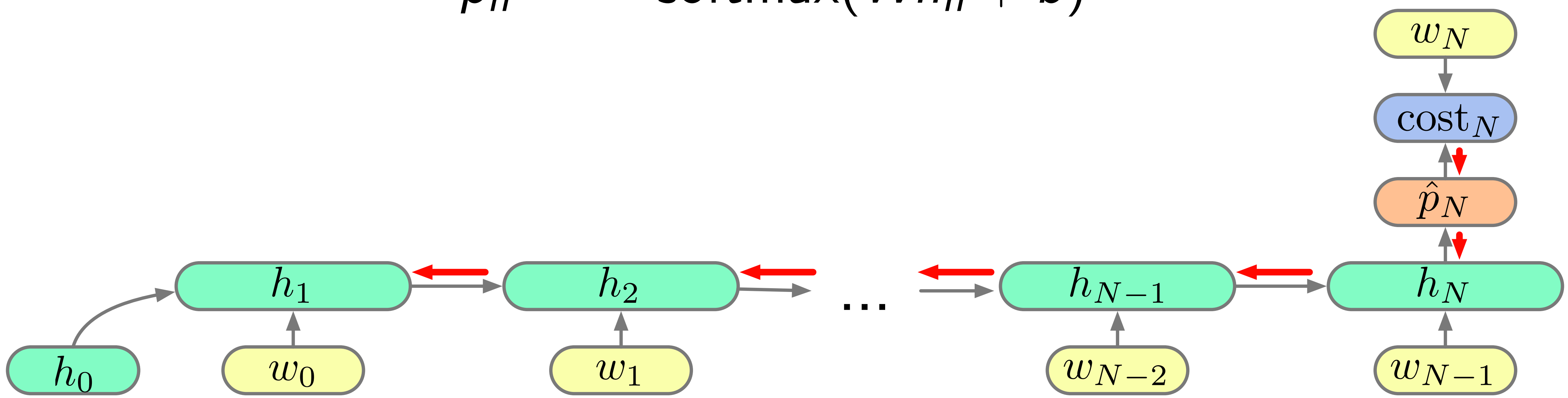
$$\frac{\partial \text{cost}_4}{\partial h_1} = \frac{\partial \text{cost}_4}{\partial \hat{p}_4} \frac{\partial \hat{p}_4}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1}$$

[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_N to changes in h_1:

$$
\begin{aligned}
h_n &= g(V[x_n; h_{n-1}] + c) \\
\hat{p}_n &= \text{softmax}(Wh_n + b)
\end{aligned}
$$



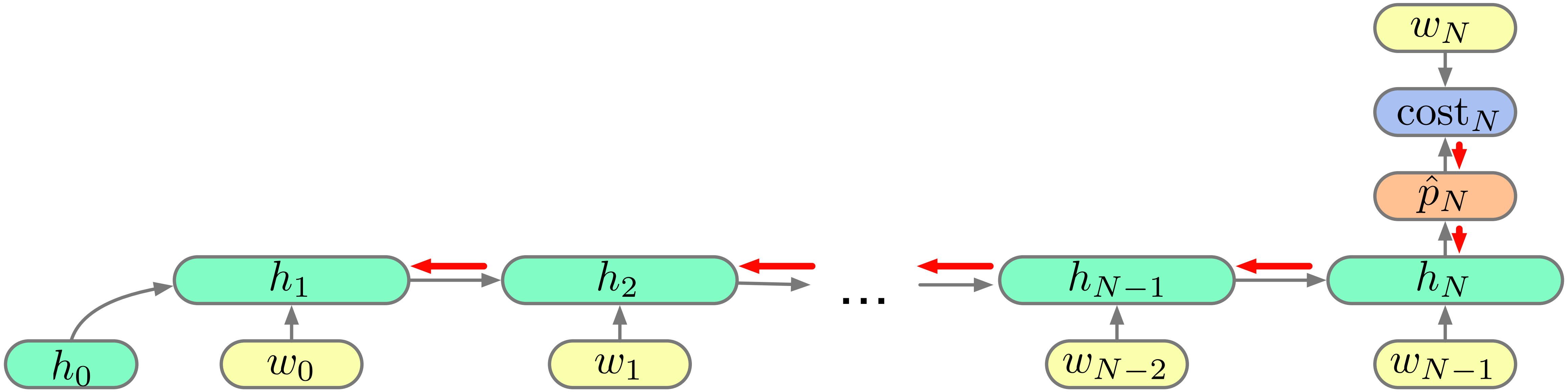
$$
\frac{\partial \text{cost}_N}{\partial h_1} = \frac{\partial \text{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N, \dots, 2\}} \frac{\partial h_n}{\partial h_{n-1}} \right)
$$

[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_N to changes in h_1:

$$h_n = g(V[x_n; h_{n-1}] + c), \qquad \frac{\partial \text{cost}_N}{\partial h_1} = \frac{\partial \text{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N,\ldots,2\}} \frac{\partial h_n}{\partial h_{n-1}} \right)$$

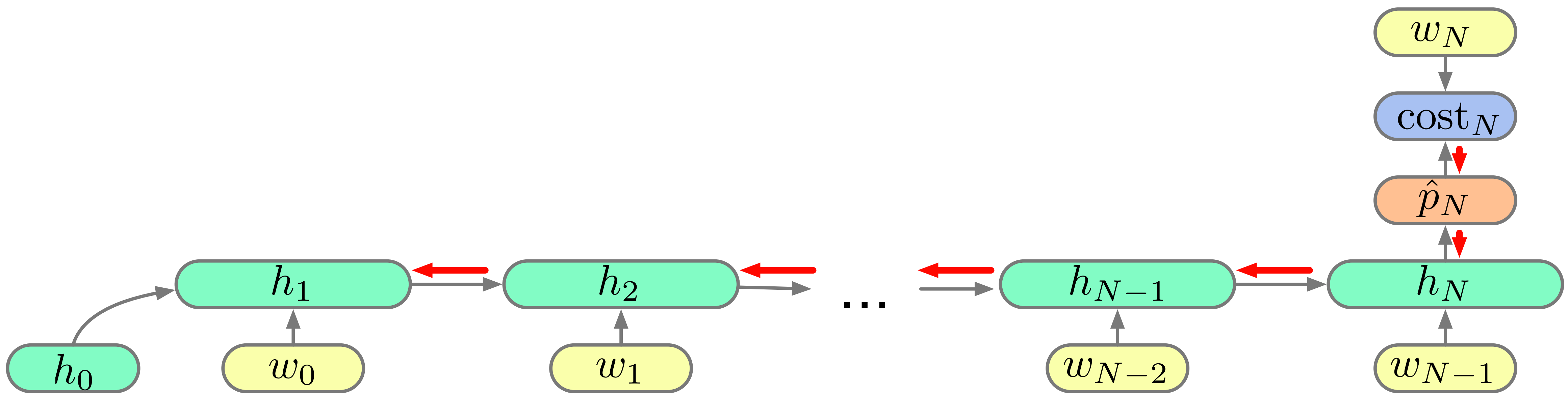


[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_N to changes in h_1:

$$h_n = g(\underbrace{V_x x_n + V_h h_{n-1} + c}_{z_n}), \quad \frac{\partial \mathrm{cost}_N}{\partial h_1} = \frac{\partial \mathrm{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N,\dots,2\}} \frac{\partial h_n}{\partial z_n} \frac{\partial z_n}{\partial h_{n-1}} \right)$$



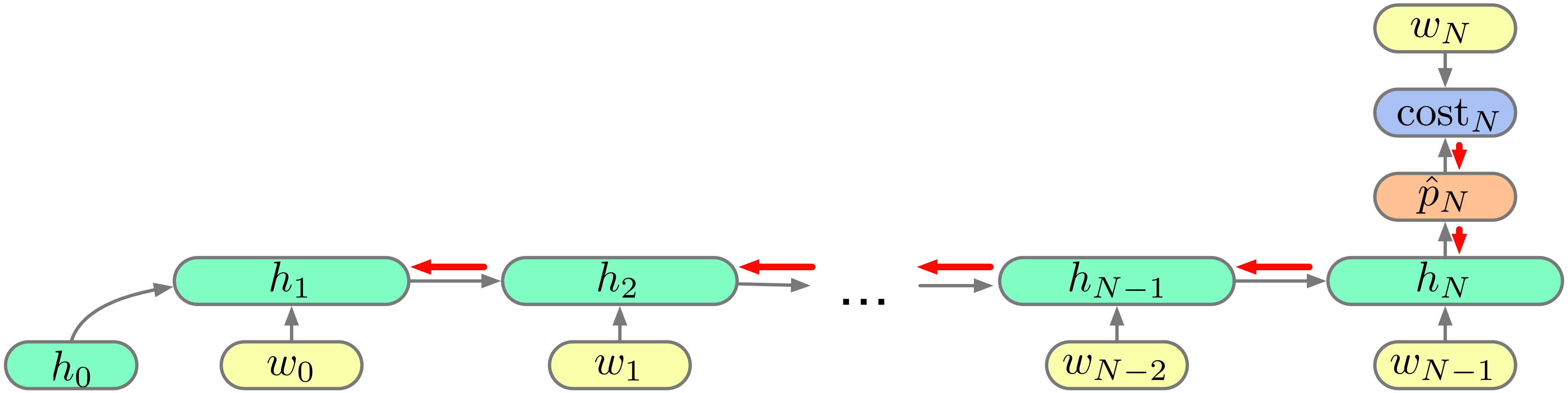
[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_N to changes in h_1:

$$h_n = g(\underbrace{V_x x_n + V_h h_{n-1} + c}_{z_n}), \quad \frac{\partial \text{cost}_N}{\partial h_1} = \frac{\partial \text{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N,\ldots,2\}} \frac{\partial h_n}{\partial z_n} \frac{\partial z_n}{\partial h_{n-1}} \right)$$

$$\frac{\partial h_n}{\partial z_n} = \text{diag}\left(g'(z_n)\right) \qquad \frac{\partial z_n}{\partial h_{n-1}} = V_h$$



[Slide: Phil Blunsom]

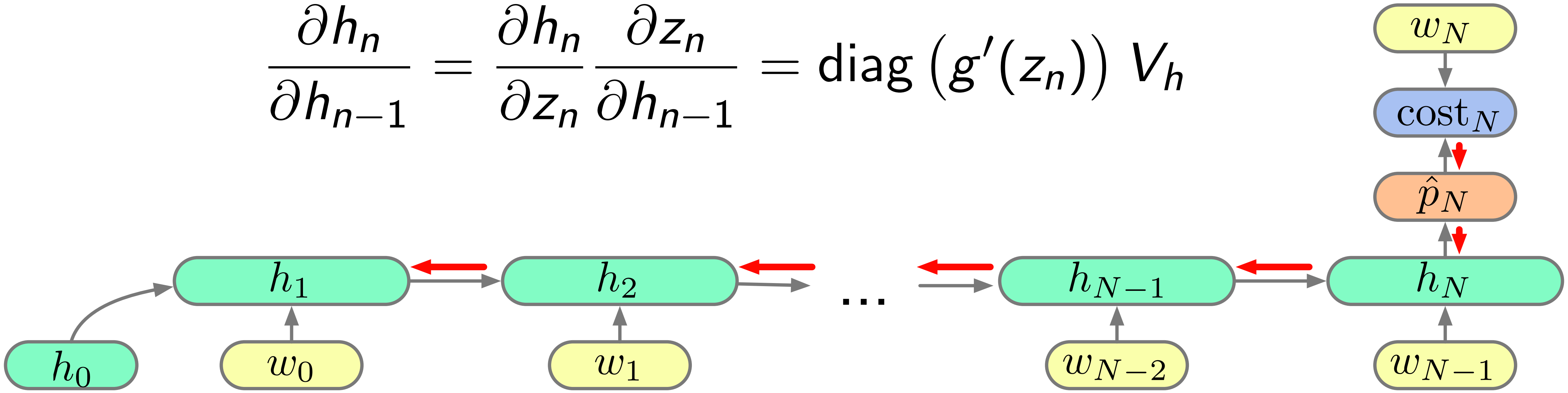
RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_N to changes in h_1:

$$h_n = g(\underbrace{V_x x_n + V_h h_{n-1} + c}_{z_n}), \quad \frac{\partial \text{cost}_N}{\partial h_1} = \frac{\partial \text{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N,\dots,2\}} \frac{\partial h_n}{\partial z_n} \frac{\partial z_n}{\partial h_{n-1}} \right)$$

$$\frac{\partial h_n}{\partial z_n} = \text{diag}\left(g'(z_n)\right) \qquad \frac{\partial z_n}{\partial h_{n-1}} = V_h$$

$$\frac{\partial h_n}{\partial h_{n-1}} = \frac{\partial h_n}{\partial z_n} \frac{\partial z_n}{\partial h_{n-1}} = \text{diag}\left(g'(z_n)\right) V_h$$



[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

$$\frac{\partial \mathrm{cost}_N}{\partial h_1} = \frac{\partial \mathrm{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N, \ldots, 2\}} \mathrm{diag}\left(g'(z_n)\right) V_h \right)$$

The core of the recurrent product is the repeated multiplication of V_h. If the largest eigenvalue of V_h is:

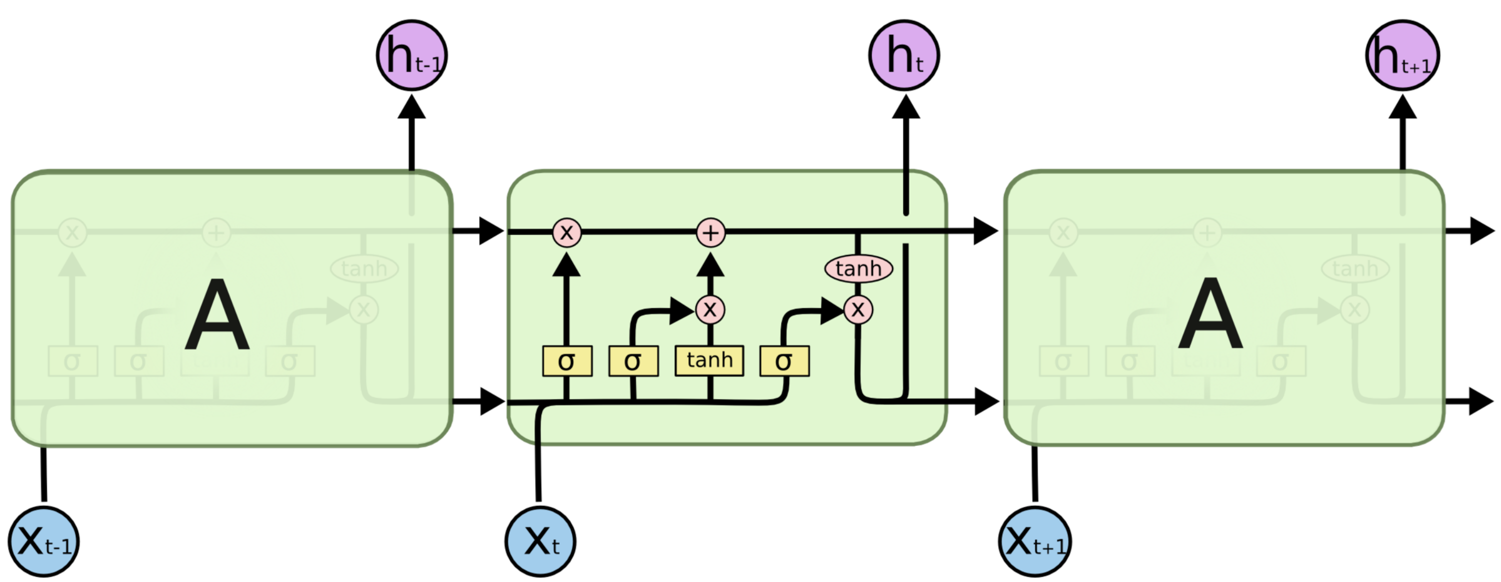
- 1, then gradient will propagate,
- >1, the product will grow exponentially (explode),
- <1, the product shrinks exponentially (vanishes).

[Slide: Phil Blunsom]

LSTM (Long short-term memory)

- Goals:
  - 1. Be able to "remember" for longer distances
  - 2. Stable backpropagation during training
- Augment individual timesteps with a number of specialized vectors and gating functions *(Simpler alternative: GRU. But LSTM is most standard.)*

- Main state
  - **c**: Memory cell
  - **h**: Hidden state
- Update system
  - **g**: proposed new values
  - **f, i, o**: Forget, Input, Output gates control acceptance of **g** into new state



$$\mathbf{c_j} = \mathbf{c_{j-1}} \odot \mathbf{f} + \mathbf{g} \odot \mathbf{i}$$
$$\mathbf{h_j} = \tanh(\mathbf{c_j}) \odot \mathbf{o}$$
$$\mathbf{i} = \sigma(\mathbf{x_j W^{xi}} + \mathbf{h_{j-1} W^{hi}})$$
$$\mathbf{f} = \sigma(\mathbf{x_j W^{xf}} + \mathbf{h_{j-1} W^{hf}})$$
$$\mathbf{o} = \sigma(\mathbf{x_j W^{xo}} + \mathbf{h_{j-1} W^{ho}})$$
$$\mathbf{g} = \tanh(\mathbf{x_j W^{xg}} + \mathbf{h_{j-1} W^{hg}})$$

Christopher Olah: Understanding LSTM Networks
`colah.github.io/posts/2015-08-Understanding-LSTMs/`

memory component ("cell")

c_{j-1}

c_j

h_{j-1}

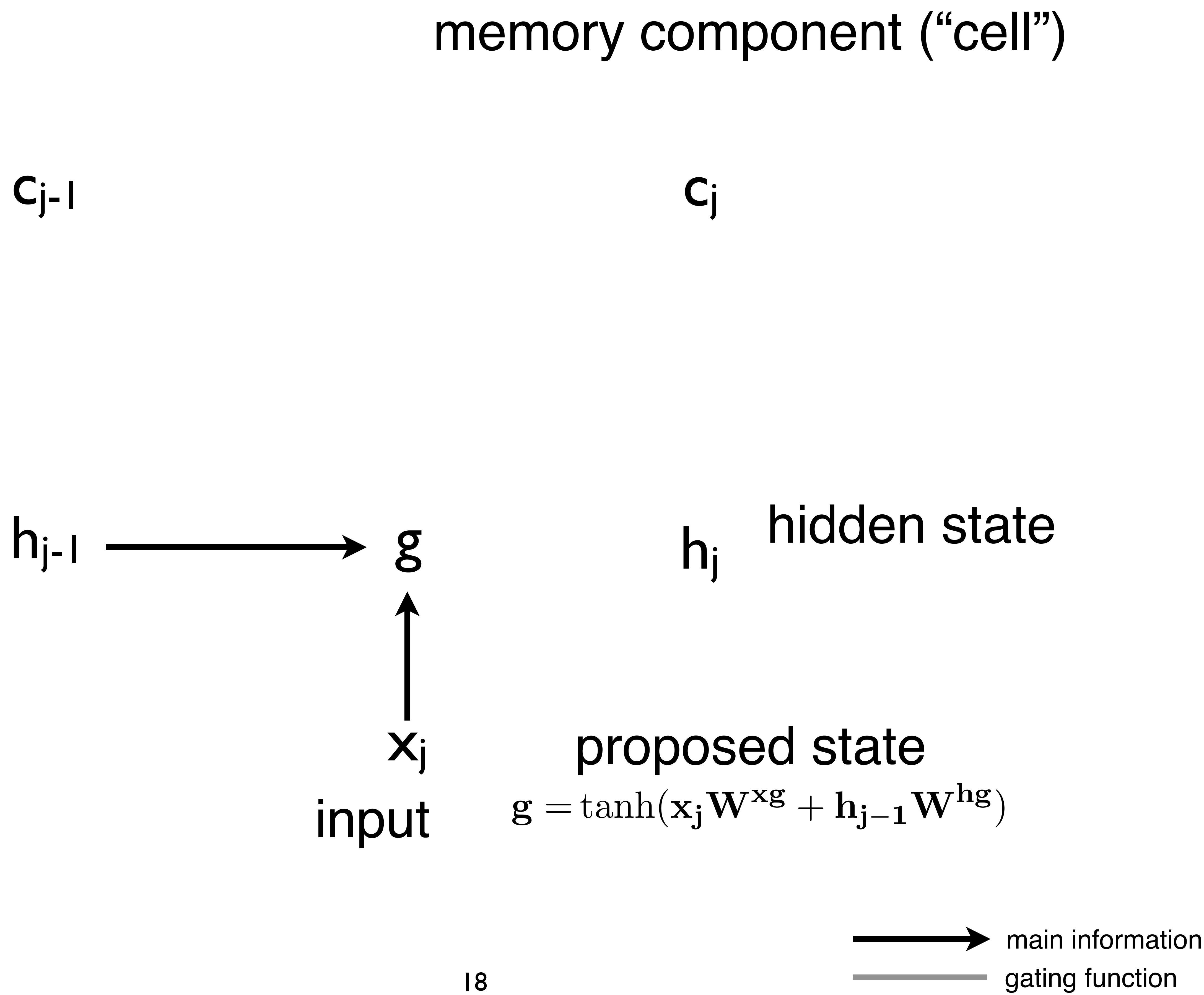
h_j hidden state

x_j

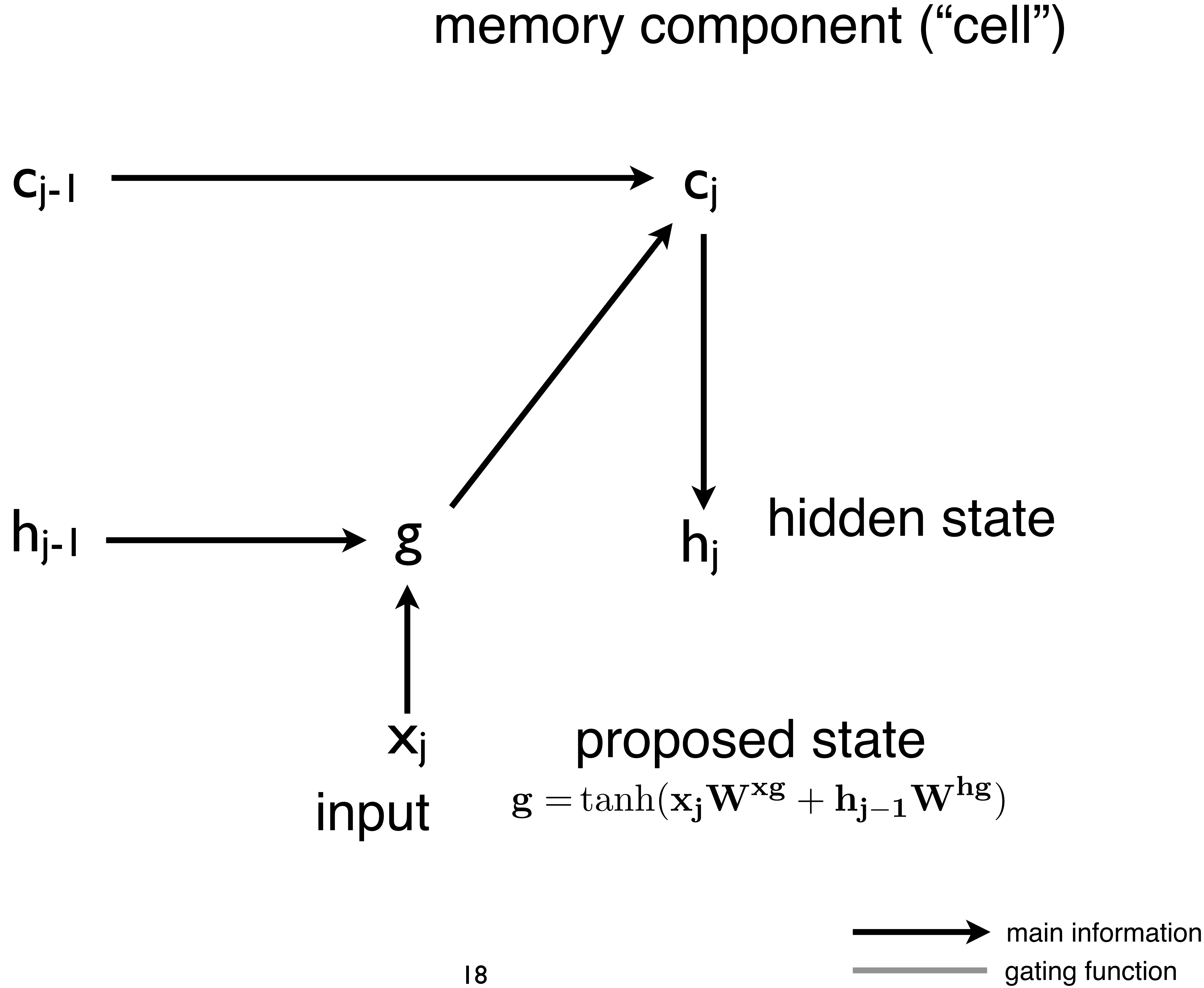
input

main information

gating function

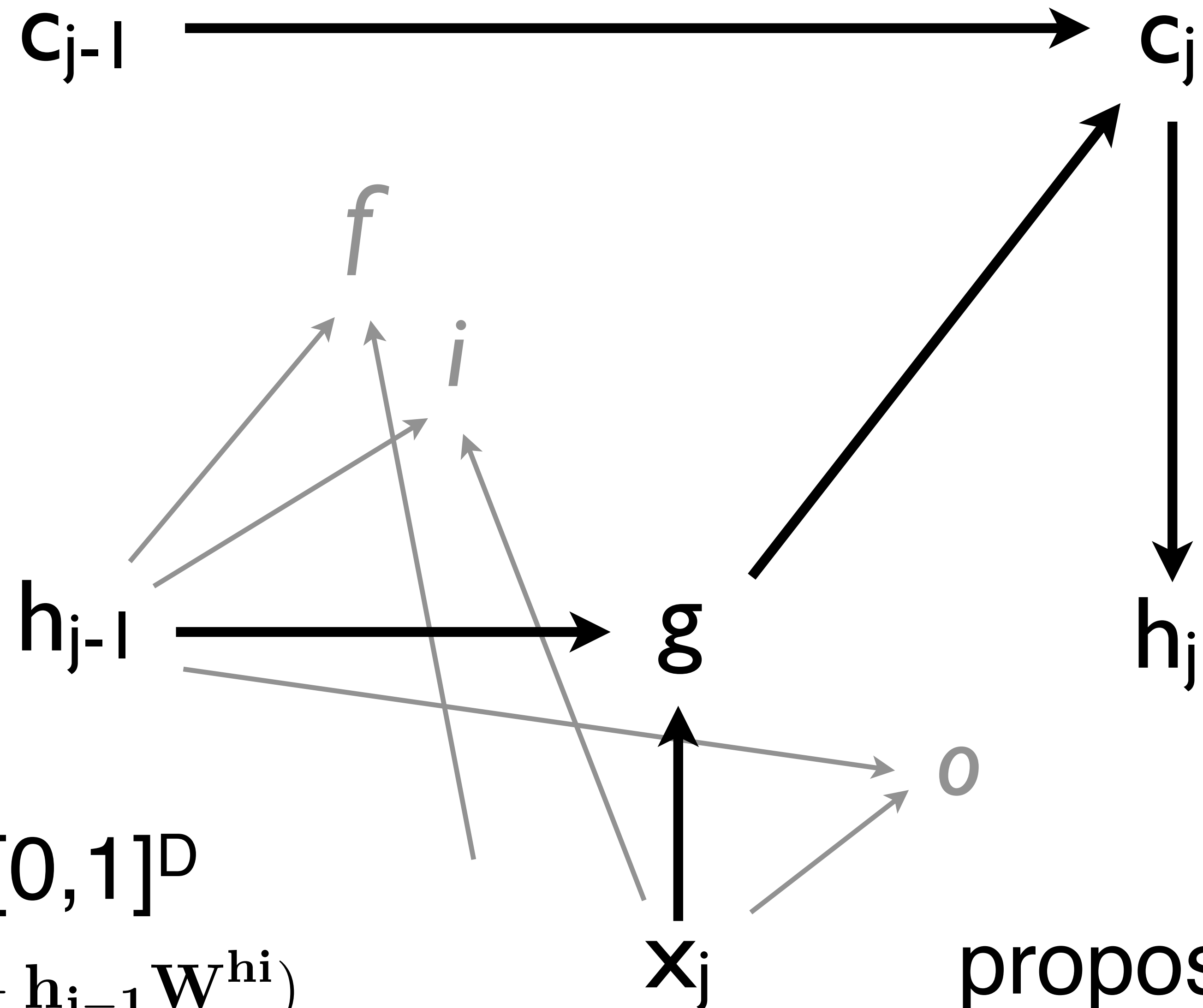


$$\mathbf{g} = \tanh(\mathbf{x_j W^{xg}} + \mathbf{h_{j-1} W^{hg}})$$



$$\mathbf{g} = \tanh(\mathbf{x_j} \mathbf{W^{xg}} + \mathbf{h_{j-1}} \mathbf{W^{hg}})$$

memory component ("cell")



hidden state

gates $\in [0,1]^D$

$$\mathbf{i} = \sigma(\mathbf{x_j W^{xi}} + \mathbf{h_{j-1} W^{hi}})$$

$$\mathbf{f} = \sigma(\mathbf{x_j W^{xf}} + \mathbf{h_{j-1} W^{hf}})$$

$$\mathbf{o} = \sigma(\mathbf{x_j W^{xo}} + \mathbf{h_{j-1} W^{ho}})$$

input

proposed state

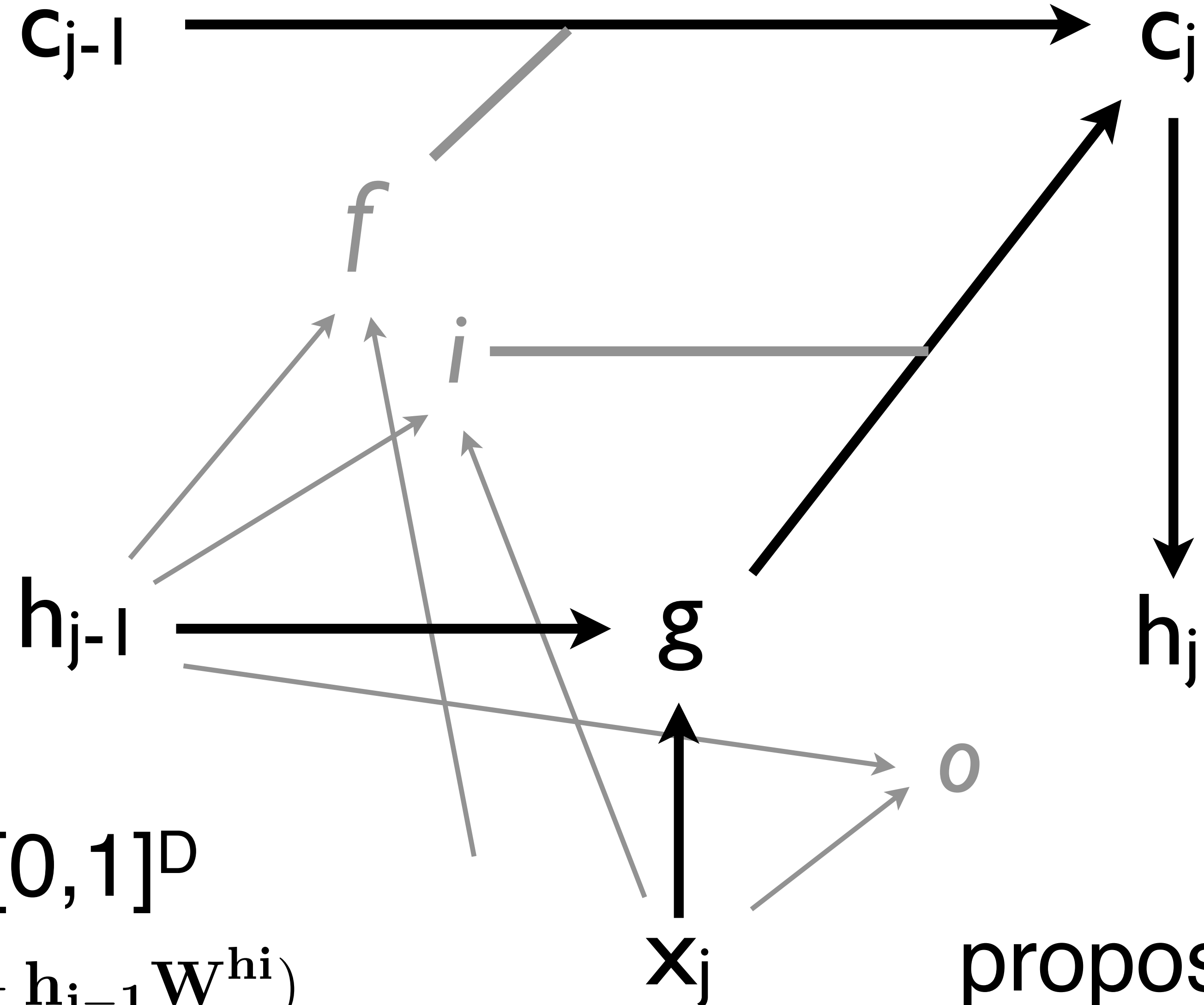
$$\mathbf{g} = \tanh(\mathbf{x_j W^{xg}} + \mathbf{h_{j-1} W^{hg}})$$

main information

gating function

memory component ("cell")

$$\mathbf{c_j} = \mathbf{c_{j-1}} \odot \mathbf{f} + \mathbf{g} \odot \mathbf{i}$$



hidden state

gates ∈ [0,1]^D

$$\mathbf{i} = \sigma(\mathbf{x_j W^{xi}} + \mathbf{h_{j-1} W^{hi}})$$

$$\mathbf{f} = \sigma(\mathbf{x_j W^{xf}} + \mathbf{h_{j-1} W^{hf}})$$

$$\mathbf{o} = \sigma(\mathbf{x_j W^{xo}} + \mathbf{h_{j-1} W^{ho}})$$

input

proposed state

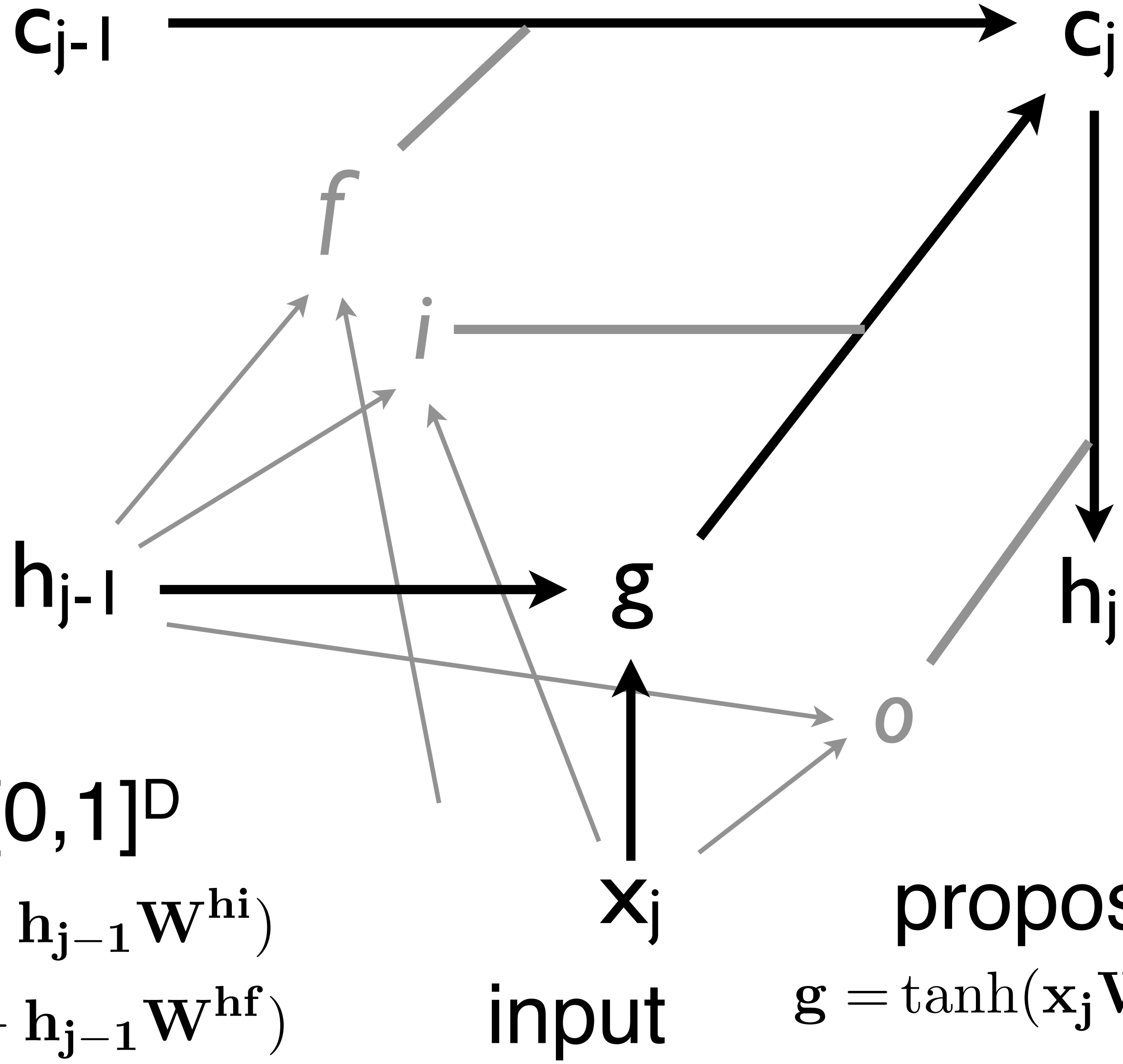
$$\mathbf{g} = \tanh(\mathbf{x_j W^{xg}} + \mathbf{h_{j-1} W^{hg}})$$

main information

gating function

memory component ("cell")

$$\mathbf{c_j} = \mathbf{c_{j-1}} \odot \mathbf{f} + \mathbf{g} \odot \mathbf{i}$$



hidden state

$$\mathbf{h_j} = \tanh(\mathbf{c_j}) \odot \mathbf{o}$$

gates $\in [0,1]^D$

$$\mathbf{i} = \sigma(\mathbf{x_j} \mathbf{W^{xi}} + \mathbf{h_{j-1}} \mathbf{W^{hi}})$$

$$\mathbf{f} = \sigma(\mathbf{x_j} \mathbf{W^{xf}} + \mathbf{h_{j-1}} \mathbf{W^{hf}})$$

$$\mathbf{o} = \sigma(\mathbf{x_j} \mathbf{W^{xo}} + \mathbf{h_{j-1}} \mathbf{W^{ho}})$$

proposed state

$$\mathbf{g} = \tanh(\mathbf{x_j} \mathbf{W^{xg}} + \mathbf{h_{j-1}} \mathbf{W^{hg}})$$

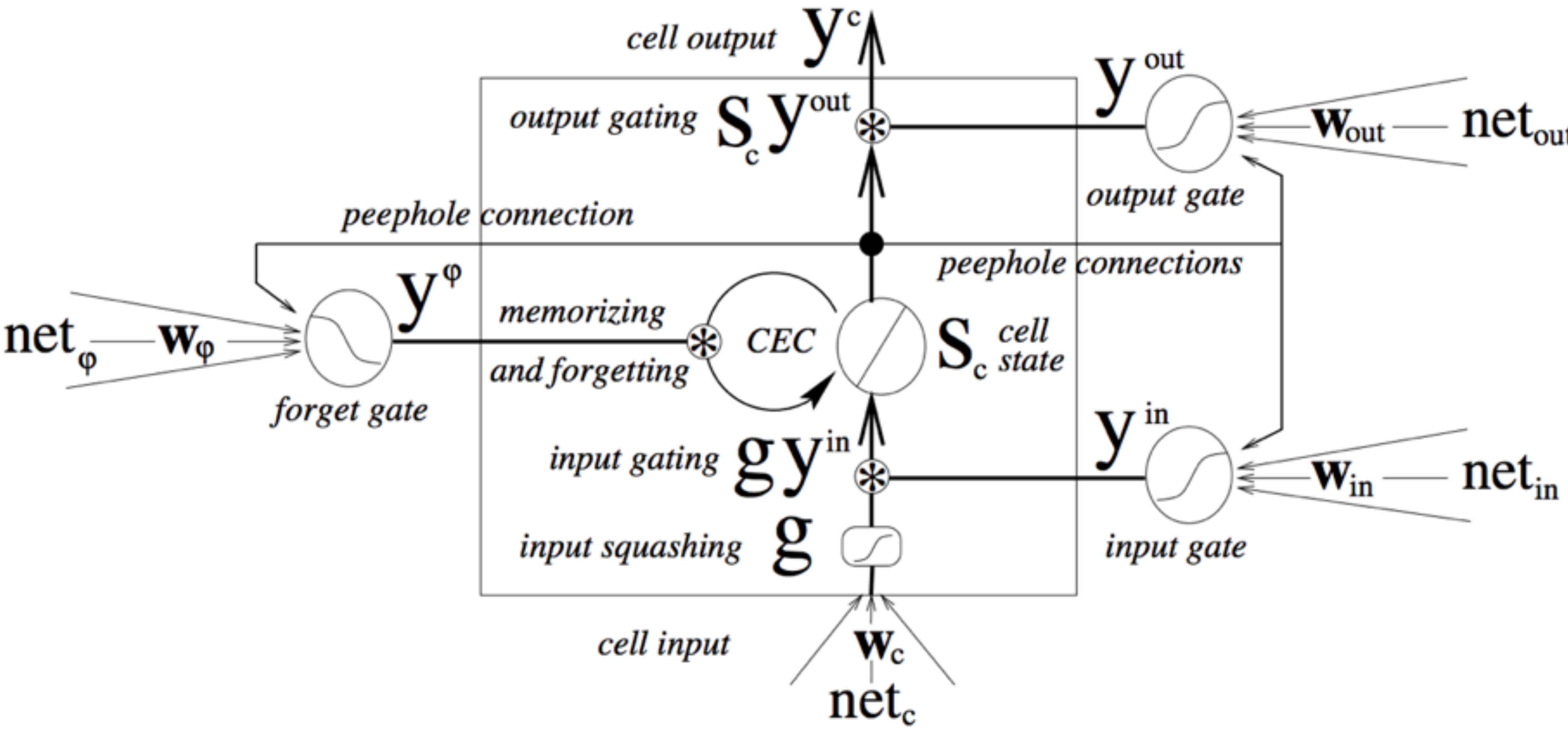


Figure 1: LSTM memory block with one cell.

- Note many LSTM variants (peephole or not; ci+use (1-f) or not...)
  [diagram: Gers and Schmidhuber 2001]
- LSTMs have a poor reputation for understandability... yet do something right... usually just used as a black-box

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.
```

```
First, the devishin it son?

MONTANO:
'Tis true as full Squellen the rest me, my passacre. and nothink
my fairs,' done to vision of actious to thy to love, brings gods!

THUR:
Will comfited our flight offend make thy love;
Brothere is oats at on thes:'--why, cross and so
her shouldestruck at one their hearina in all go to lives of
Costag,
To his he tyrant of you our the fill we hath trouble an over me?

KING JOHN:
Great though I gain; for talk to mine and to the Christ: a right
him out
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/
http://nbviewer.jupyter.org/gist/yoavg/d76121dfde2618422139

Structure awareness

Cell sensitive to position in line:

```
The sole importance of the crossing of the Berezina lies in the fact
that it plainly and indubitably proved the fallacy of all the plans for
cutting off the enemy's retreat and the soundness of the only possible
line of action--the one Kutuzov and the general mass of the army
demanded--namely, simply to follow the enemy up. The French crowd fled
at a continually increasing speed and all its energy was directed to
reaching its goal. It fled like a wounded animal and it was impossible
to block its path. This was shown not so much by the arrangements it
made for crossing as by what took place at the bridges. When the bridges
broke down, unarmed soldiers, people from Moscow and women with children
who were with the French transport, all--carried on by vis inertiae--
pressed forward into boats and into the ice-covered water and did not,
surrender.
```

Cell that turns on inside quotes:

```
"You mean to imply that I have nothing to eat out of.... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."
```

Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
   siginfo_t *info)
{
 int sig = next_signal(pending, mask);
 if (sig) {
  if (current->notifier) {
   if (sigismember(current->notifier_mask, sig)) {
    if (!(current->notifier)(current->notifier_data)) {
     clear_thread_flag(TIF_SIGPENDING);
     return 0;
    }
   }
  }
  collect_signal(sig, pending, info);
 }
 return sig;
}
```

A large portion of cells are not easily interpretable. Here is a typical example:

```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
 char *str;
 if (!*bufp || (len == 0) || (len > *remain))
  return ERR_PTR(-EINVAL);
 /* Of the currently implemented string fields, PATH_MAX
  * defines the longest valid length.
  */
```

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

- LSTMs used as a generic, sequence-aware model within language modeling, translation generation, classification and tagging

- Various LSTM-analyzing-text visualizations
  - http://karpathy.github.io/2015/05/21/rnn-effectiveness/
  - http://lstm.seas.harvard.edu/

- Question: can they learn interactions we *know* are in natural language?
  - Thursday: Linzen et al.!