# Overview of Natural Language Processing
# Part II & ACS L90

## Lecture 8: Recurrent Neural Networks

Guy Emerson
Based on slides by Weiwei Sun

Department of Computer Science and Technology
University of Cambridge

Michaelmas 2021/22

(1) a. I convinced her children are noisy.

b. The old man the boats.

c. The florist sent the flowers was pleased.

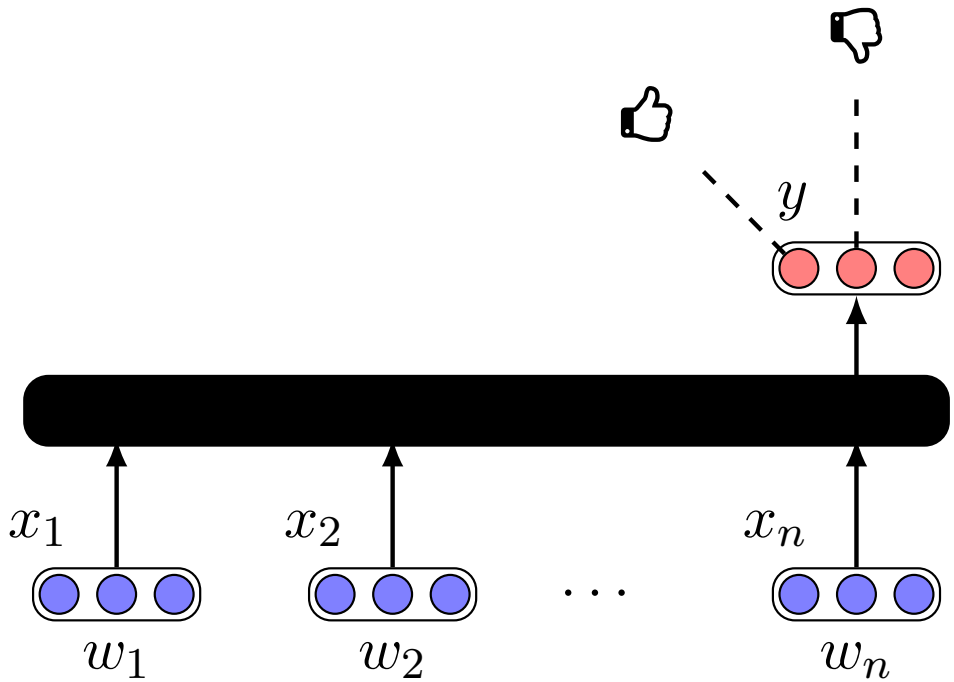
Language is an inherently temporal phenomenon

## Lecture 8: Recurrent Neural Networks

1. Modelling sequences
2. Recurrent Neural Networks
3. Neural Language Models

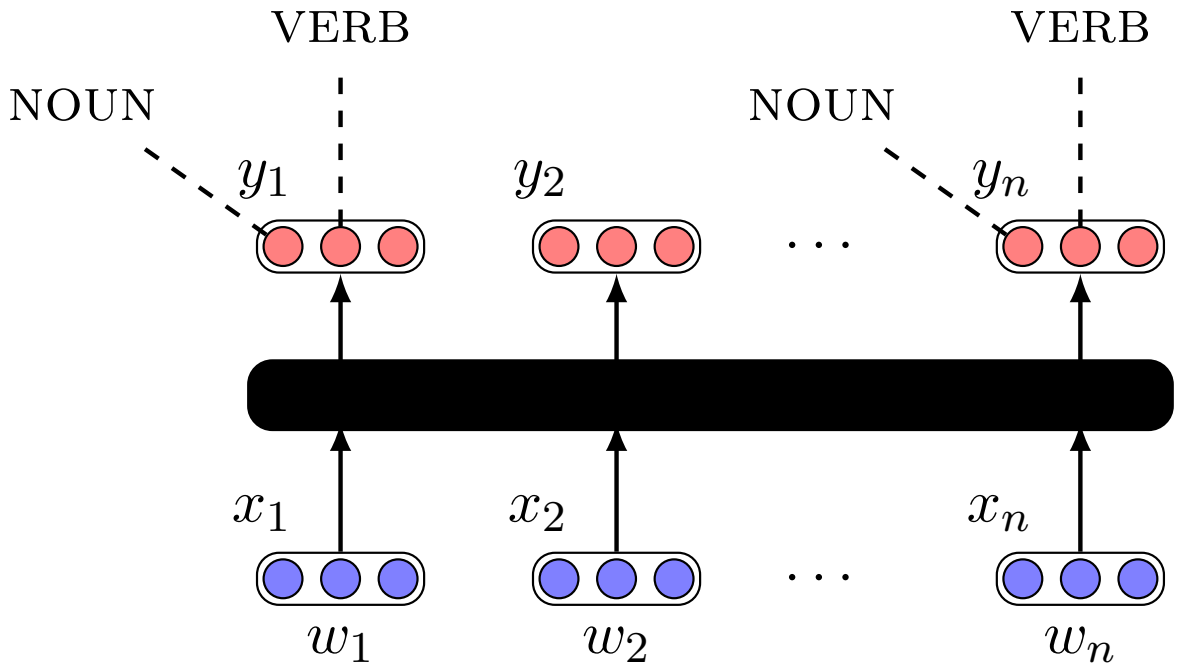
# Modelling Sequences

# Many input tokens; one output token



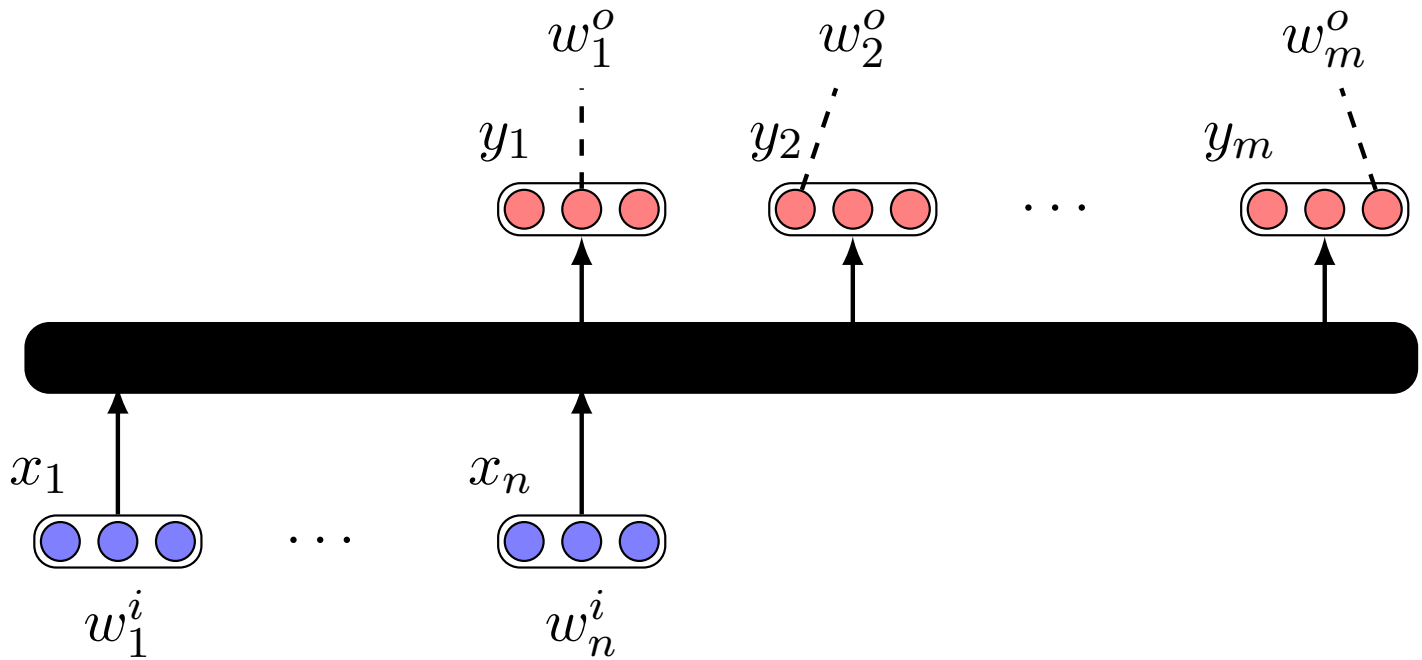

- Sentiment classification
- Document classification
- Automatic essay scoring
- ...

# Many input tokens; many output tokens (labelling)



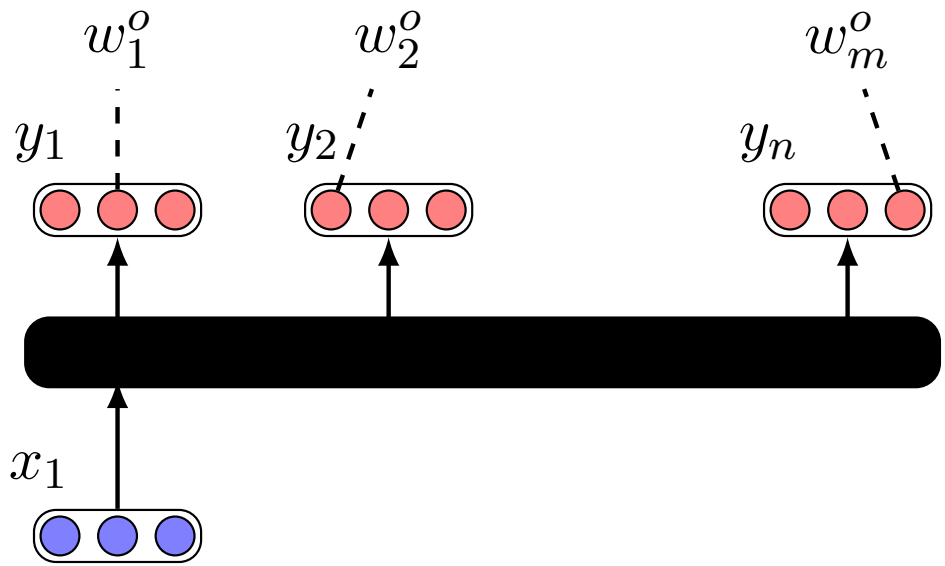

- POS tagging
- Segmentation and chunking
- Information extraction
- Dependency parsing
- ...

# Many input tokens; many output tokens (string-to-string)



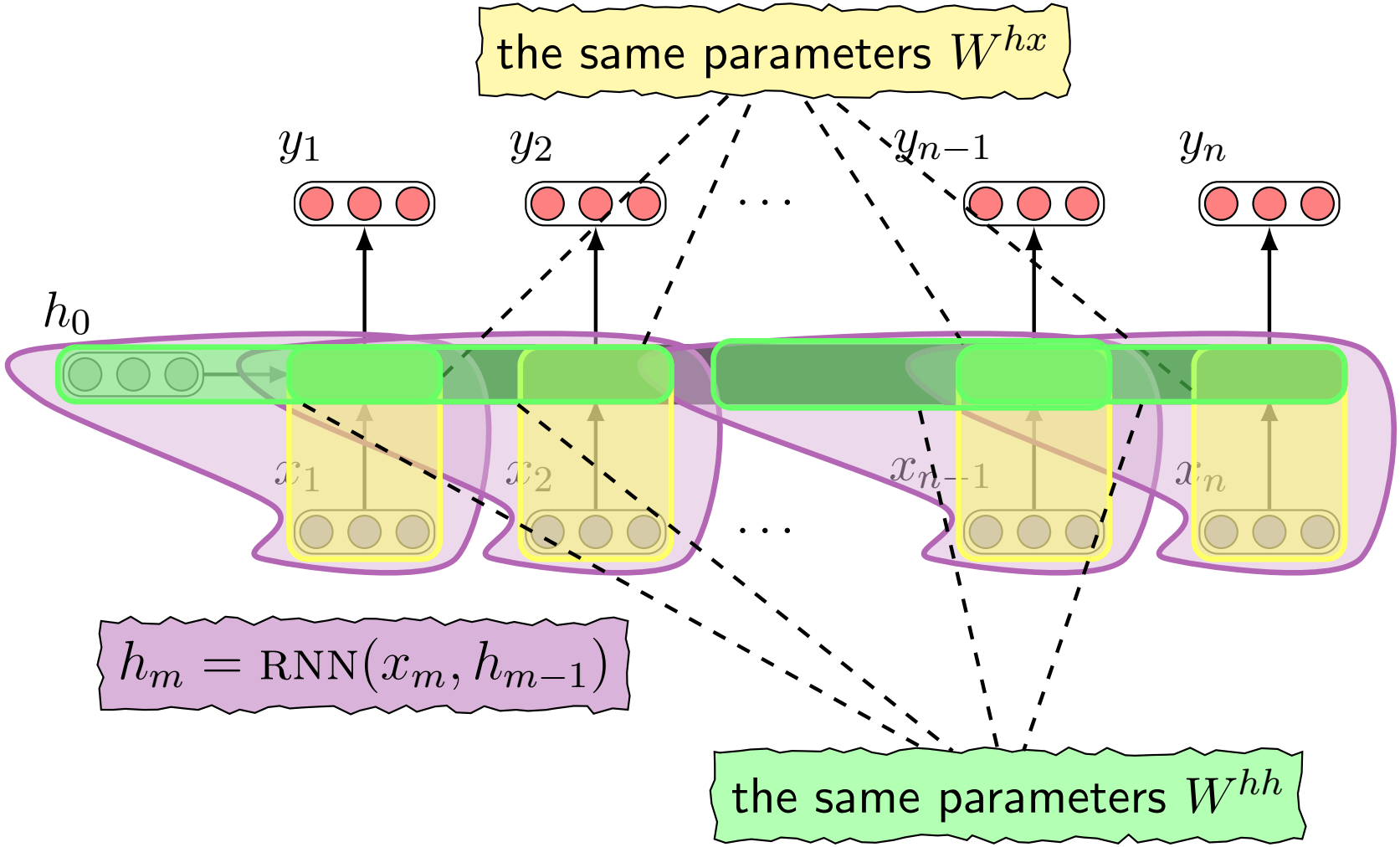

- Machine translation
- Text summarization
- ChatBot?
- ...

# One input token; many output tokens




- Image captioning
- ...

# Recurrent Neural Networks

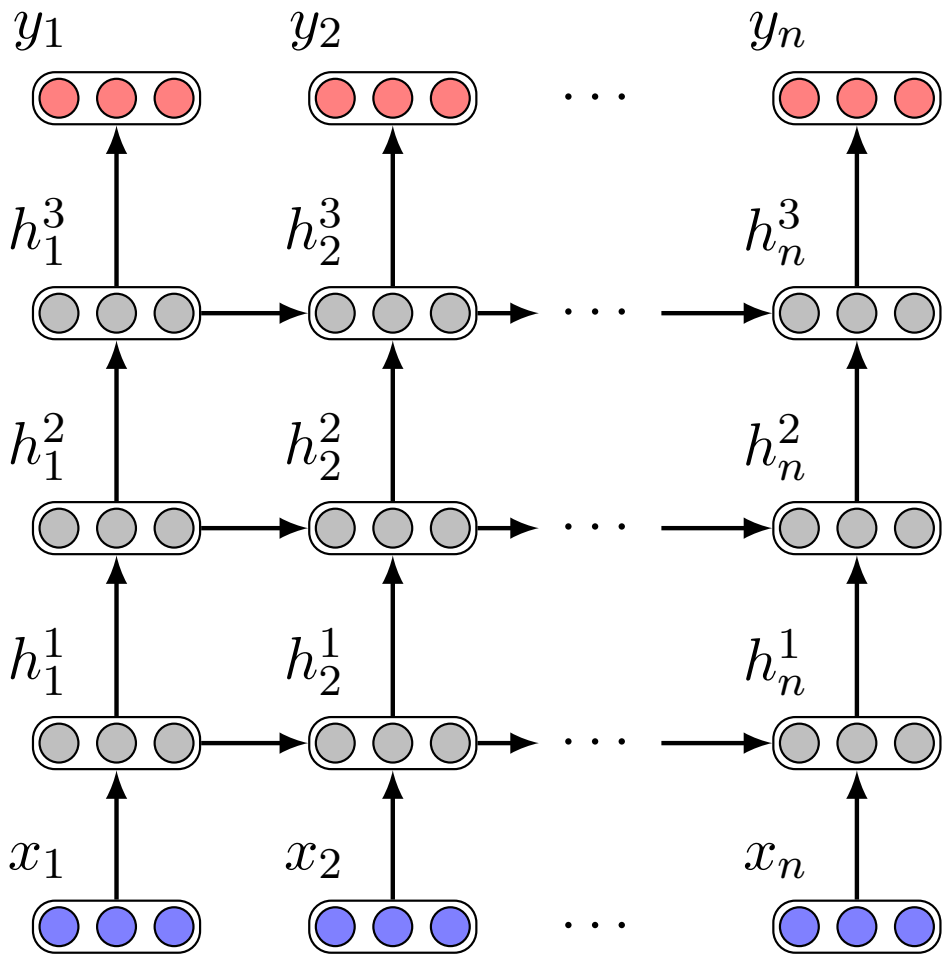



The context at token $m$ is summarized by a recurrently-updated vector:

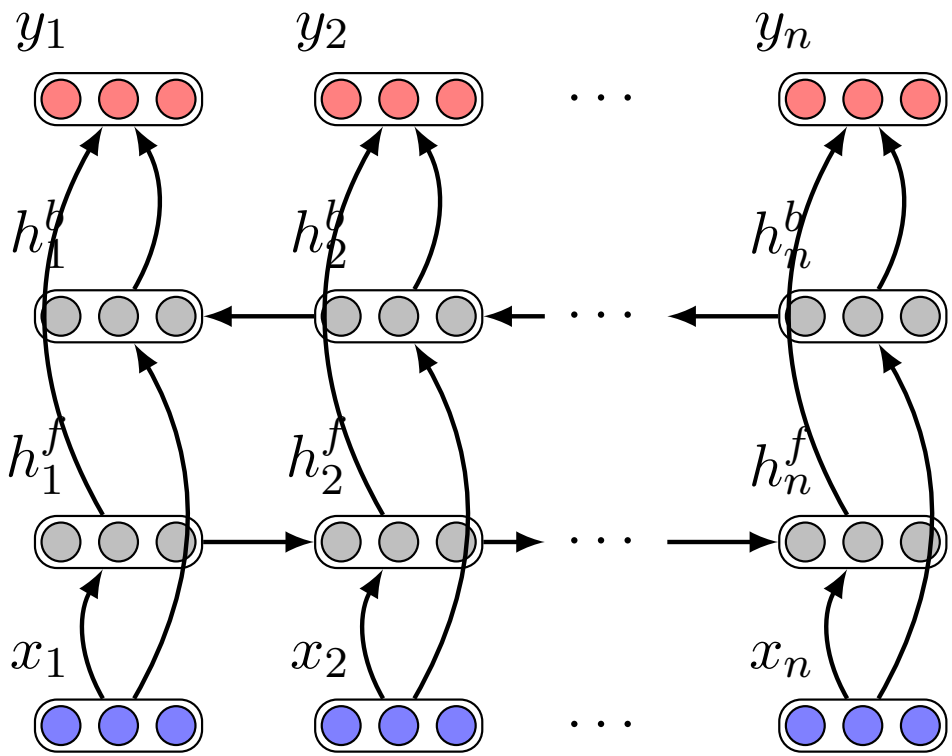
$$h_m = g(W^{hx}x_m + W^{hh}h_{m-1} + b^h)$$

A straightforward application to sequence labelling is to score each tag $y_m$ with a log-linear function: $\texttt{softmax}(W^{yh}h_i + b^y)$

# Multiple hidden layers



# Bidirectional RNN




- $h_i^f = \text{RNN}_{\text{forward}}(x, i)$
- $h_i^b = \text{RNN}_{\text{backward}}(x, i)$
- $h_i = h_i^f \oplus h_i^b$

# Difficulties

- Despite having access to the entire preceding sequence, the information encoded in hidden states tends to be fairly local, more relevant to the most recent parts of the input sequence and recent decisions.
- It is often the case, however, that distant information is critical to many language applications.

*The flights the airline was cancelling were full.*

- A second difficulty with training simple RNNs arises from the need to backpropagate the error signal back through time.
- A frequent result of this process is that the gradients are eventually driven to zero — the so-called vanishing gradients problem.

How can the two problems be solved with syntax?

# Long Short-Term Memory

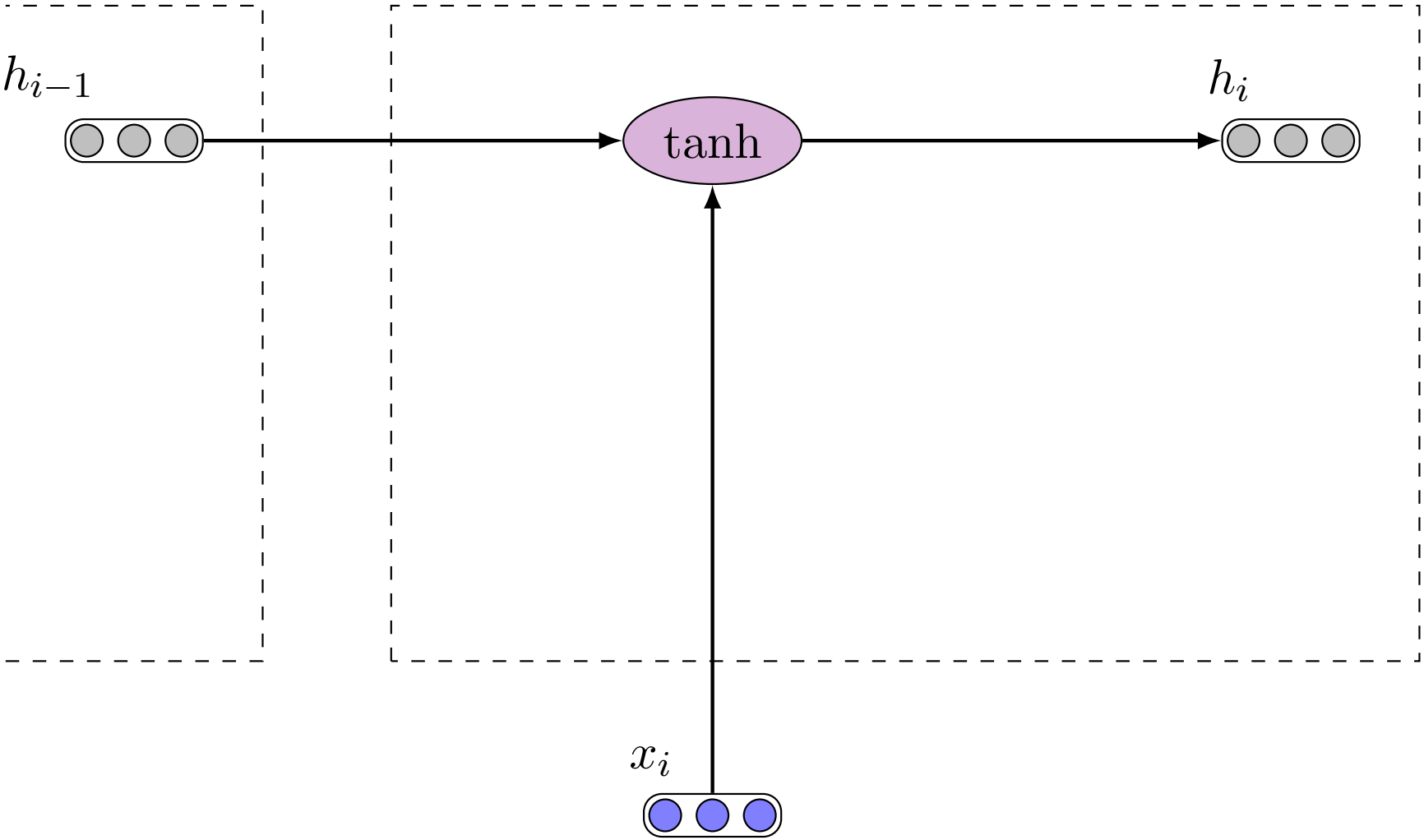
- RNNs have short-term memory
- Aim: lengthen the short-term memory

*Long short-term memory (LSTM) networks*

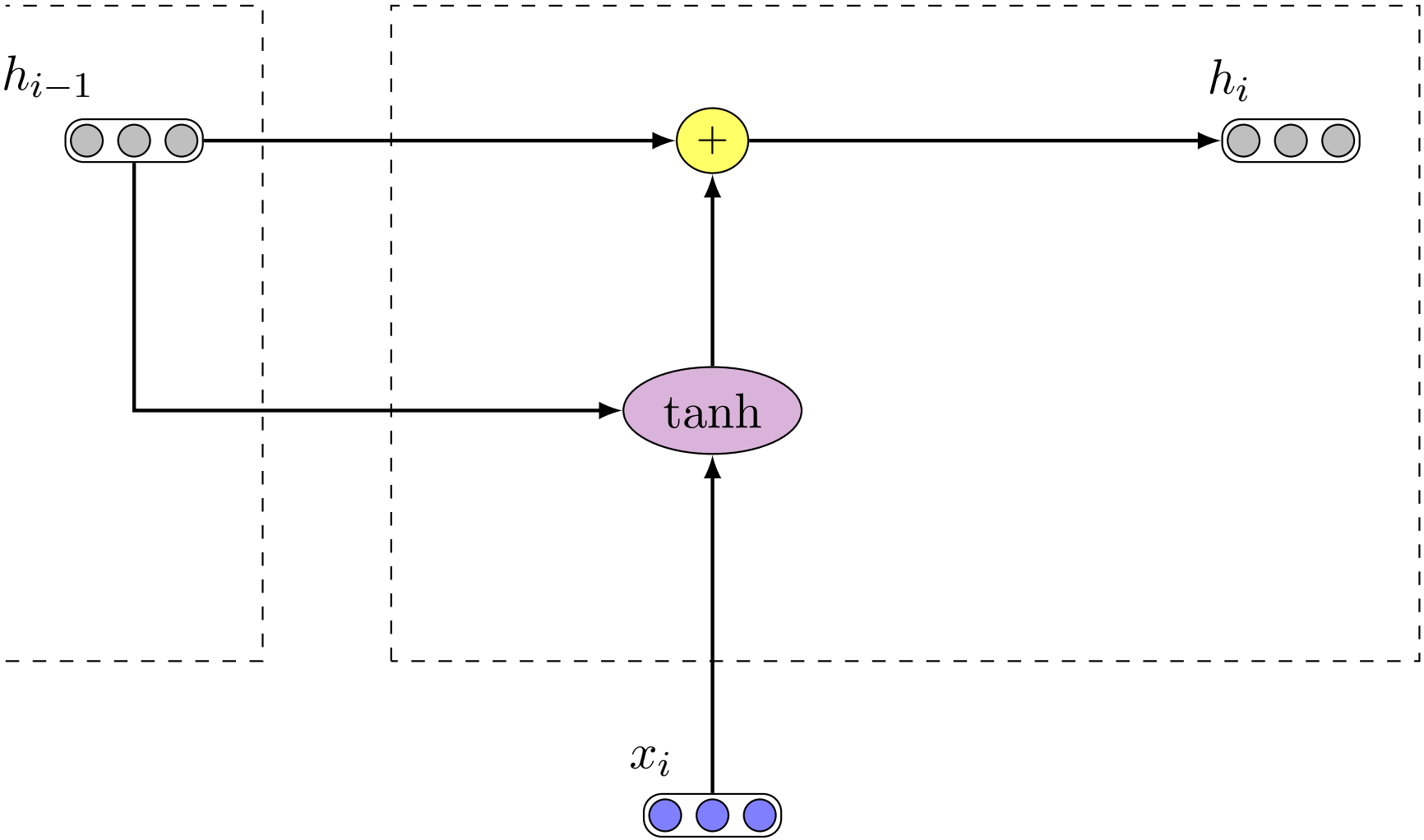
- remove information no longer needed from the context
- add information likely to be needed for later decision making

The *key* is to learn how to manage the context rather than hard-coding a strategy: *adding gates* to control the flow of information into and out of the units.

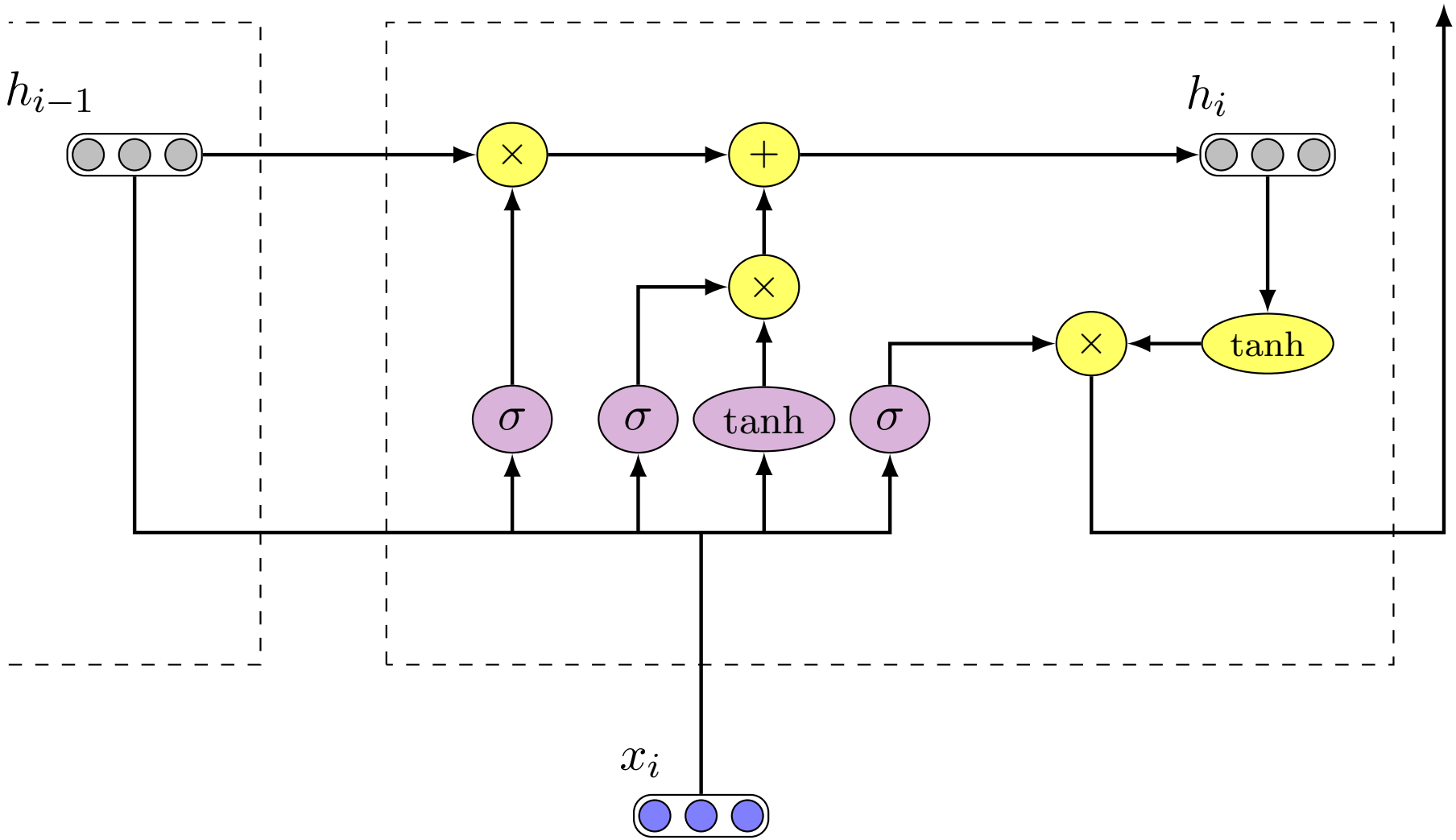
# Long Short-Term Memory



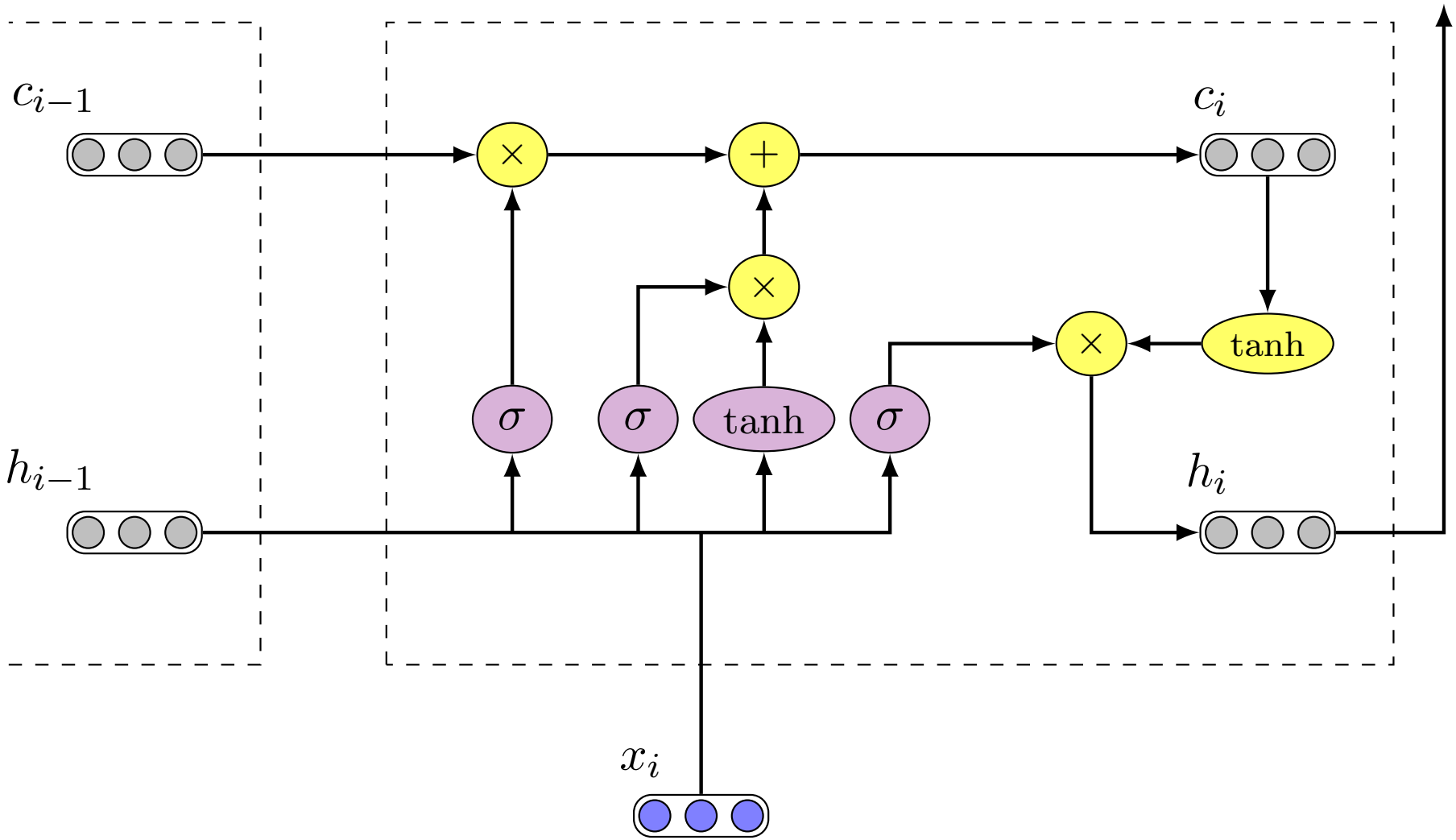
# Long Short-Term Memory



# Long Short-Term Memory



# Long Short-Term Memory



# Long Short-Term Memory

*basic computation*

$$g_i = \tanh(W^{hh} h_{i-1} + W^{hx} x_i)$$

*context vector*

$$c_i = j_i + k_i$$

*input gate*: select the information to add to the current context.

$$\begin{aligned} i_i &= \sigma(W^{i,h} h_{i-1} + W^{i,x} x_i) \\ j_i &= g_i \odot i_i \end{aligned}$$

*forget gate*: delete information from the context ▷cf LEFT-ARC

$$\begin{aligned} f_i &= \sigma(W^{f,h} h_{i-1} + W^{f,x} x_i) \\ k_i &= c_{i-1} \odot f_i \end{aligned}$$

*output gate*: decide the information for the current hidden state.

$$\begin{aligned} o_i &= \sigma(W^{o,h} h_{i-1} + W^{o,x} x_i) \\ h_i &= o_i \odot \tanh(c_i) \end{aligned}$$

# Neural Language Models

# Language Model

Assign a probability to a sentence:

$$p(w_1, w_2, \ldots, w_n)$$

Typically $p$ is decomposed into $p(w_i|\text{context})$ (aka word prediction)

## Why

- unsupervised training for various models (esp. neural networks, e.g. `word2vec`).
- word prediction for communication aids:
  e.g., to help enter text that's input to a synthesiser
  search engine (dynamically maintain a list of upcoming words/phrases),
- natural language generation: machine translation, text summarization, etc.
- spelling correction, speech recognition

# $N$-gram models

$$p(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} p(w_i | w_1, w_2, \ldots, w_{i-1})$$

Too many possible sentences! Not enough data for parameter estimation.
*$N$-gram* models assume some independence ▷ Markov assumption

$$p(w_i | \text{local left context}) = p(w_i | w_{i-l}, w_{i-l+1}, \ldots, w_{i-1})$$

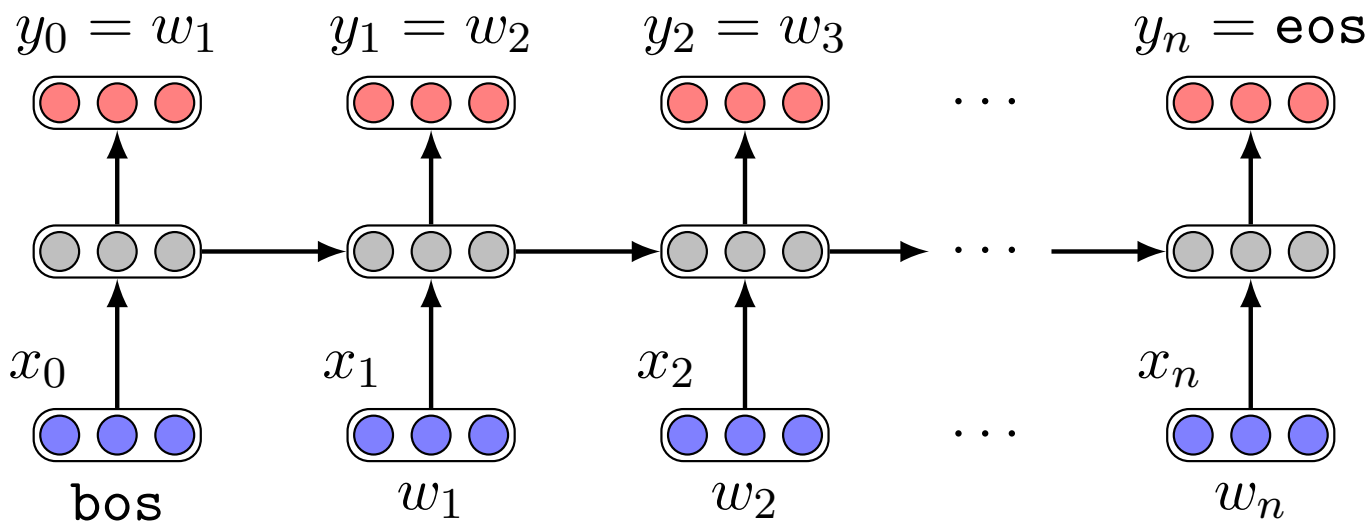
*Bigram*: a probability is assigned to a word based on the previous word

$$p(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} p(w_i | w_{i-1})$$

Probability can be estimated from counts in a training corpus:

$$\frac{\text{COUNT}(w_{i-1} w_i)}{\text{COUNT}(w_i)}$$

# Neural language models




Many variants

# ELMo: Deep Contextualized Word Representations

| Task | Previous SOTA | | Our baseline | ELMo + baseline | Increase (absolute/ relative) |
|---|---|---|---|---|---|
| SQuAD | Liu et al. (2017) | 84.4 | 81.1 | 85.8 | 4.7 / 24.9% |
| SNLI | Chen et al. (2017) | 88.6 | 88.0 | $88.7 \pm 0.17$ | 0.7 / 5.8% |
| SRL | He et al. (2017) | 81.7 | 81.4 | 84.6 | 3.2 / 17.2% |
| Coref | Lee et al. (2017) | 67.2 | 67.2 | 70.4 | 3.2 / 9.8% |
| NER | Peters et al. (2017) | $91.93 \pm 0.19$ | 90.15 | $92.22 \pm 0.10$ | 2.06 / 21% |
| SST-5 | McCann et al. (2017) | 53.7 | 51.4 | $54.7 \pm 0.5$ | 3.3 / 6.8% |

- Peters et al. (2018)
- Bidirectional LSTM language model with 2 layers
- Trained on 1 billion tokens of English
- Model's internal vectors $\rightarrow$ contextualized word representations

# Model "Zoo"

- LSTM variants (e.g. Gated Recurrent Units, "peephole" connections)
- LSTMs with attention (use weighted sums of vectors)
- Highway Networks (gates between layers)
- Residual connections (simple alternative to gates)
- Convolutional Neural Nets (word windows instead of recurrence)
- Transformers (e.g. BERT) (attention instead of recurrence)
- Any sequence model, plus a Conditional Random Field (useful for structured prediction)
- Any sequence model, as part of a probabilistic graphical model (useful for making structural assumptions)
- ...

Remember to think about the data!

Remember to think about methodology!

# Reading

- D Jurafsky and J Martin. *Speech and Language Processing* Chapter 9. Sequence Processing with Recurrent Networks. `web.stanford.edu/~jurafsky/slp3/9.pdf`