Recurrent neural networks

Marco Kuhlmann
Department of Computer and Information Science

LINKÖPING UNIVERSITY

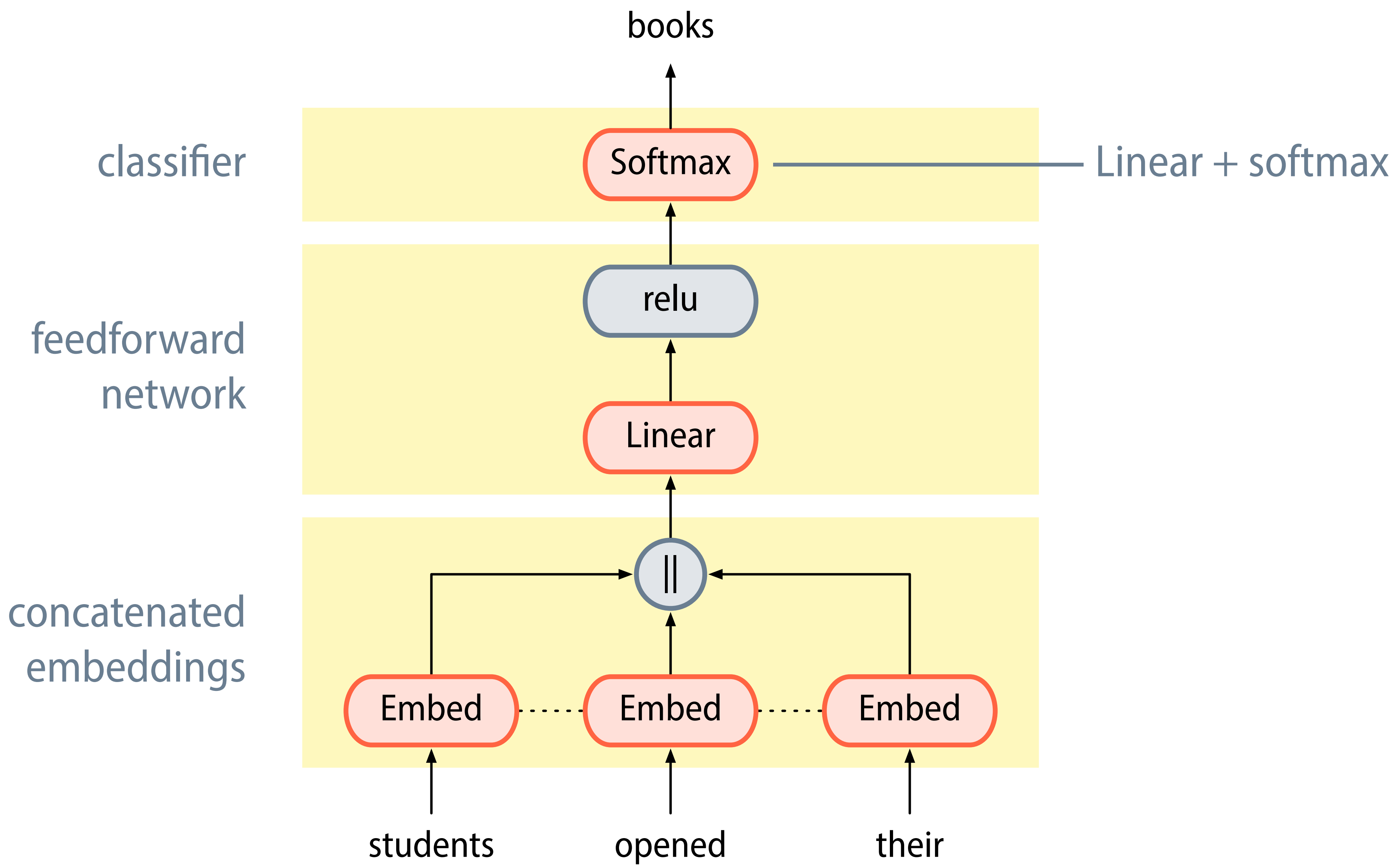
Limitations of n-gram language models

Goldberg § 9.3.2

- Scaling to larger *n*-gram sizes is problematic, both for computational reasons and because of increased sparsity.
- Smoothing techniques are intricate and require careful engineering to retain a well-defined probabilistic interpretation.
- Without additional effort, *n*-gram models are unable to share statistical strength across word boundaries.

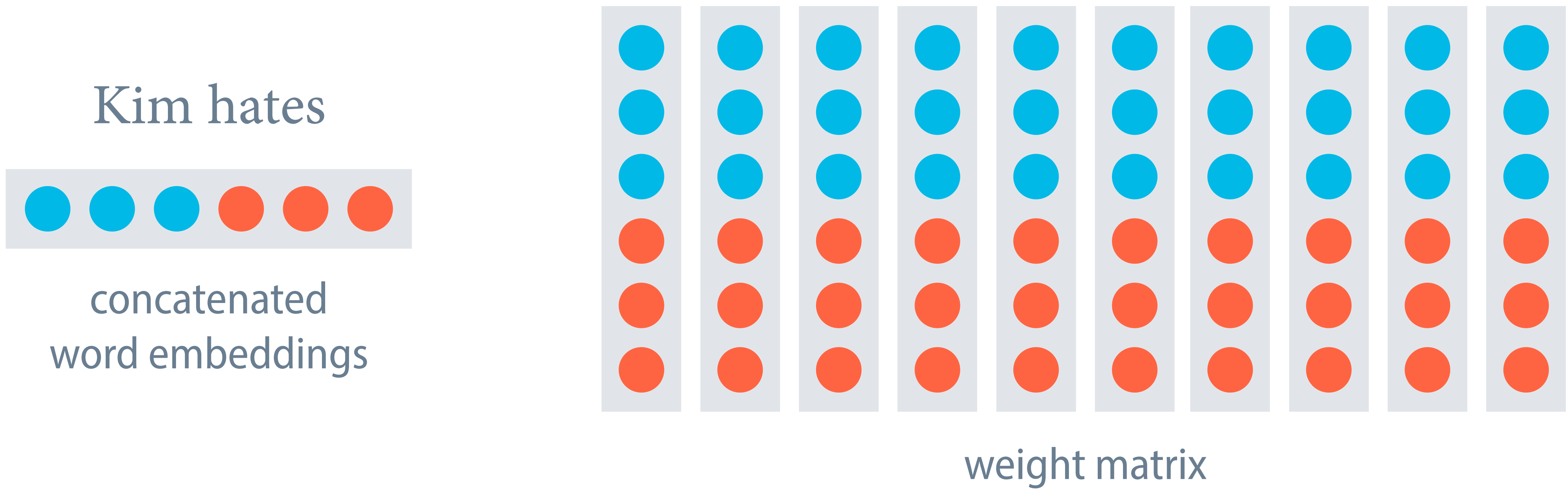
  Observations of *red apple* do not affect estimates for *green apple*.

Fixed-window neural language model



Bengio et al. (2003)

Inefficient use of parameters



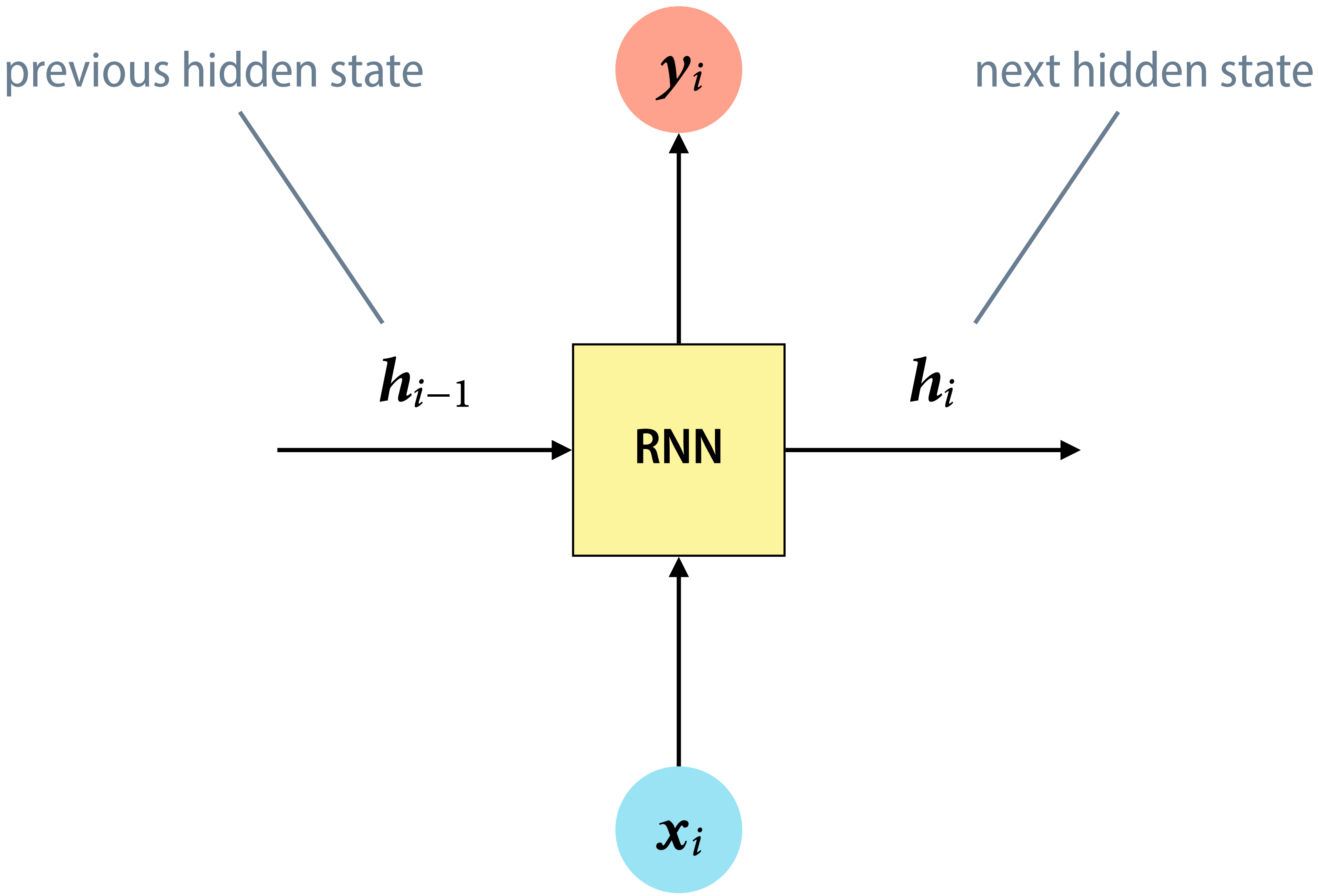
The different parts of the concatenation vector are transformed by completely different weights.

Recurrent neural networks

- **Recurrent neural networks (RNNs)** can process variable length sequences of inputs, such as sequences of letters or words.
- For any input sequence, a recurrent neural network is 'unrolled' into a deep feedforward network.

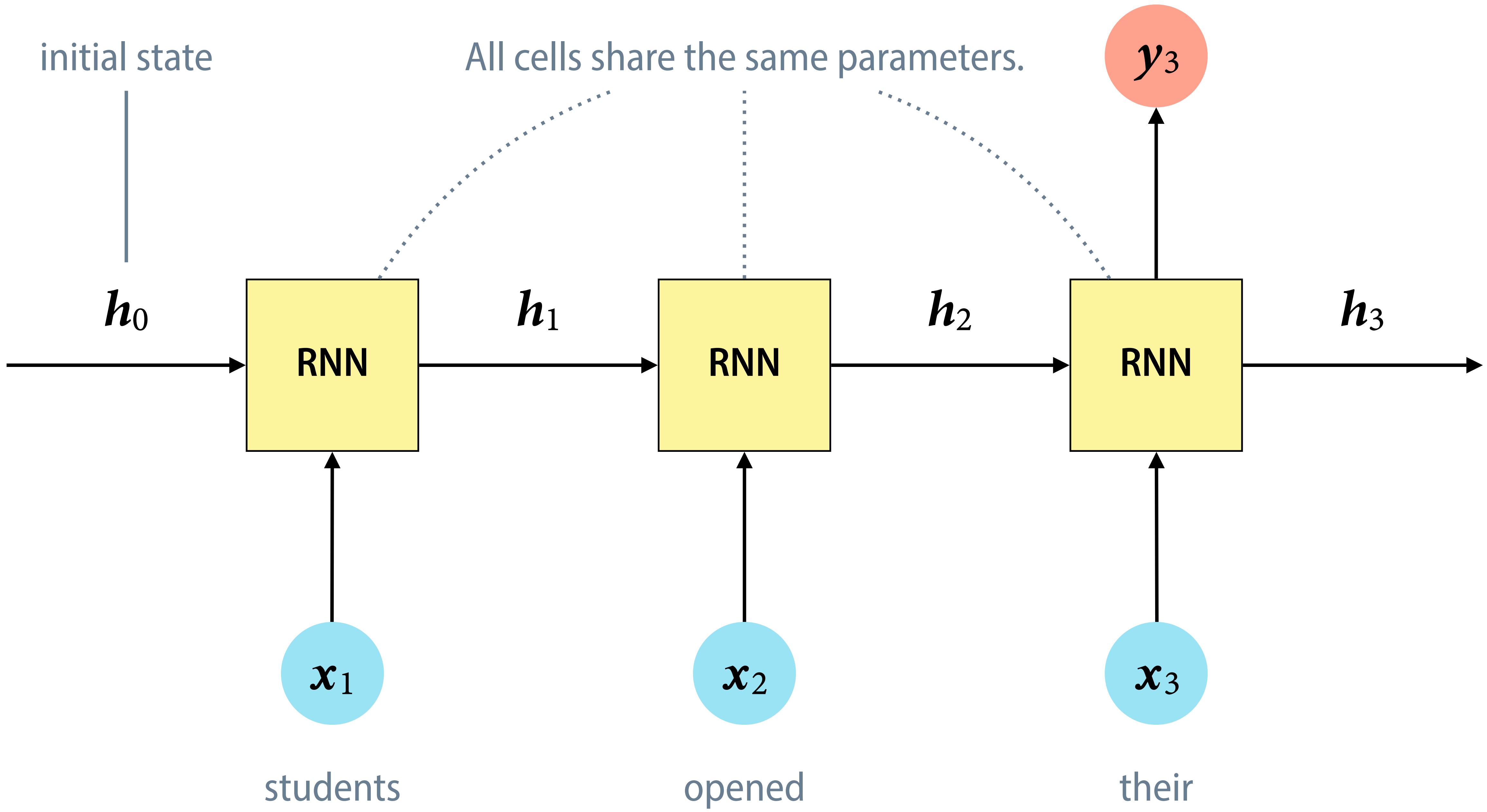
  Depth is proportional to the length of the sequence.
- In contrast to the situation with deep feedforward networks, all parameters are shared across all positions of the sequence.

RNN, recursive view



$$\boldsymbol{h}_i = H(\boldsymbol{h}_{i-1}, \boldsymbol{x}_i) \qquad \boldsymbol{y}_i = O(\boldsymbol{h}_{i-1}, \boldsymbol{x}_i)$$

RNN, unrolled view



Properties of recurrent neural networks

- The parameters of the model are shared across all positions.

  The number of parameters does not grow with the sequence length.

- The output can be influenced by the entire input seen so far.

  Contrast this with the locality constraint of CNNs.

- The hidden state can be a 'lossy summary' of the input sequence.

  Hopefully, it will encode useful information for the task at hand.

Training recurrent neural networks

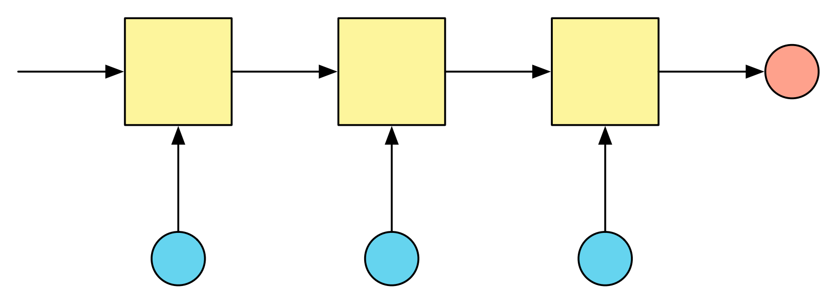
- Unrolled recurrent neural networks are just feedforward networks, and can therefore be trained using backpropagation.

  No specialised algorithm necessary!

- This way of training recurrent neural networks is called **backpropagation through time**.

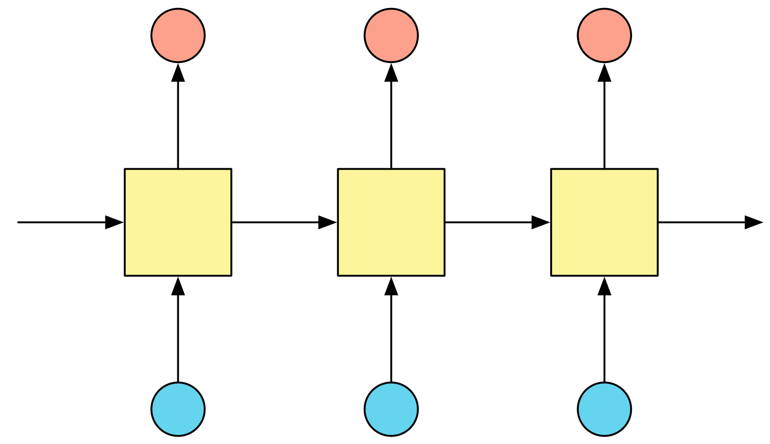
- Shared weights are updated by summing over the gradients computed for each position.

Common usage patterns for RNNs



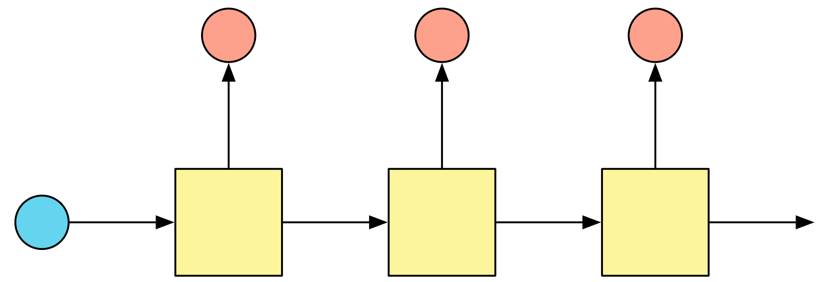
encoder

example: text classification



transducer

example: part-of-speech tagging



decoder

example: text generation

Extensions of the basic RNN architecture

- **Stacked RNNs** are RNNs with several layers, where the outputs of one layer become the inputs of the next.
- **Bidirectional RNNs** combine one RNN that moves forward through the input with another RNN that moves backward.

  outputs at each position are concatenated