RNN Sentiment Analysis

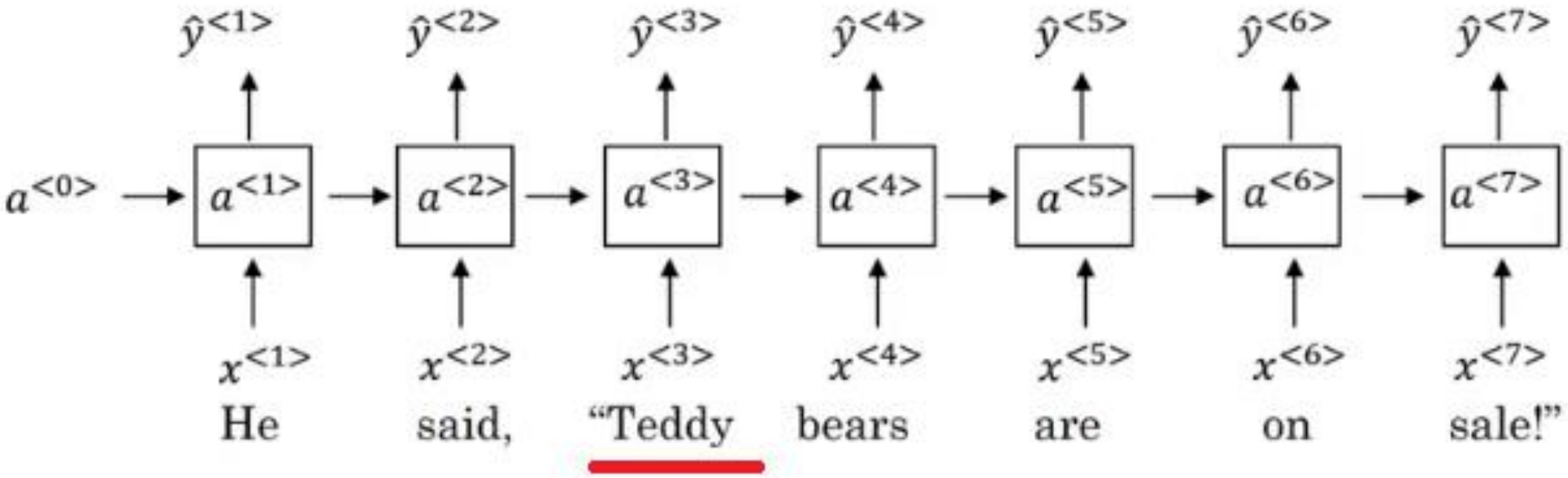
Sentiment Analysis

The dessert is excellent.	★★★★☆
Service was quite slow.	★★☆☆☆
Good for a quick meal, but nothing special.	★★★☆☆
Completely lacking in good taste, good service, and good ambience.	★☆☆☆☆

Slide Credits: Andrew Ng

Bidirectional and Deep RNN

BiRNN Motivation



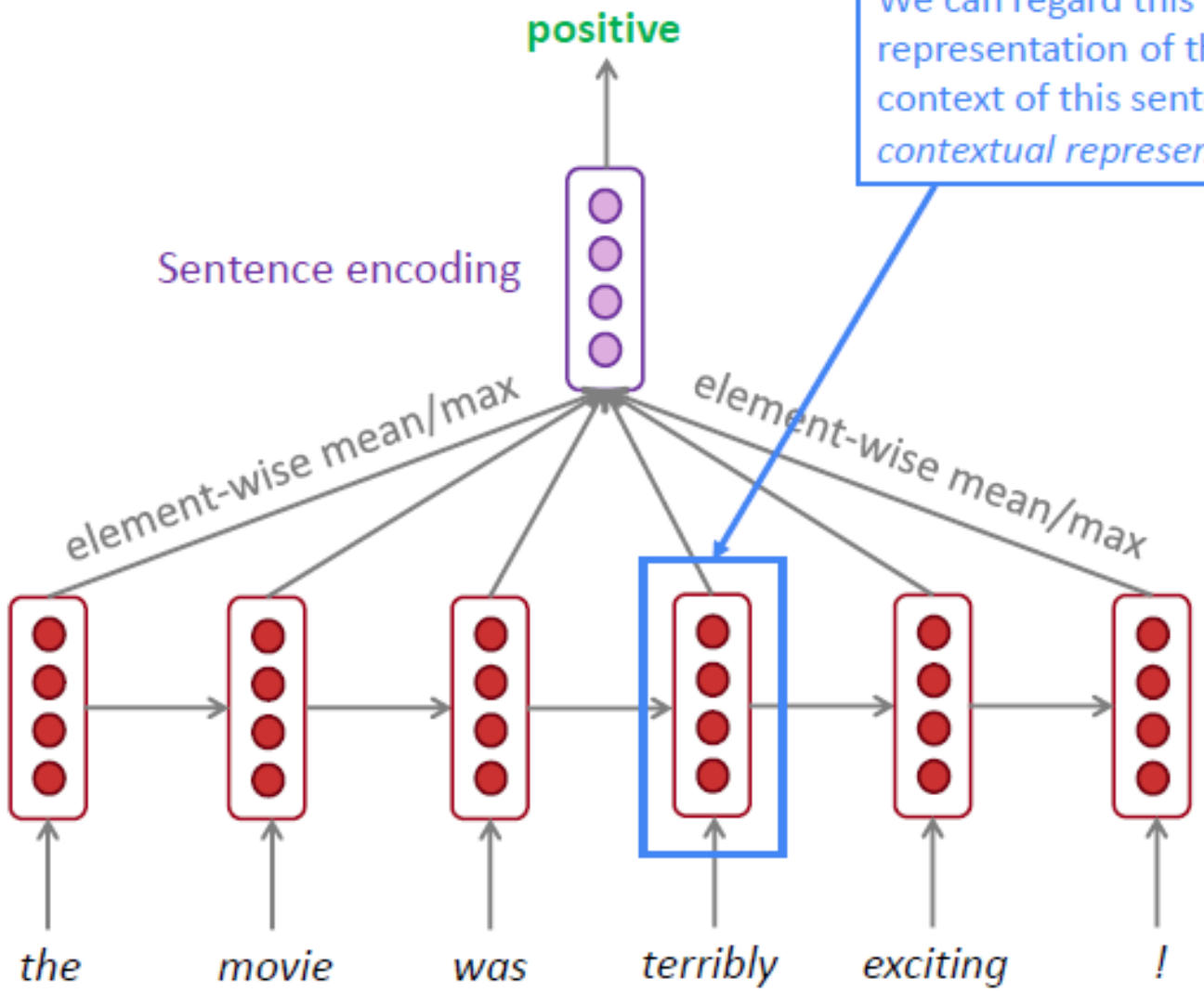
The name **Teddy** cannot be learned from **He** and **said**, but can be learned from **bears**

BiRNNs fixes this issue

Slide Credits: Andrew Ng

Bidirectional RNNs: motivation

Task: Sentiment Classification



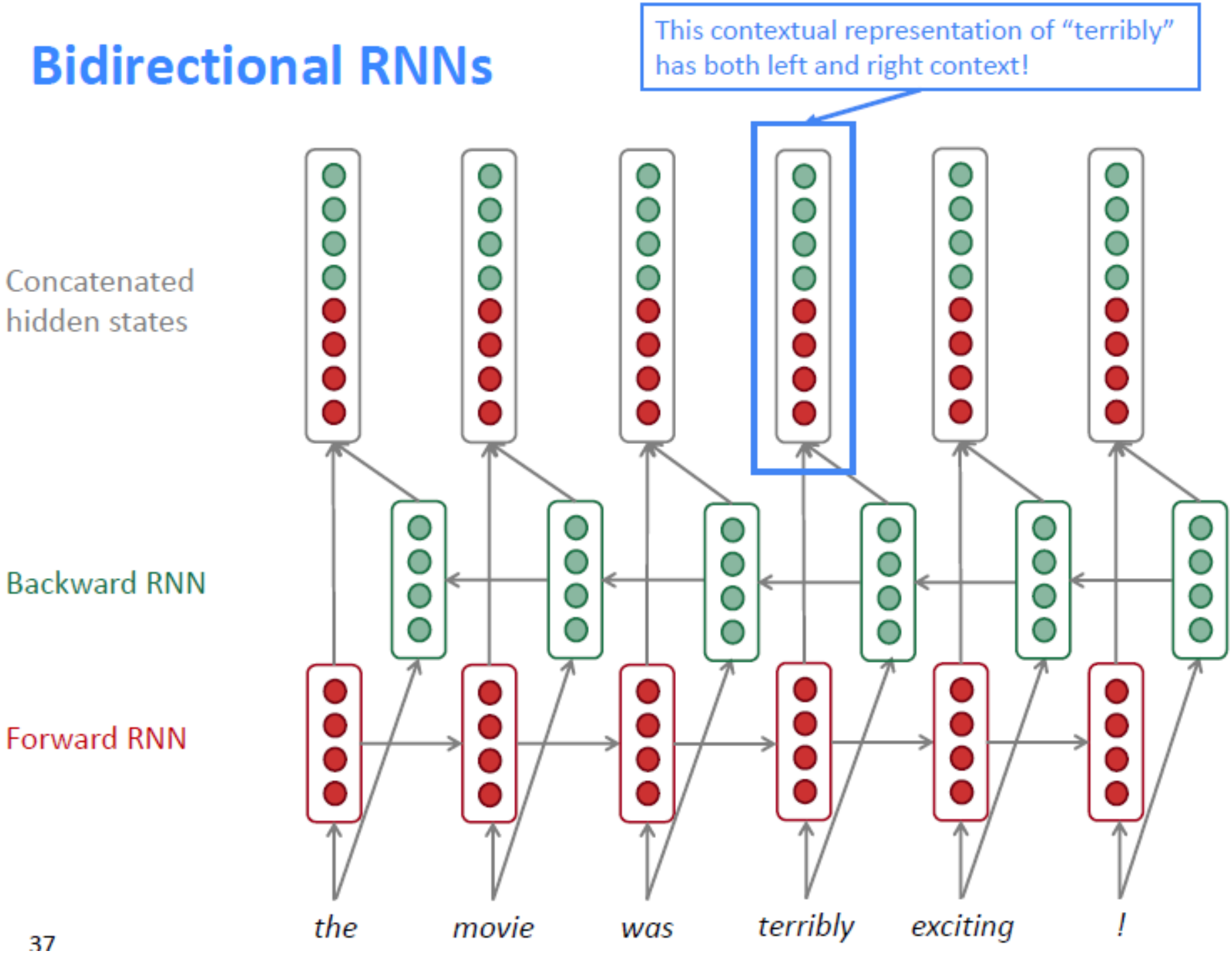
We can regard this hidden state as a representation of the word *"terribly"* in the context of this sentence. We call this a *contextual representation.*

These contextual representations only contain information about the *left* context (e.g. *"the movie was"*).

What about *right* context?

In this example, *"exciting"* is in the right context and this modifies the meaning of *"terribly"* (from negative to positive)

Bidirectional RNNs



Bidirectional RNNs

On timestep t:

This is a general notation to mean "compute one forward step of the RNN" – it could be a vanilla, LSTM or GRU computation.

Forward RNN $\overrightarrow{\boldsymbol{h}}^{(t)} = \mathrm{RNN}_{\mathrm{FW}}(\overrightarrow{\boldsymbol{h}}^{(t-1)}, \boldsymbol{x}^{(t)})$

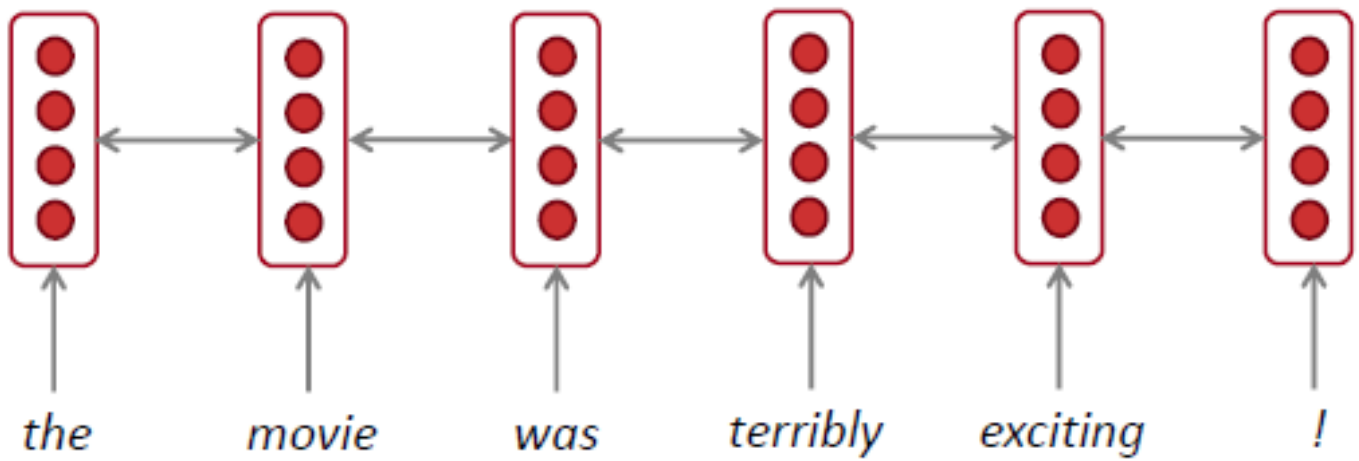
Backward RNN $\overleftarrow{\boldsymbol{h}}^{(t)} = \mathrm{RNN}_{\mathrm{BW}}(\overleftarrow{\boldsymbol{h}}^{(t+1)}, \boldsymbol{x}^{(t)})$

Generally, these two RNNs have separate weights

Concatenated hidden states $\boldsymbol{h}^{(t)} = [\overrightarrow{\boldsymbol{h}}^{(t)}; \overleftarrow{\boldsymbol{h}}^{(t)}]$

We regard this as "the hidden state" of a bidirectional RNN. This is what we pass on to the next parts of the network.

Bidirectional RNNs: simplified diagram



The two-way arrows indicate bidirectionality and the depicted hidden states are assumed to be the concatenated forwards+backwards states.

Bidirectional RNNs

- Note: bidirectional RNNs are only applicable if you have access to the entire input sequence.

- They are **not** applicable to Language Modeling, because in LM you *only* have left context available.

- If you do have entire input sequence (e.g. any kind of encoding), bidirectionality is powerful (you should use it by default).

Multi-layer RNNs

- RNNs are already "deep" on one dimension (they unroll over many timesteps)

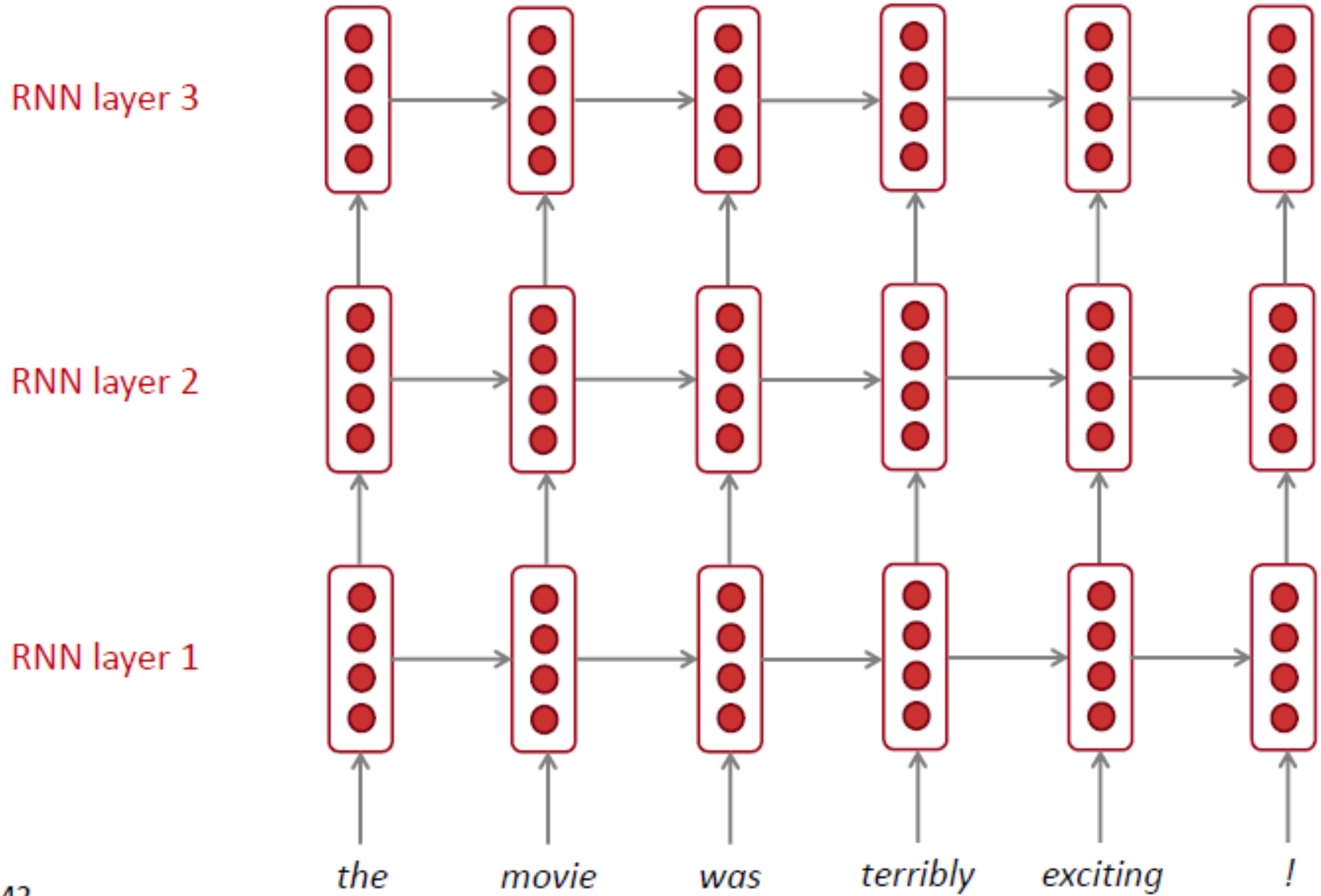
- We can also make them "deep" in another dimension by applying multiple RNNs – this is a multi-layer RNN.

- This allows the network to compute more complex representations
  - The lower RNNs should compute lower-level features and the higher RNNs should compute higher-level features.

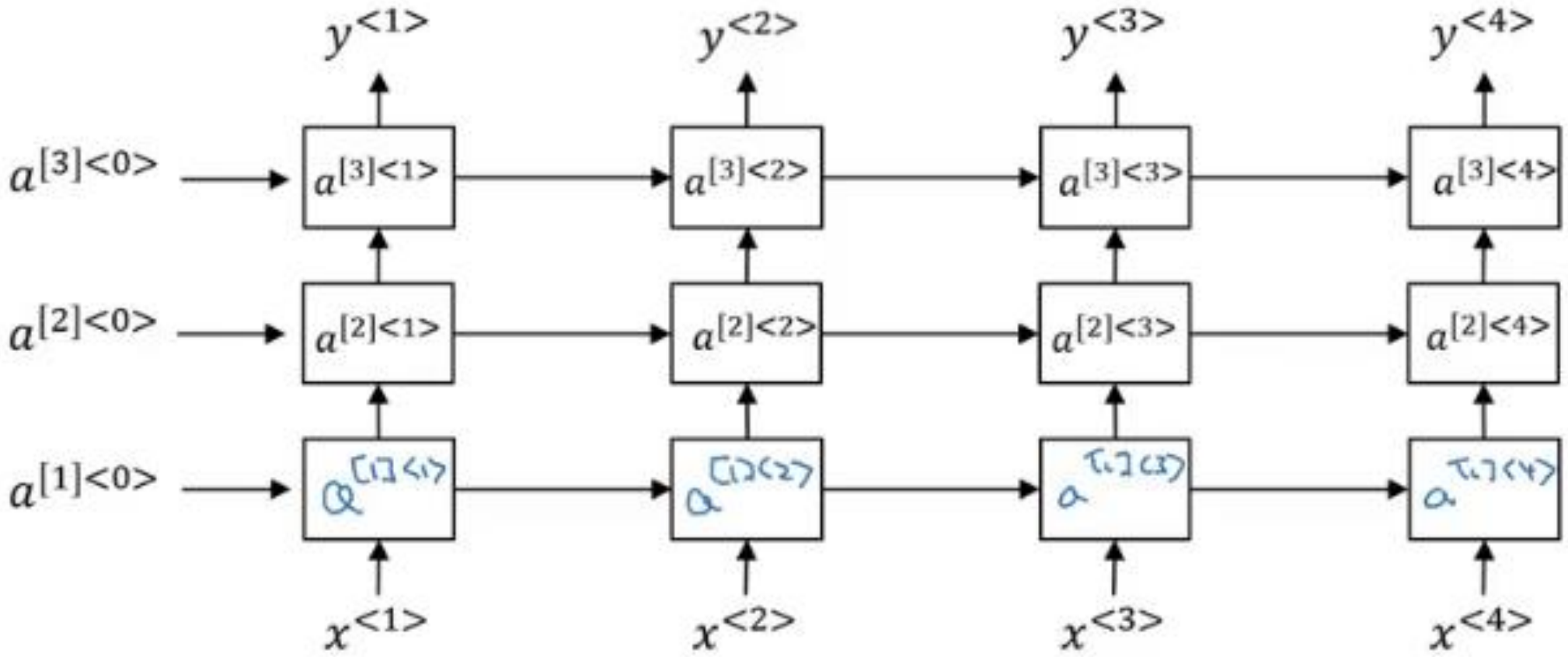
- Multi-layer RNNs are also called *stacked RNNs*.

Multi-layer RNNs

The hidden states from RNN layer *i* are the inputs to RNN layer *i+1*



Deep RNN



Slide Credits: Andrew Ng

Multi-layer RNNs in practice

- High-performing RNNs are often multi-layer (but aren't as deep as convolutional or feed-forward networks)

- For example: In a 2017 paper, Britz et al find that for Neural Machine Translation, 2 to 4 layers is best for the encoder RNN, and 4 layers is best for the decoder RNN
  - However, skip-connections/dense-connections are needed to train deeper RNNs (e.g. 8 layers)

- Transformer-based networks (e.g. BERT) can be up to 24 layers
  - You will learn about Transformers later; they have a lot of skipping-like connections

Reading

- Chapter 9, Speech and Language Processing. Daniel Jurafsky & James H. Martin. Third edition

  https://web.stanford.edu/~jurafsky/slp3/9.pdf