Modeling with Machine Learning: RNN (part 1)

Outline (part 1)

- Modeling sequences
- The problem of encoding sequences
- Recurrent Neural Networks (RNNs)

Temporal/sequence problems

- How to cast as a supervised learning problem?

Temporal/sequence problems

- How to cast as a supervised learning problem?

- Historical data can be broken down into feature vectors and target values (sliding window)

$$\underbrace{\begin{bmatrix} 0.82 \\ 0.80 \\ 0.73 \\ 0.72 \end{bmatrix}}_{\phi(t)} \qquad \underbrace{0.89}_{y^{(t)}}$$

Temporal/sequence problems

- Language modeling: what comes next?

This course has been a tremendous …

Temporal/sequence problems

- Language modeling: what comes next?

This course has been a tremendous ...

tremendous $\begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}$

?

a $\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

$\phi(t)$ $y^{(t)}$

Temporal/sequence problems

- Language modeling: what comes next?

This course has been a tremendous …

a $\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

tremendous

been $\begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$

$\phi(t)$ $y^{(t)}$

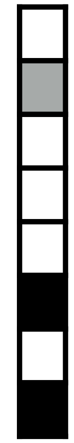
What are we missing?

- Sequence prediction problems can be recast in a form amenable to feed-forward neural networks
- But we have to engineer how “history” is mapped to a vector (representation). This vector is then fed into, e.g., a neural network
  - how many steps back should we look at?
  - how to retain important items mentioned far back?
- Instead, we would like to learn how to encode the “history” into a vector

Learning to encode/decode

- Language modeling

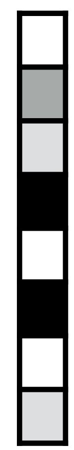
This course has been a



success (?)

- Sentiment classification

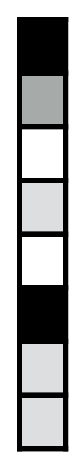
I have seen better lectures



-1

- Machine translation

I have seen better lectures



Olen nähnyt parempia luentoja

encoding **decoding**

Key concepts

- **Encoding** (this lecture)
  - e.g., mapping a sequence to a vector
- **Decoding** (next lecture)
  - e.g., mapping a vector to, e.g., a sequence

Encoding everything

words

$$\begin{bmatrix} .1 \\ .3 \\ .4 \end{bmatrix} \begin{bmatrix} .7 \\ .1 \\ .0 \end{bmatrix} \begin{bmatrix} .2 \\ .8 \\ .3 \end{bmatrix} \ldots$$

“Efforts and courage are not enough without purpose and direction” — JFK

sentences

$$\begin{bmatrix} .2 \\ .3 \\ .6 \end{bmatrix}$$

images

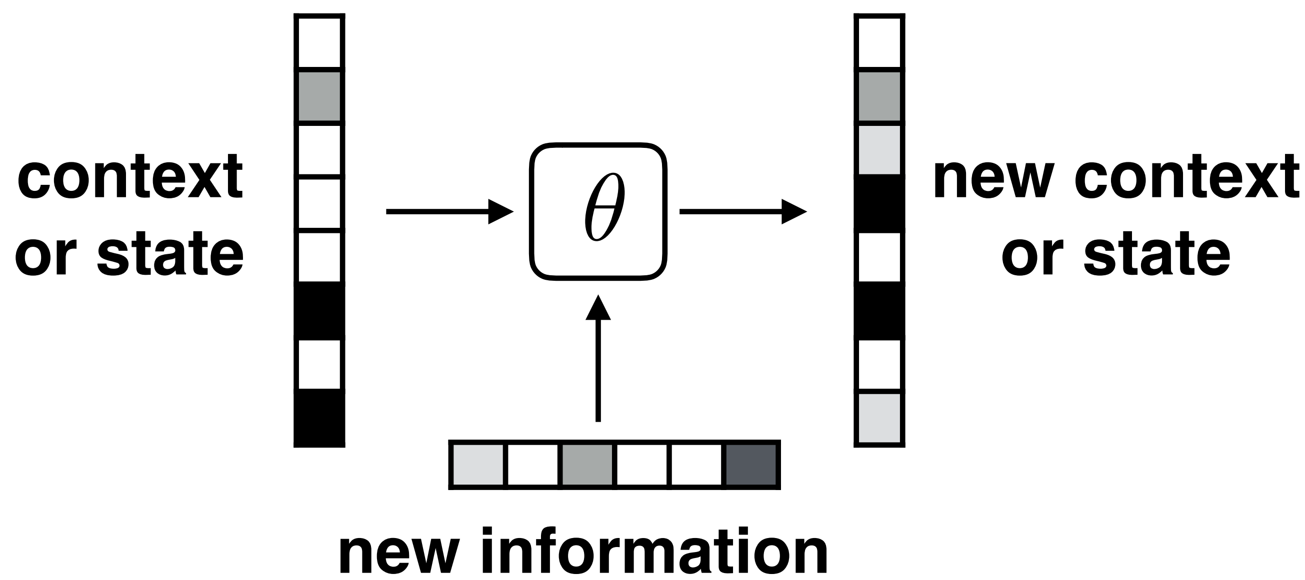
$$\begin{bmatrix} .3 \\ .3 \\ .5 \end{bmatrix}$$

$$\begin{bmatrix} .2 \\ .4 \\ .6 \end{bmatrix}$$

events

Example: encoding sentences

- Easy to introduce adjustable “lego pieces” and optimize them for end-to-end performance



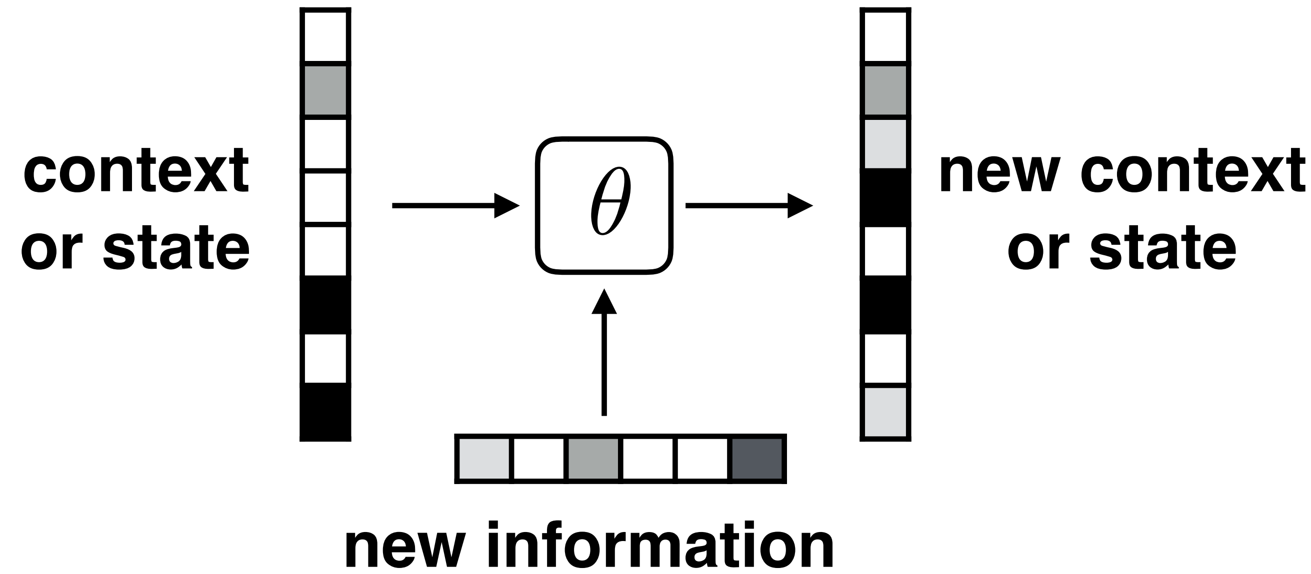
RNN

<null>

Efforts and courage are not …

Example: encoding sentences

- Easy to introduce adjustable “lego pieces” and optimize them for end-to-end performance



RNN

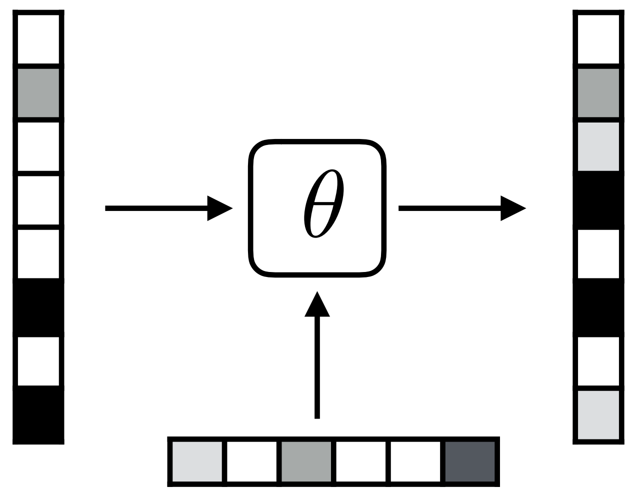
$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

<null>

Efforts and courage are not …

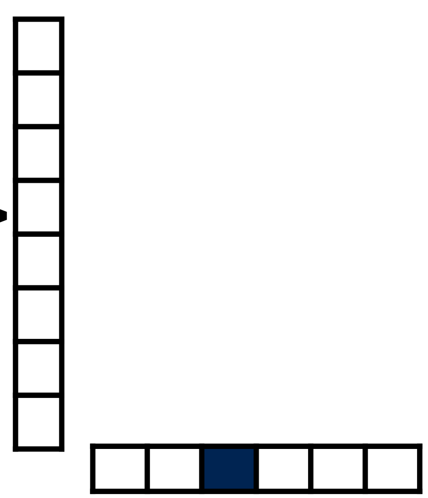
Example: encoding sentences

- Easy to introduce adjustable “lego pieces” and optimize them for end-to-end performance



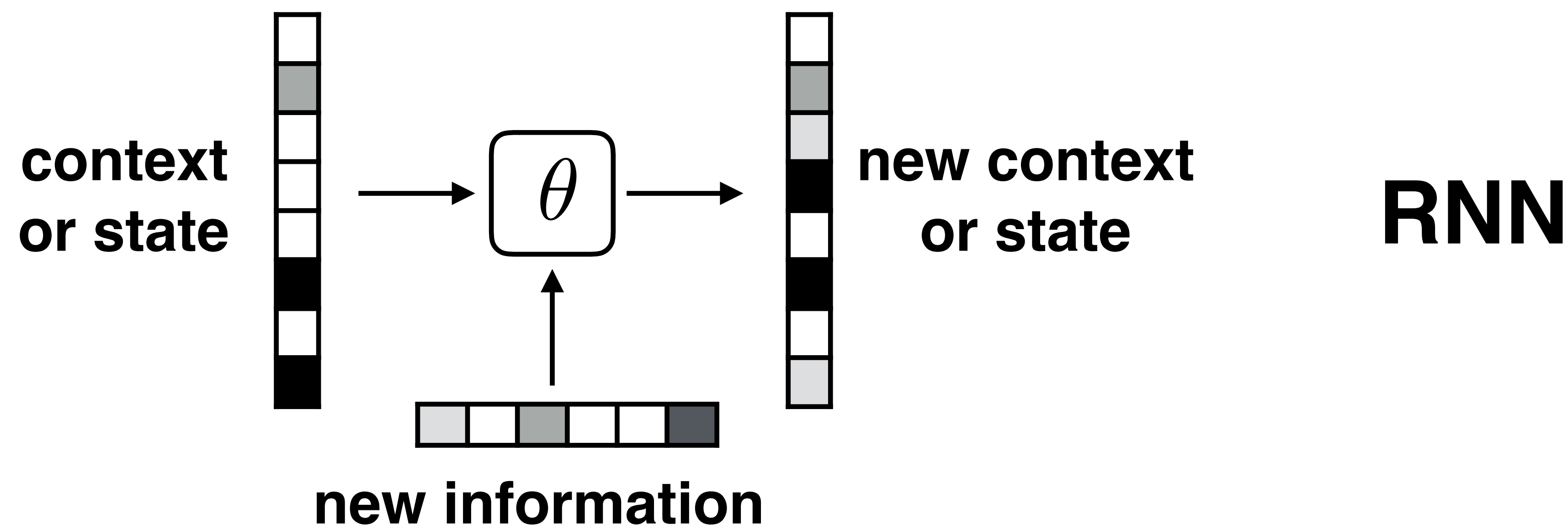
RNN

$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$



Example: encoding sentences

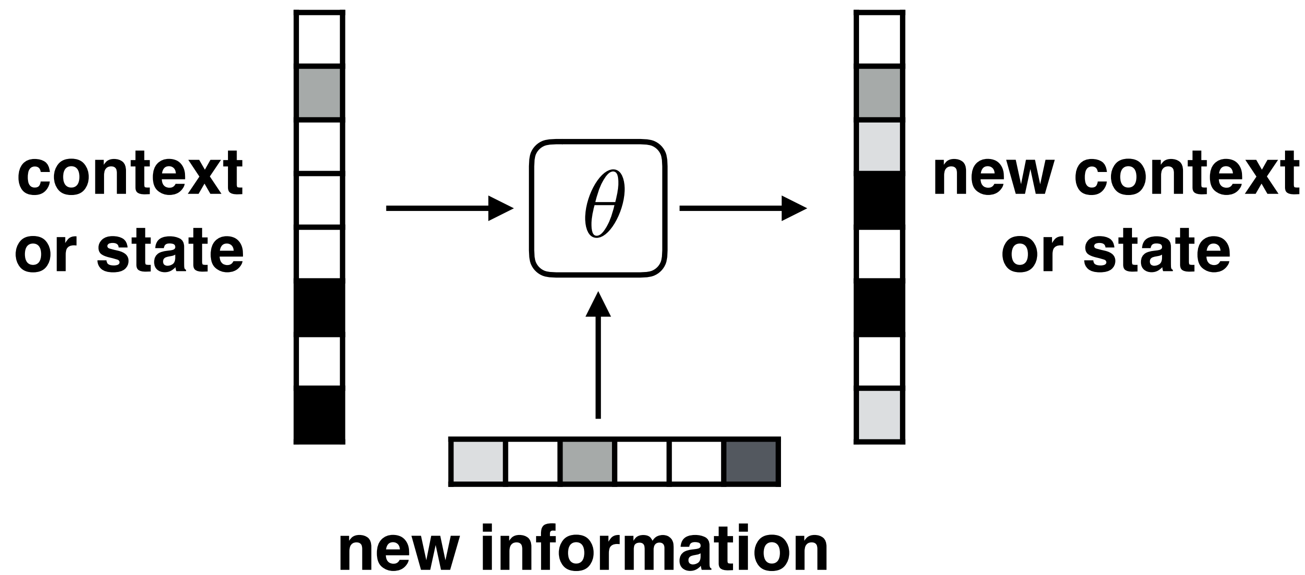
- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance



$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

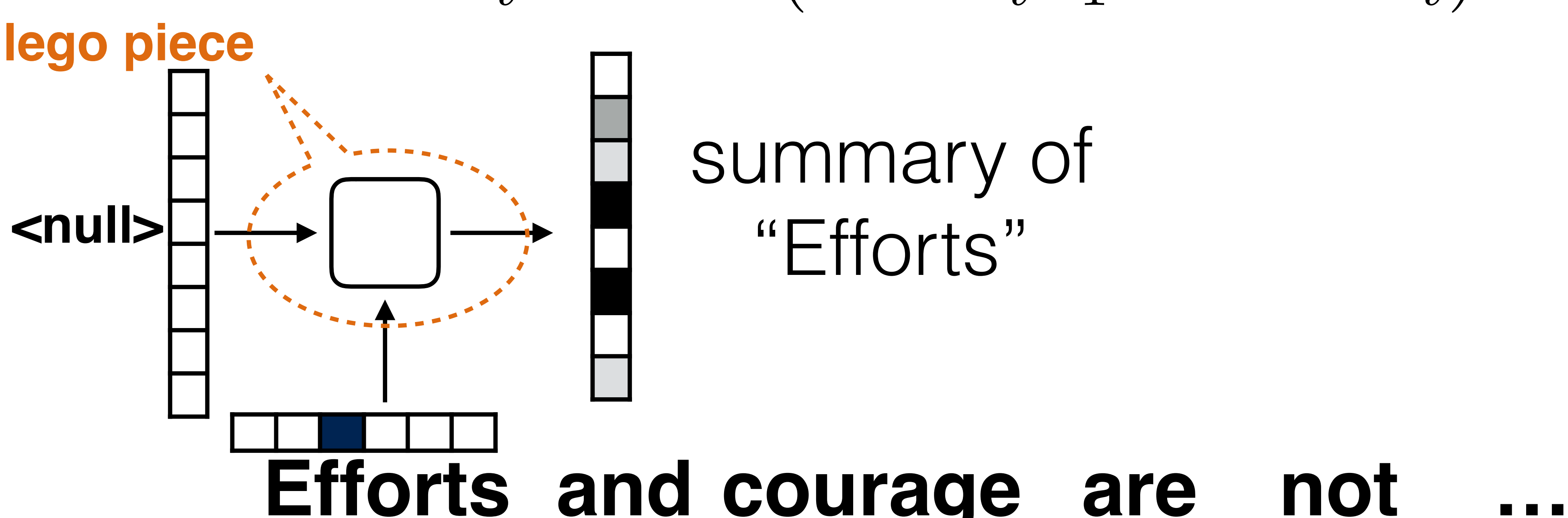
Example: encoding sentences

- Easy to introduce adjustable “lego pieces” and optimize them for end-to-end performance



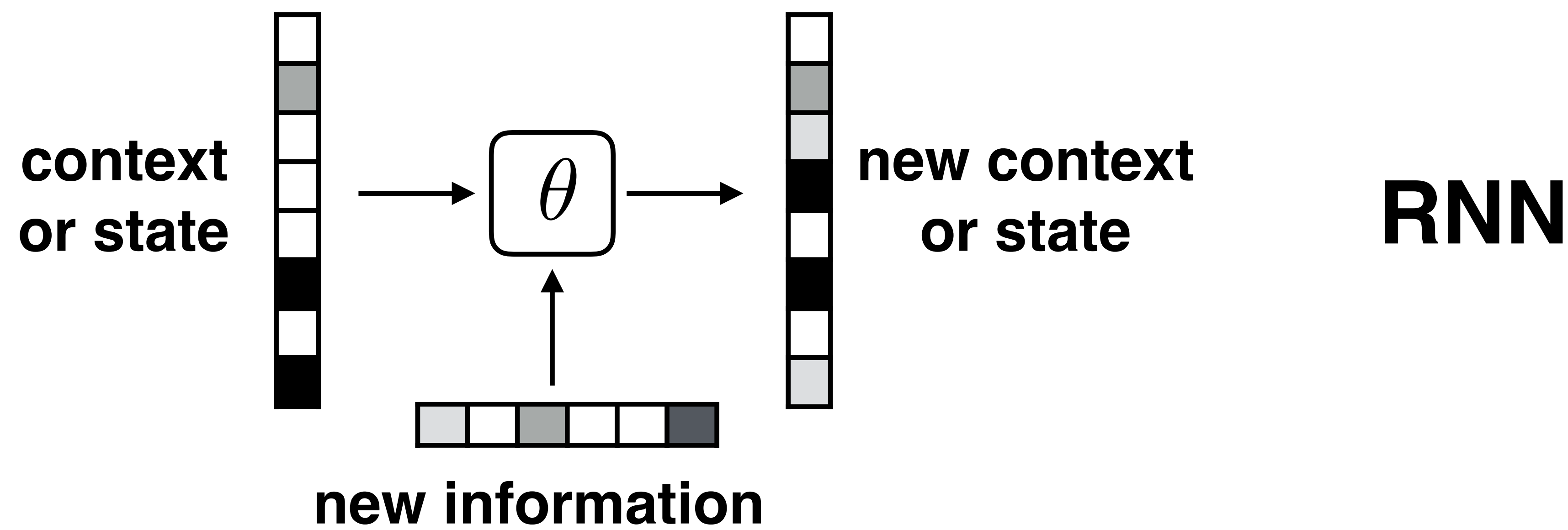
RNN

$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

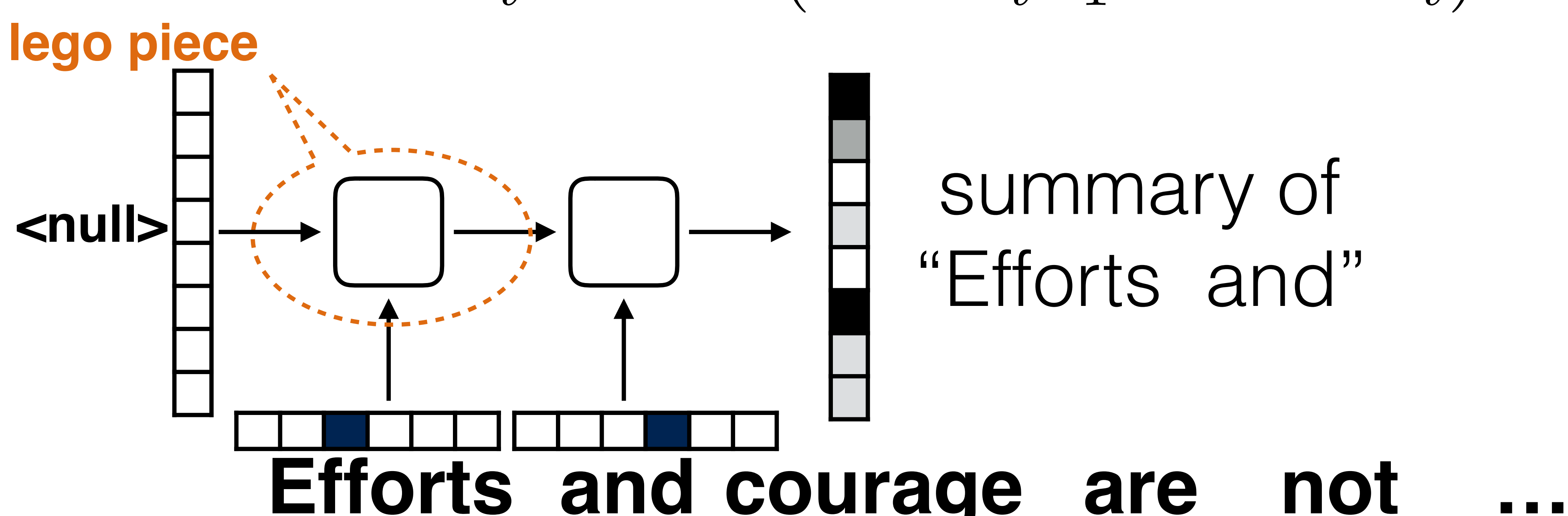


Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance

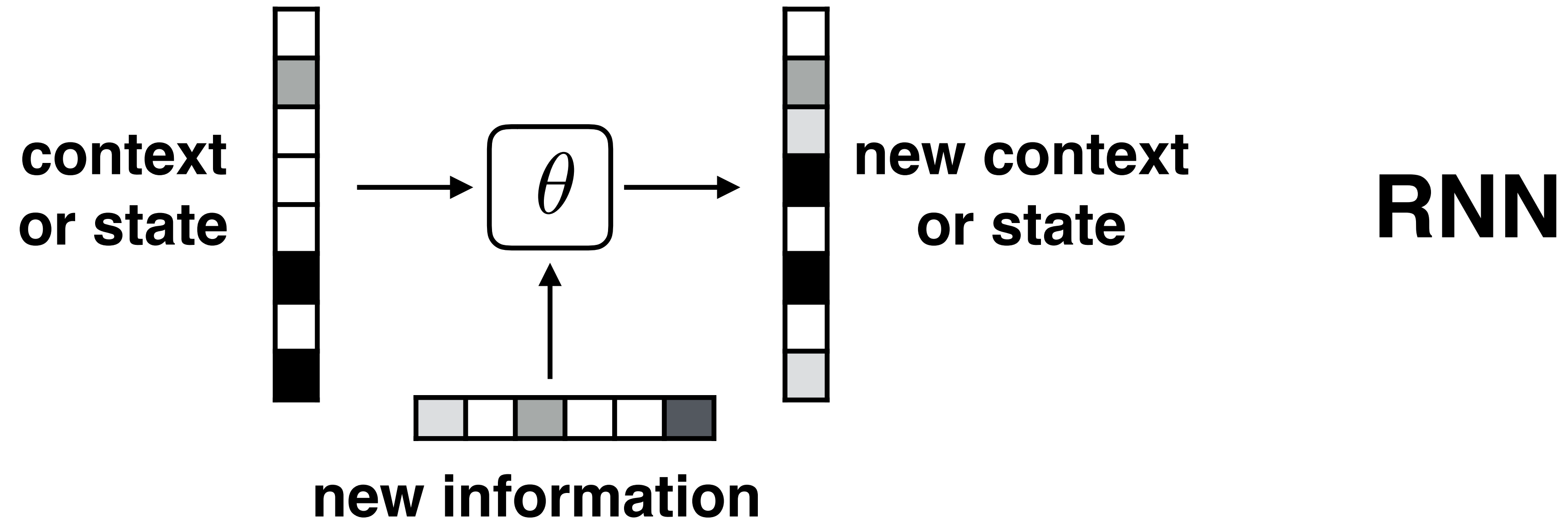


$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

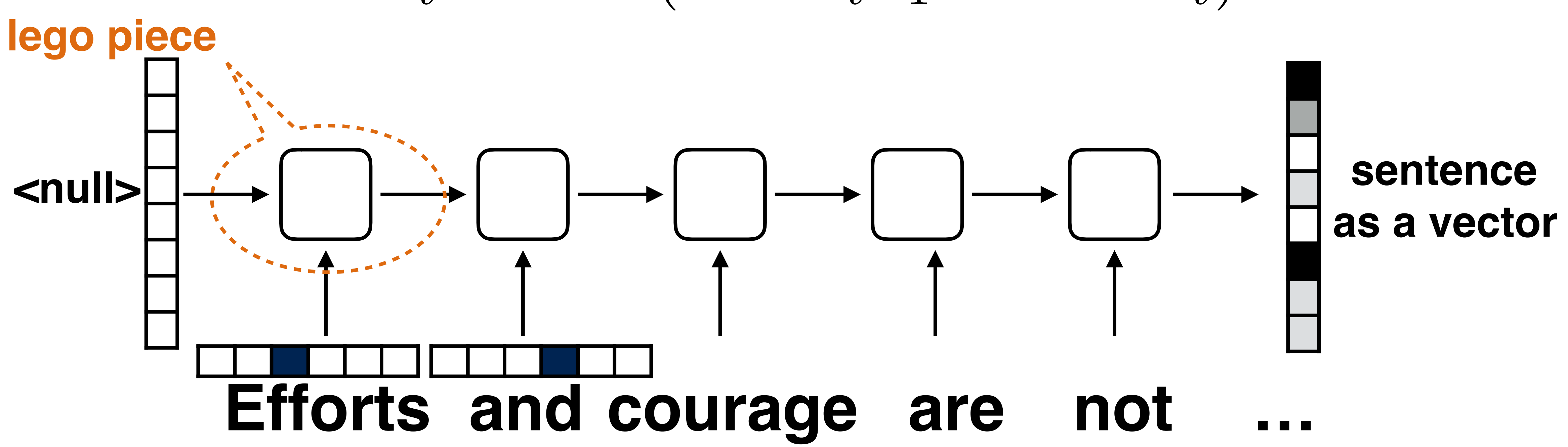


Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance

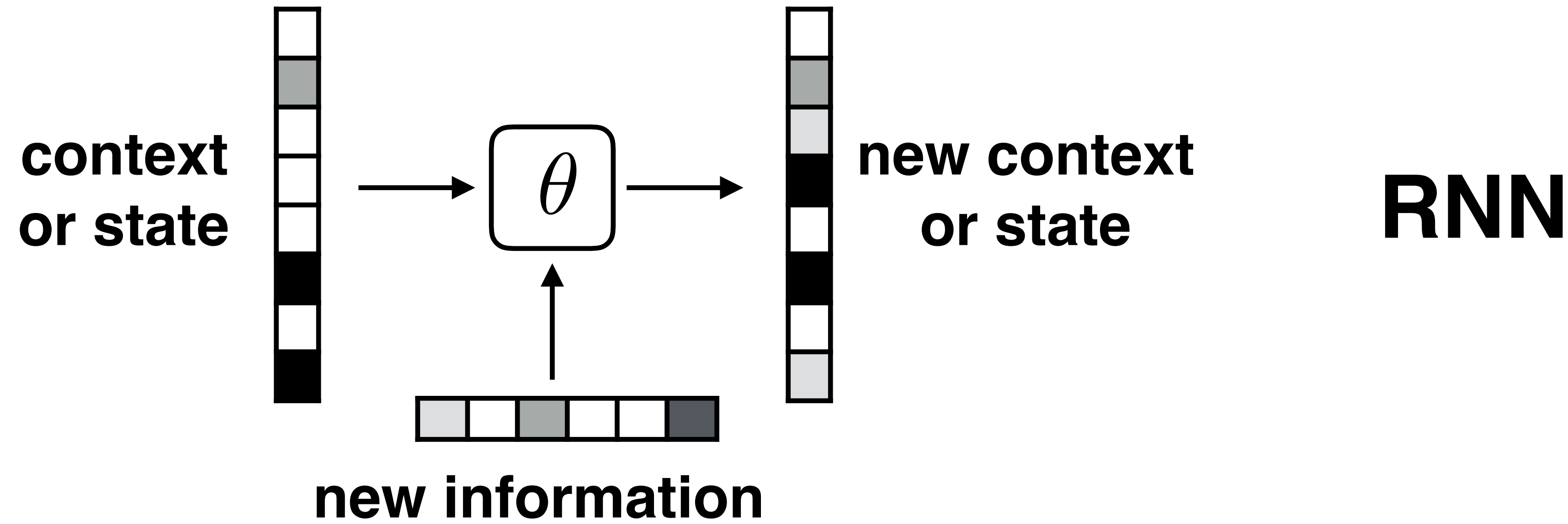


$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

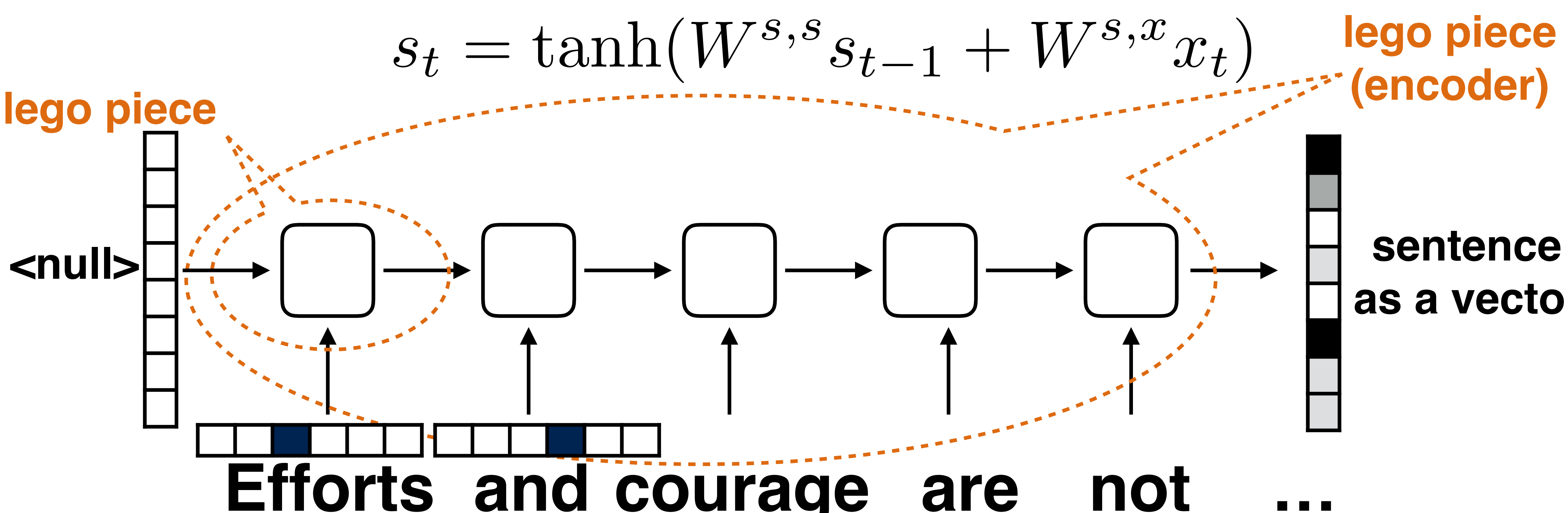


Example: encoding sentences

- Easy to introduce adjustable “lego pieces” and optimize them for end-to-end performance

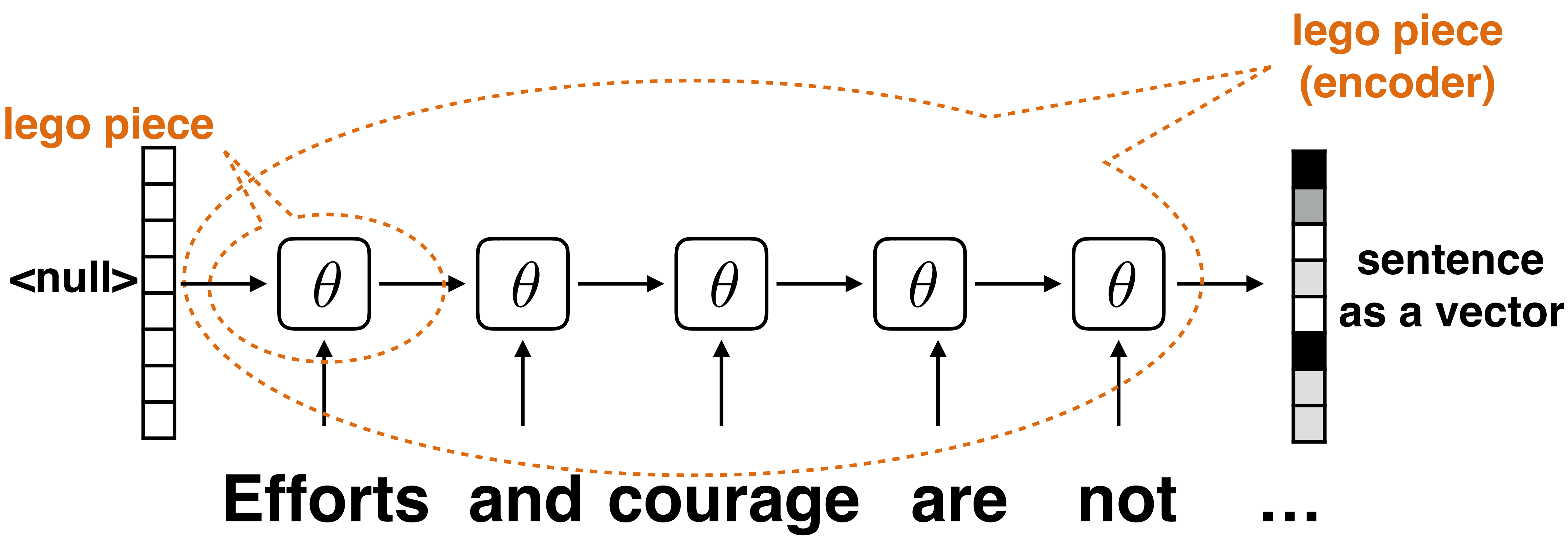


$$s_t = \tanh(W^{s,s}s_{t-1} + W^{s,x}x_t)$$



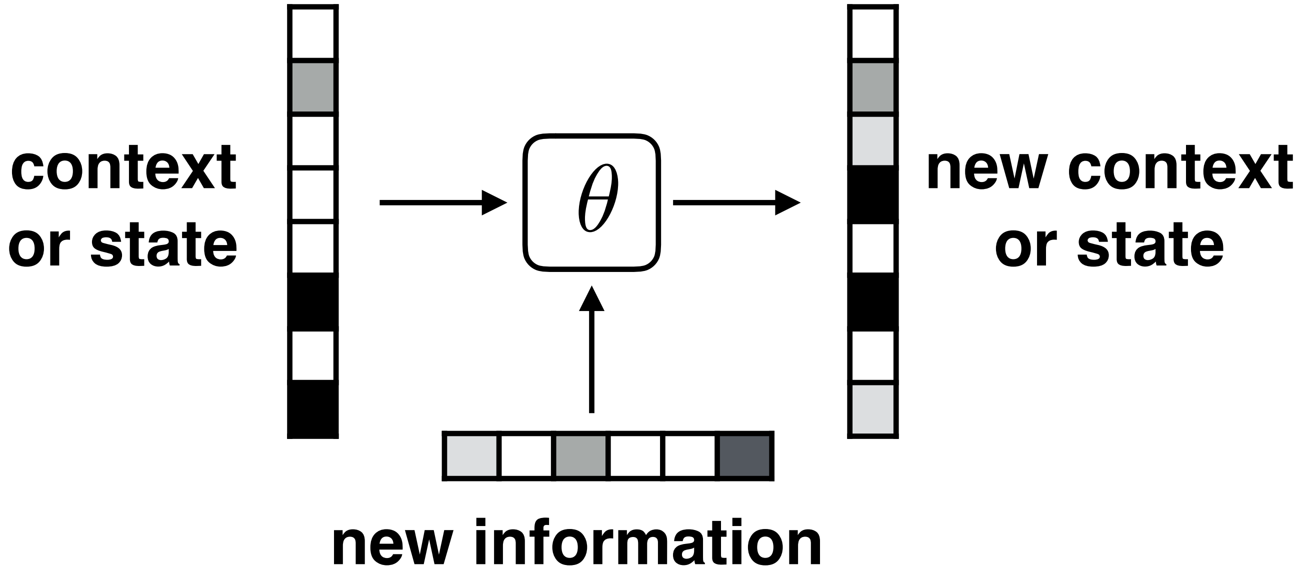
Example: encoding sentences

- There are three differences between the encoder (unfolded RNN) and a standard feed-forward architecture
  - input is received at each layer (per word), not just at the beginning as in a typical feed-forward network
  - the number of layers varies, and depends on the length of the sentence
  - parameters of each layer (representing an application of an RNN) are shared (same RNN at each step)



What's in the box?

- We can make the RNN more sophisticated…

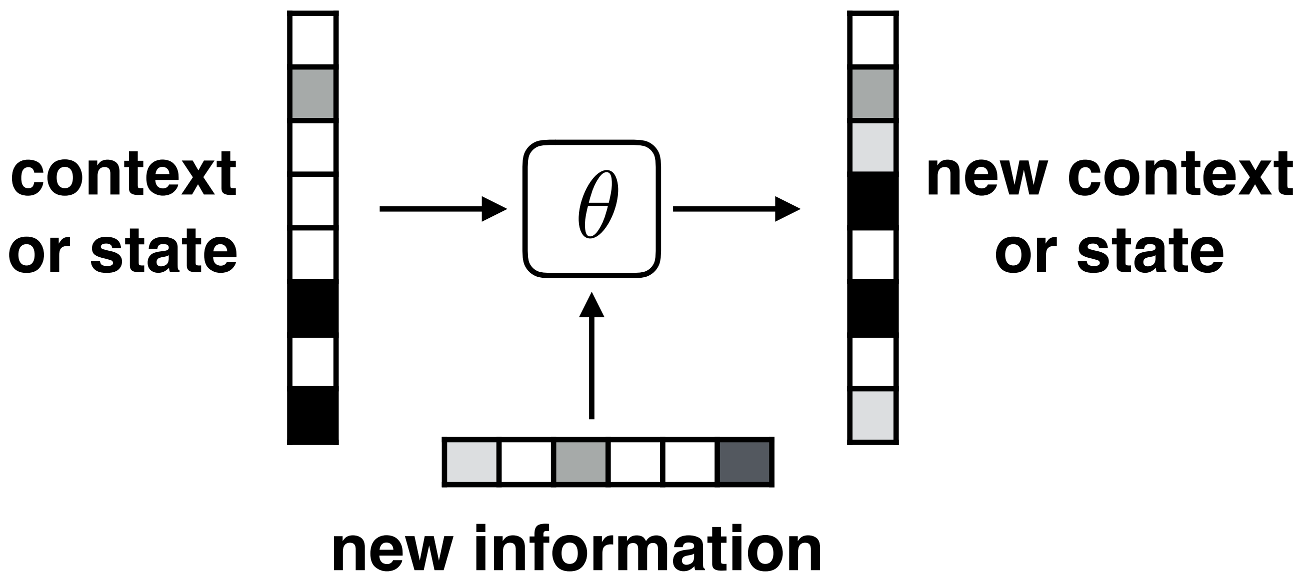


basic RNN

$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

What's in the box?

- We can make the RNN more sophisticated…



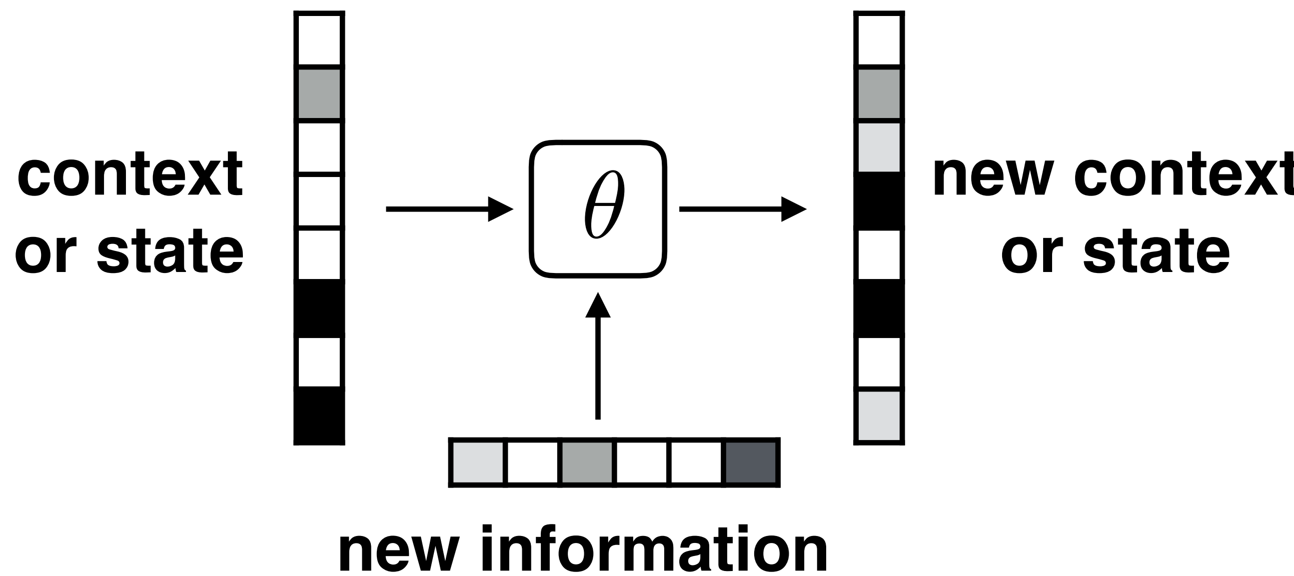
simple gated RNN

$$g_t = \mathrm{sigmoid}(W^{g,s} s_{t-1} + W^{g,x} x_t)$$

$$s_t = (1 - g_t) \odot s_{t-1} + g_t \odot \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

What's in the box?

- We can make the RNN more sophisticated…



LSTM
(one of many)

$$f_t = \text{sigmoid}(W^{f,h} h_{t-1} + W^{f,x} x_t)$$ forget gate

$$i_t = \text{sigmoid}(W^{i,h} h_{t-1} + W^{i,x} x_t)$$ input gate

$$o_t = \text{sigmoid}(W^{o,h} h_{t-1} + W^{o,x} x_t)$$ output gate

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W^{c,h} h_{t-1} + W^{c,x} x_t)$$ memory cell

$$h_t = o_t \odot \tanh(c_t)$$ visible state

Key things

- Neural networks for sequences: encoding
- RNNs, unfolded
  - state evolution, gates
  - relation to feed-forward neural networks
  - back-propagation (conceptually)
- Issues: vanishing/exploding gradient
- LSTM (operationally)