CS 378 Lecture 16

Today

- RNNs
- LSTMs (the type of RNN you will be using)
- Implementation

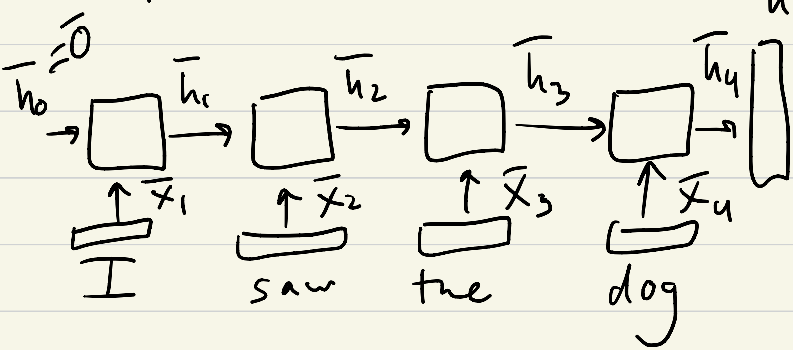
Announcements

- A4
- Midterm back soon

LM: $P(\bar{w})$ or $P(w_i \mid w_1, \ldots w_{i-1})$
"predict the next word"

Recap RNNs + Language modeling

$\bar{h}$ d-dim

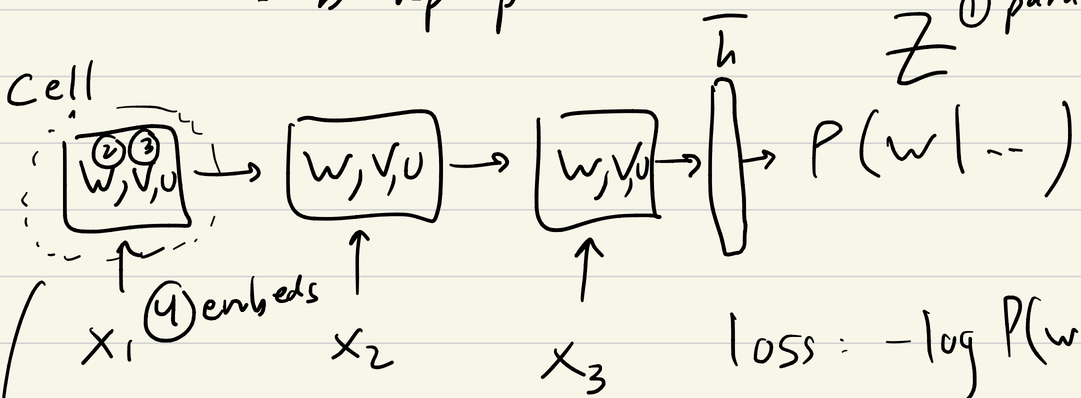


Z : $|V| \times d$ parameter matrix

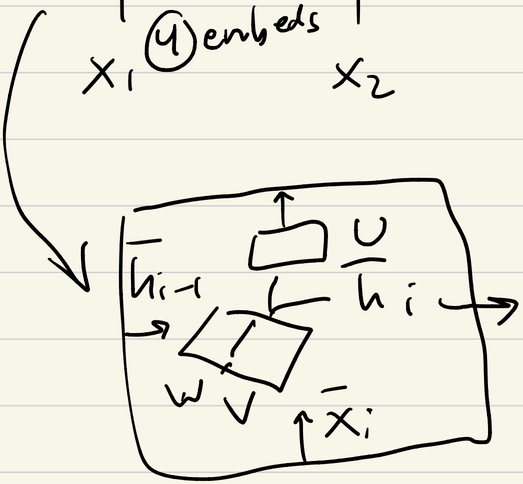
$$P(w \mid \text{I saw the dog}) = \text{softmax}(Z\bar{h})$$

Training "Backpropagation through time"

= backprop



loss: $-\log P(w_i | \dots)$

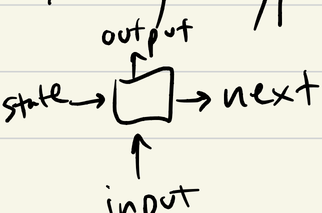


Multiple updates for W, V, U ⇒ no problem

Elman network

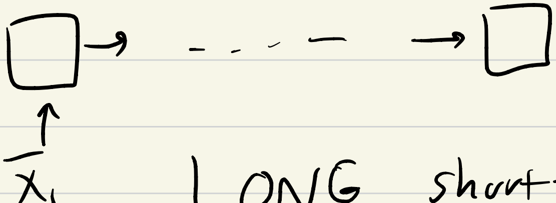
Long Short-term memory networks (LSTMs)

Many types of RNNs



LSTMs (1998)

short-term memory: what the model can remember in its state

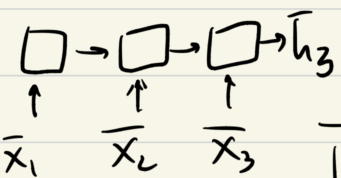


? does the model "remember" $\bar{x}_1$?

LONG short-term memory (remember for longer)

Problem w/ Elman networks

vanishing/exploding gradients



$$\bar{h}_i = \tanh(W\bar{x}_i + V\bar{h}_{i-1})$$

$$\bar{h}_3 = \tanh(W\bar{x}_3 + V \cdot \tanh(W\bar{x}_2 + V \cdot \tanh(W\bar{x}_1 + \bar{0})))$$

Assume tanh is the identity fcn

$$\bar{h}_3 = W\bar{x}_3 + VW\bar{x}_2 + V^2 W\bar{x}_1$$

after n steps $\Rightarrow V^{n-1}\bar{x}_1$

LSTM gates

~~Elman:~~ $\cancel{\bar{h}_i = \tanh(W\bar{x}_i + V\bar{h}_{i-1})}$

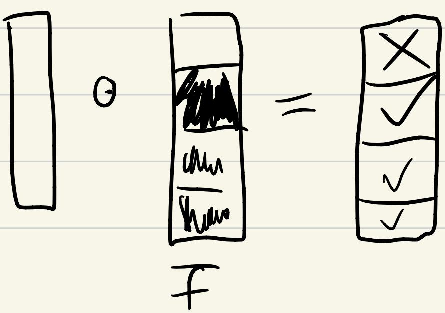
input gate

Gated: $\bar{h}_i = \bar{h}_{i-1} \odot \bar{f} + \text{function}(\bar{x}_i, \bar{h}_{i-1}) \odot \bar{i}$

prev state elementwise $\times$

$\bar{f}$: forget gate, values in $[0,1]$

$\bar{h}_{i-1}$



If $\bar{f} = 1$:

$\bar{h}_{i-1}$ is totally preserved

(added for poll exercise)

Where do f, i come from?

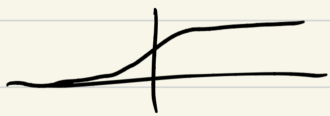
bias

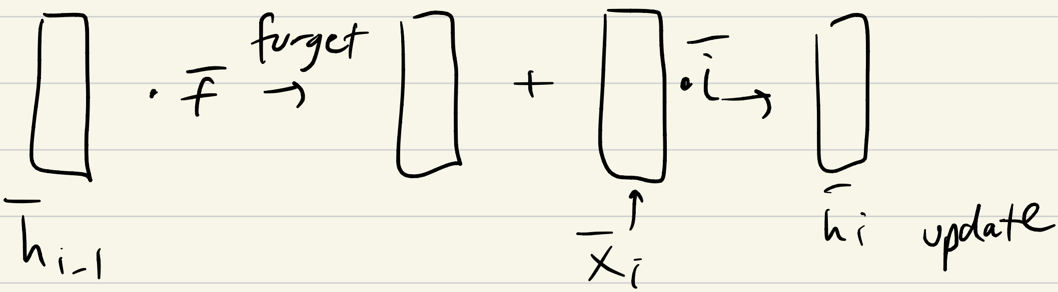
$$f = \text{sigmoid}\left(W^{(1)}\bar{x}_i + W^{(2)}\bar{h}_{i-1} + b_{forget}\right)$$

$$i = \text{sigmoid}\left(W^{(3)}\bar{x}_i + W^{(4)}\bar{h}_{i-1} + \bar{b}_{input}\right)$$

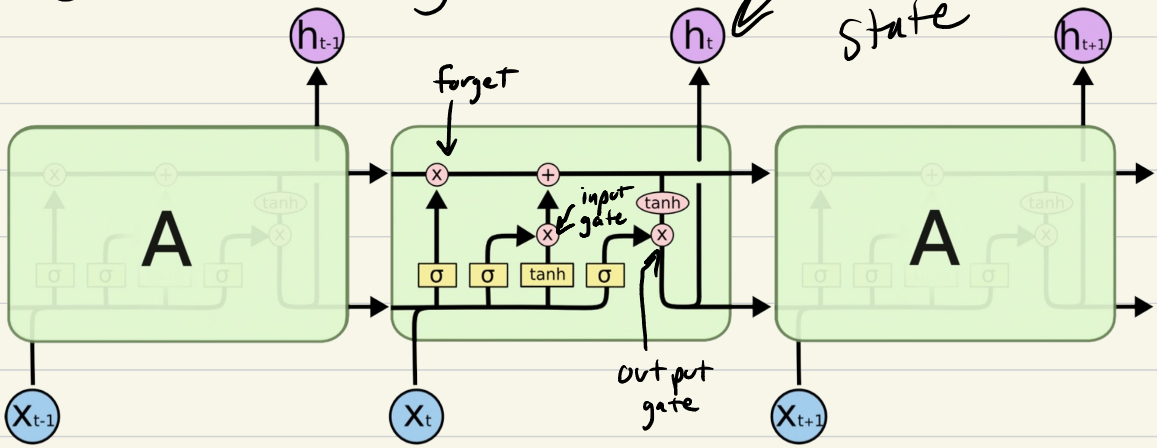
sigmoid

$$\sigma = \frac{e^x}{1+e^x}$$





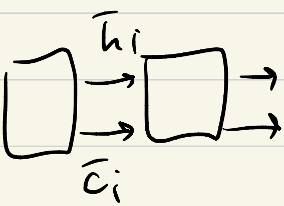
Chris Olah's blog



LSTM: 8 weight matrices

hidden state $\bar{h}$
cell state $\bar{c}$

] tuple of the LSTM state



Poll :

discussed lstm_lecture.py