Today: LSTMs, GRUs (vaguely)
Bidirectional RNNs
Deep RNNs
code examples: sentiment analysis for IMDB dataset

Problem (Vanishing gradients)

Simple RNN works for short sequences, but the hidden state acts as a primitive memory, so run into issues of gradient vanishing/exploding for longer sequences. This is because it attempts to retain all the information on the sequence prefix seen before.

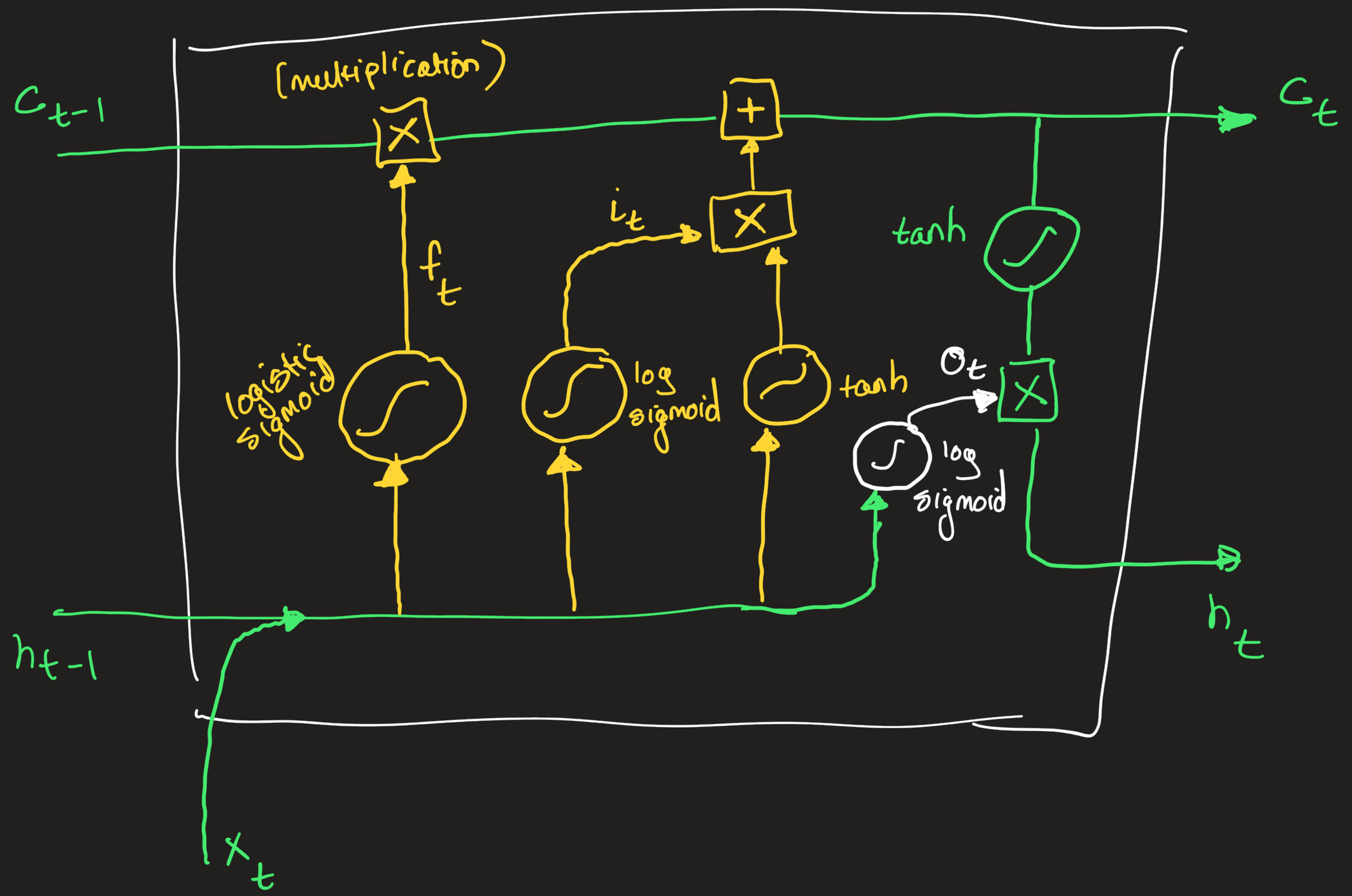
Resolution (in practice, by consensus, the most popular architectures)

LSTM (Long-Short Term Memories; 1997)
GRUs (Gated Recurrent Units; 2014)

focus on LSTMs: basic intuition is to selectively choose what you will remember or forget.

Introduce:
- cell state to augment our memory in addition to the hidden state, c_t (in cell state)
- forget gate — determine what we forget at each time, f_t
- input gate — determine what we remember in our cell state, i_t

Flow diagram for LSTM



$$i_t = \text{sigmoid}\left(\omega_{hi} h_{t-1} + \omega_{xi} x_t\right)$$

$$f_t = \text{sigmoid}\left(\omega_{hf} h_{t-1} + \omega_{xf} x_t\right)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh\left(\omega_{hc} h_{t-1} + \omega_{xc} x_t\right)$$

$$o_t = \text{sigmoid}\left(\omega_{ho} h_{t-1} + \omega_{xo} x_t\right)$$

$$h_t = o_t \odot \tanh\left(\omega_{ch} c_t\right)$$

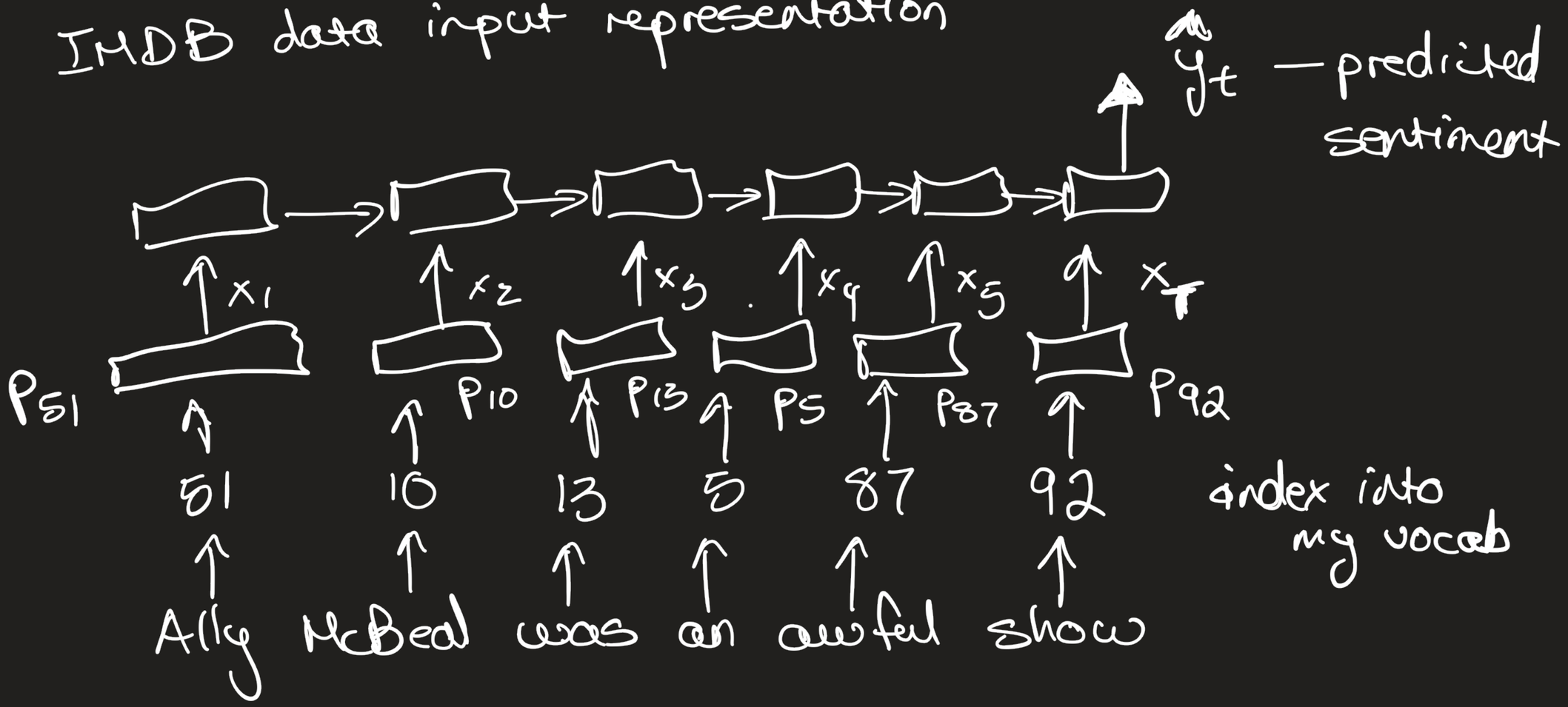
Learn via backprop all the parameters for these gates

LSTMs are very popular. GRUs were introduced to simplify the LSTM architecture:

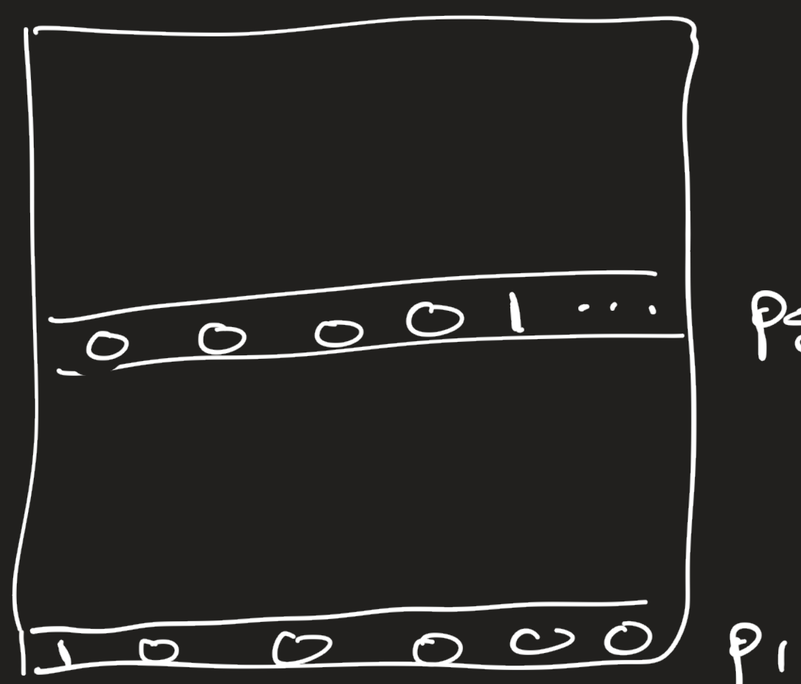
- have only a hidden state, no cell state
- only two gates: reset and update gates

Faster to train, but LSTMs are more popular.

IMDB data input representation



how to convert the integer encoding of words into vector encodings:

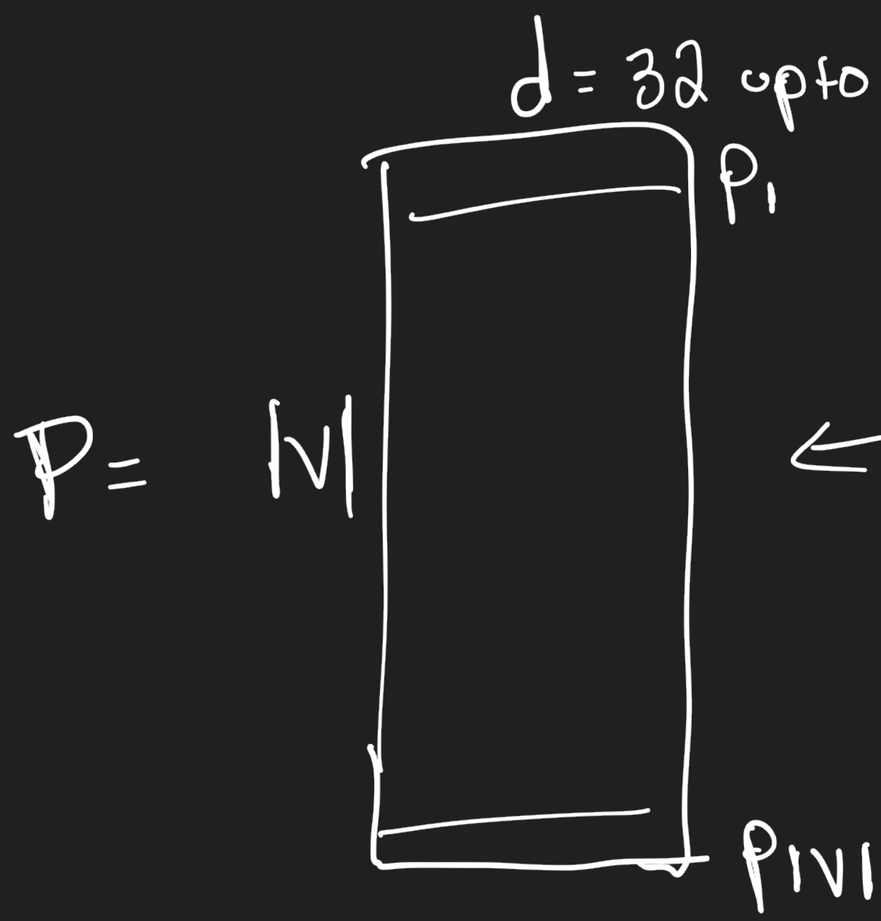


one-hot encoding is one possibility

problem: the encoding vectors are length $|V|$, very large $\approx 10K$

one hot encoding is inefficient

Better choice: fix an embedding dimension



and learn P via backpropagation

P is a parameter of our neural network