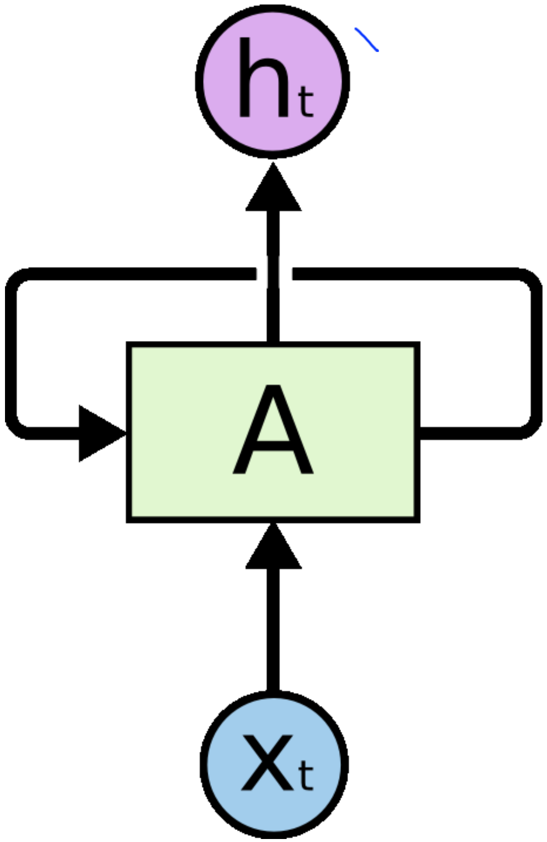
Lecture 13

RNN

Sequence data

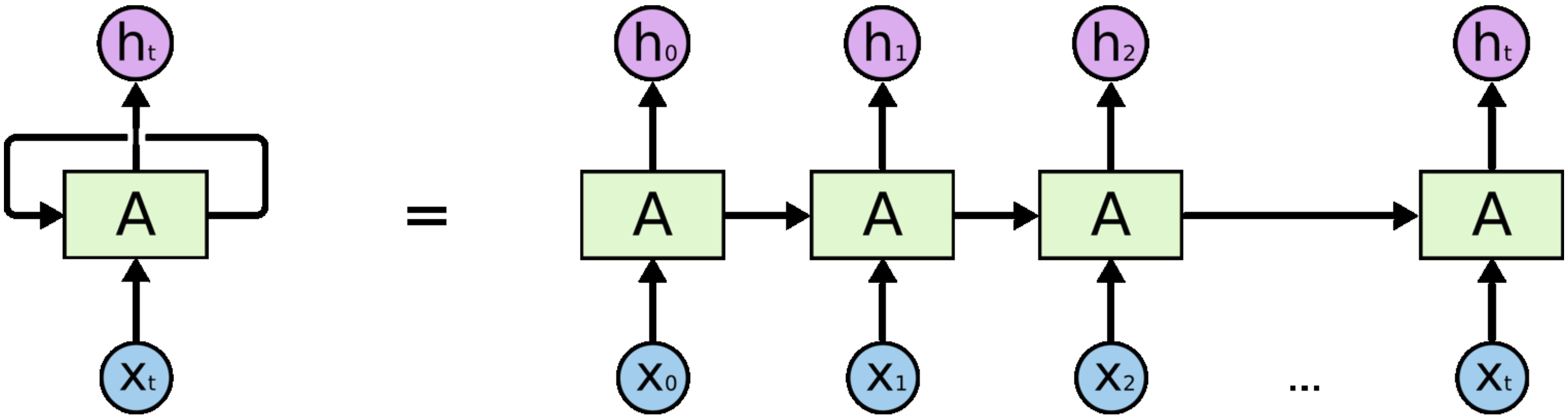
- We don't understand one word only
- We understand based on the previous words + this word. (time series)
- NN/CNN cannot do this



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Example

Speech recognition	 →	"The quick brown fox jumped over the lazy dog."
Music generation	∅ →	
Sentiment classification	"There is nothing to like in this movie." →	
DNA sequence analysis	AGCCCCTGTGAGGAACTAG →	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi? →	Do you want to sing with me?
Video activity recognition	 →	Running



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent Neural Network

We can process a sequence of vectors **x** by applying a recurrence formula at every time step:

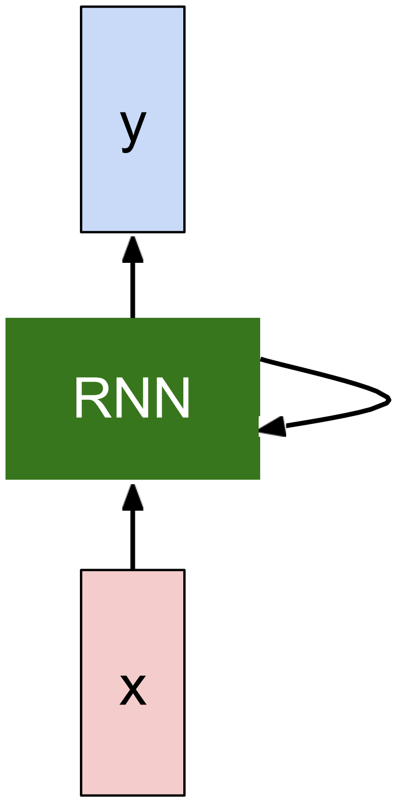
$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

old state

input vector at some time step

some function with parameters W



Recurrent Neural Network

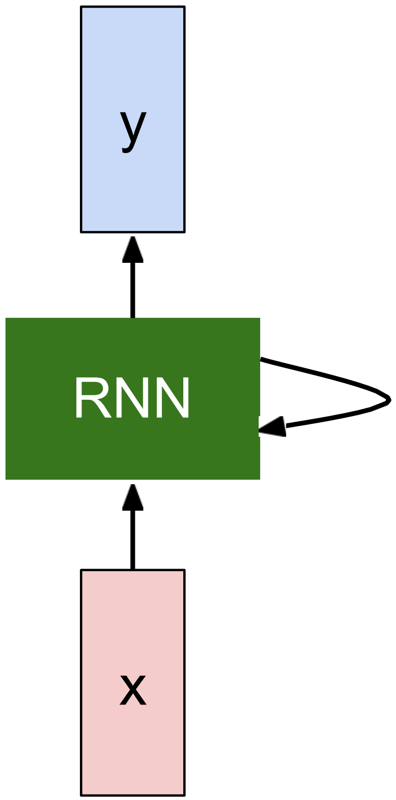
We can process a sequence of vectors **x** by applying a recurrence formula at every time step:

$$h_t = f_W(h_{t-1}, x_t)$$

Notice: the same function and the same set of parameters are used at every time step.

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

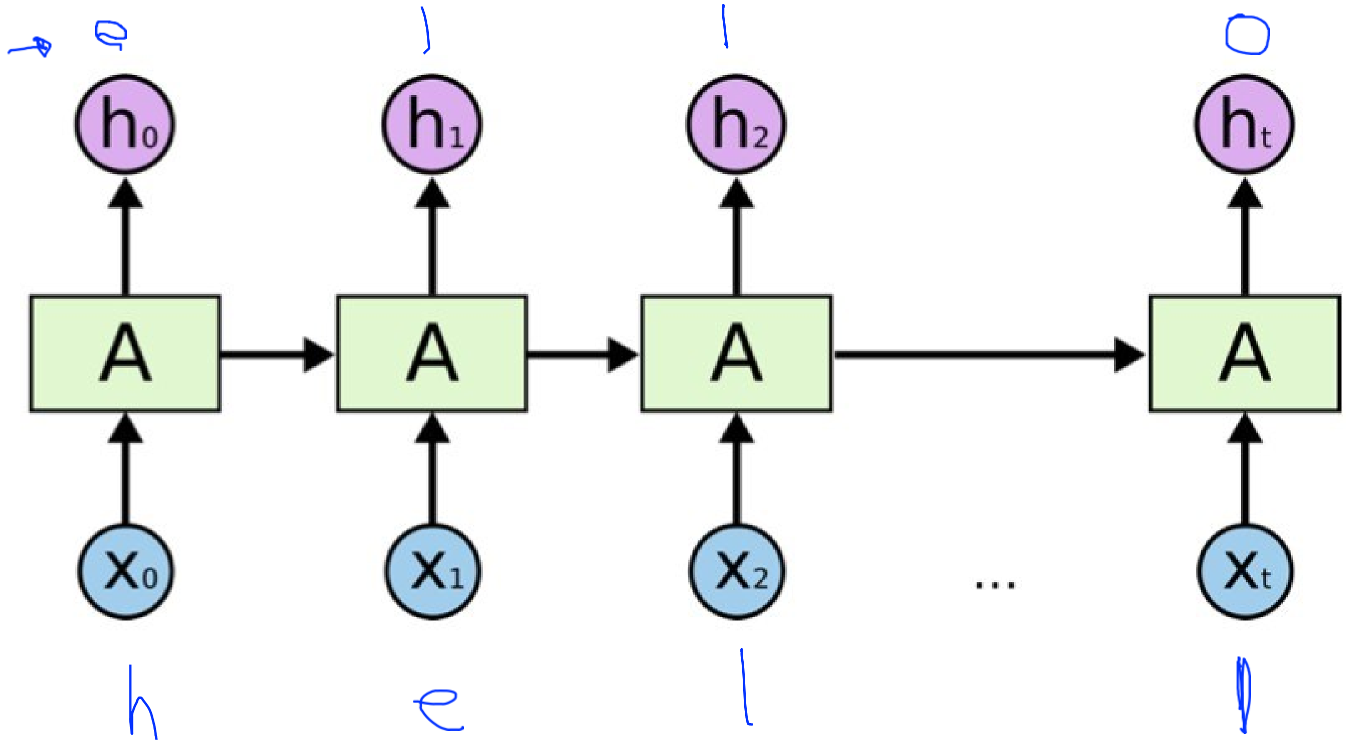


Fei-Fei Li & Andrej Karpathy

Character-level language model example

Vocabulary:
[h,e,l,o]

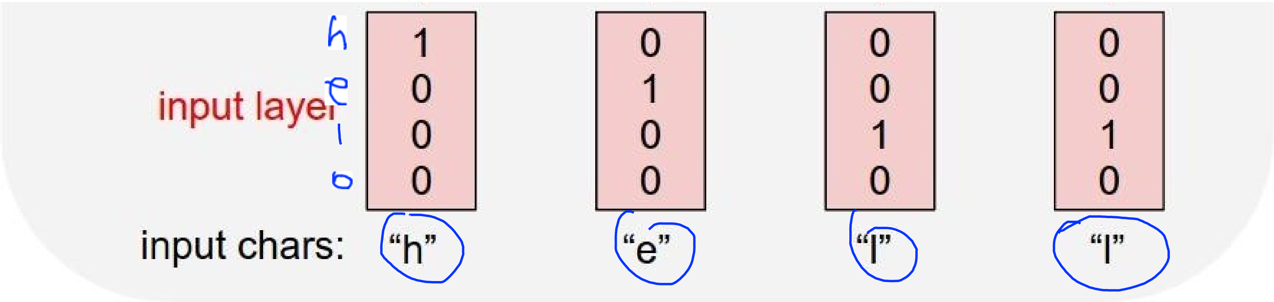
Example training sequence:
“hello”



Character-level language model example

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

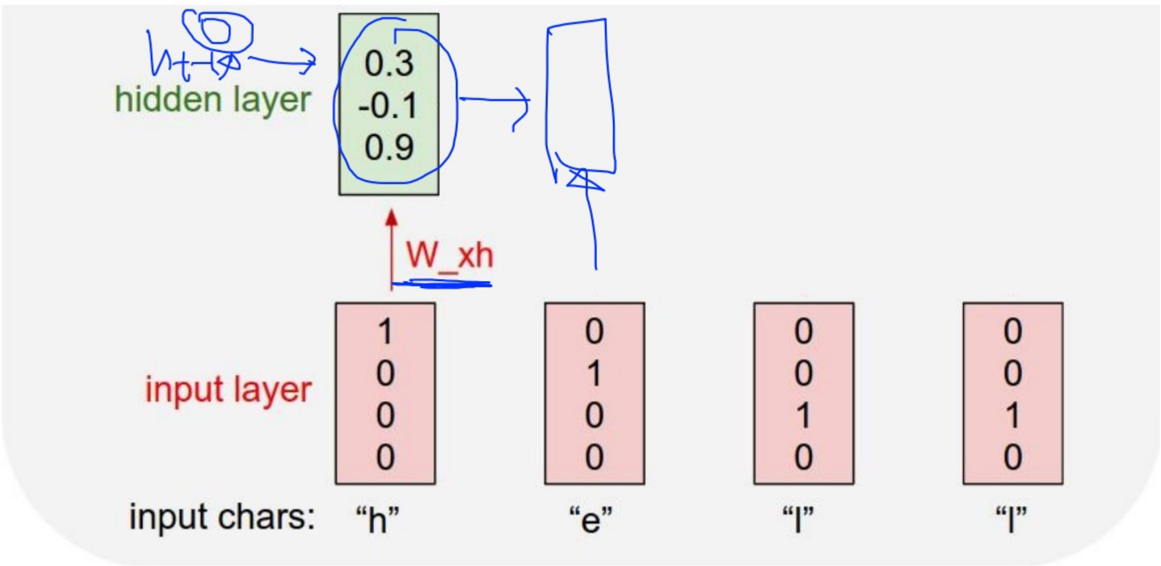


Character-level language model example

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

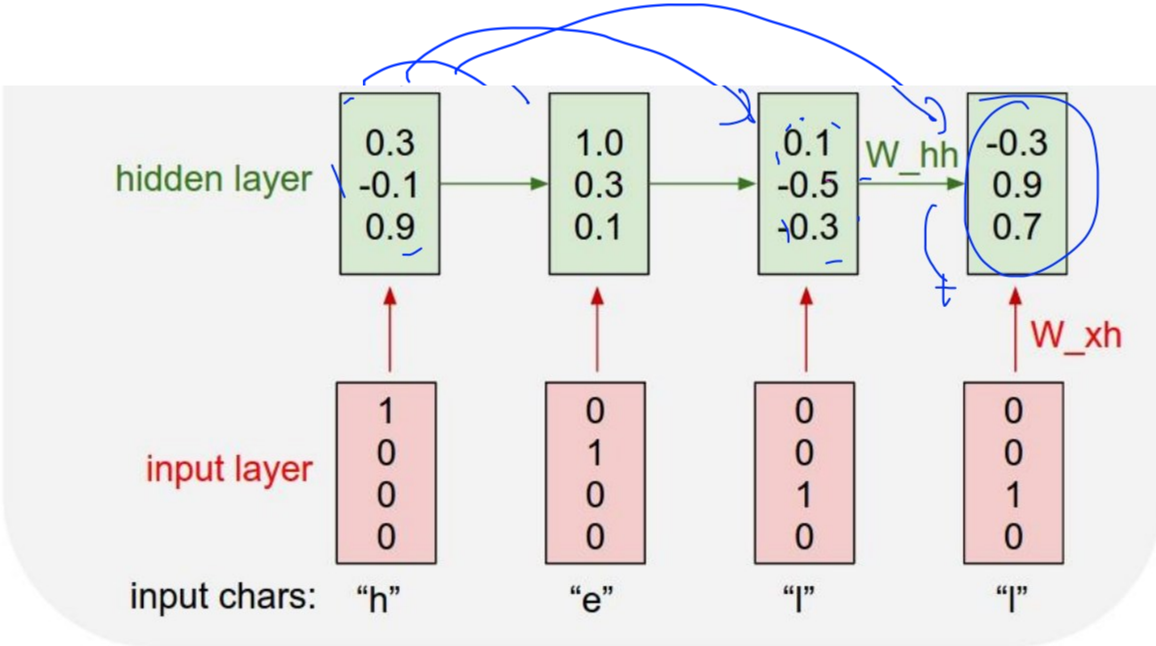


Character-level language model example

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

$$\boxed{h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)}$$



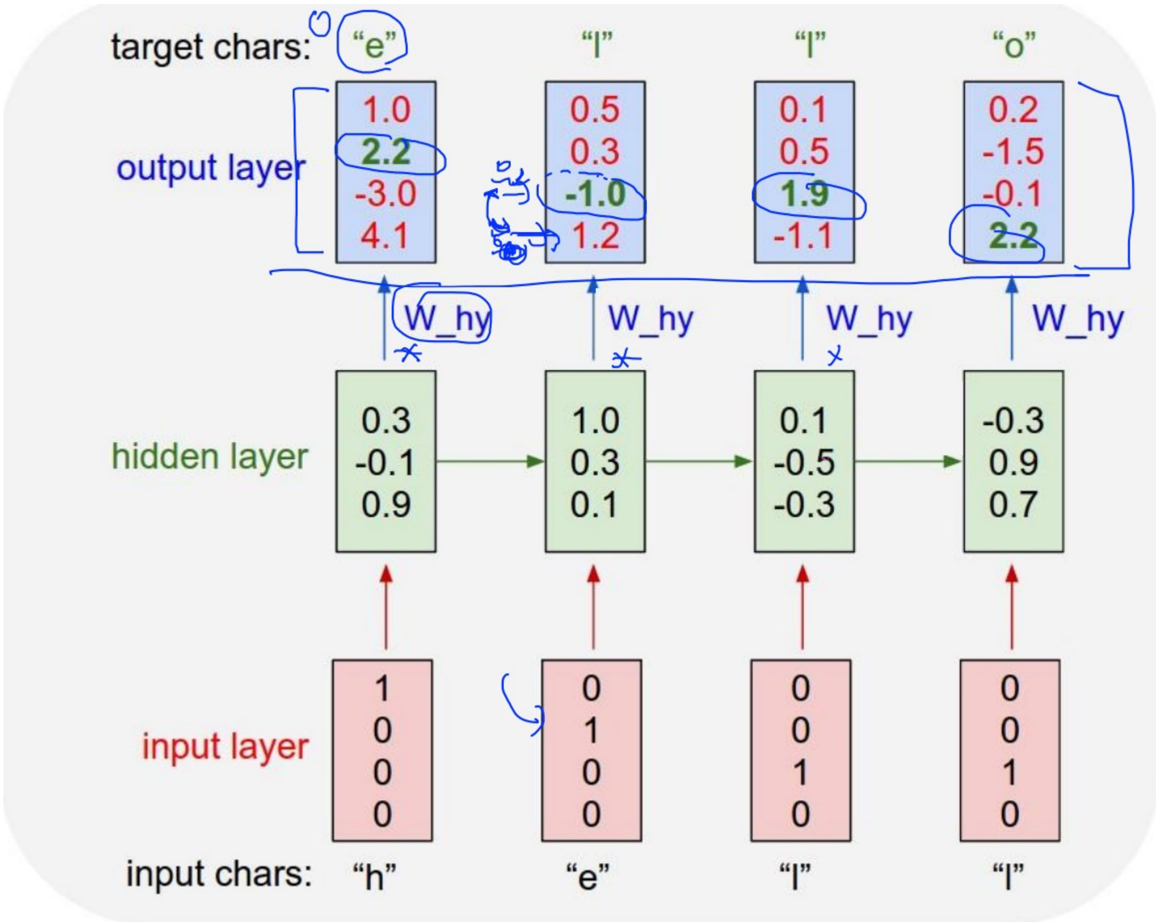
Fei-Fei Li & Andrej Karpathy

Character-level language model example

$$y_t = W_{hy} h_t$$

Vocabulary: [h,e,l,o]

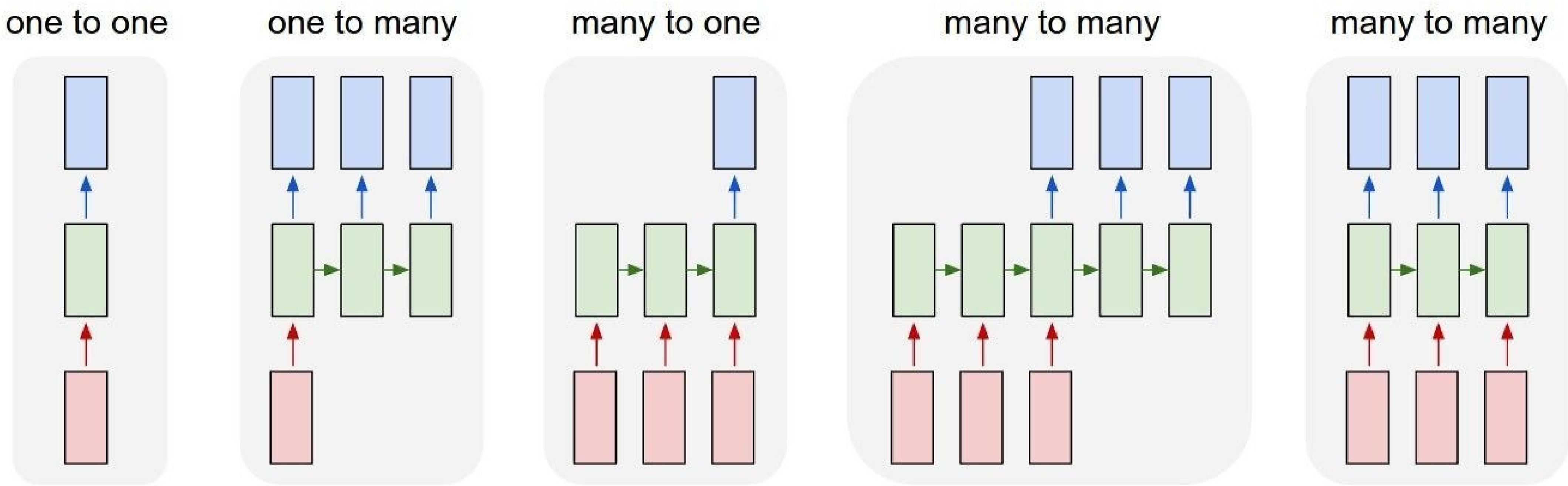
Example training sequence: **"hello"**



RNN applications

- https://github.com/TensorFlowKR/awesome_tensorflow_implementations
  - Language Modeling
  - Speech Recognition
  - MachineTranslation
  - Conversation Modeling/QuestionAnswering
  - Image/Video Captioning
  - Image/Music/Dance Generation

Recurrent Networks offer a lot of flexibility:



Vanilla Neural Networks

e.g. **Image Captioning**
image -> sequence of words

e.g. **Sentiment Classification**
sequence of words -> sentiment

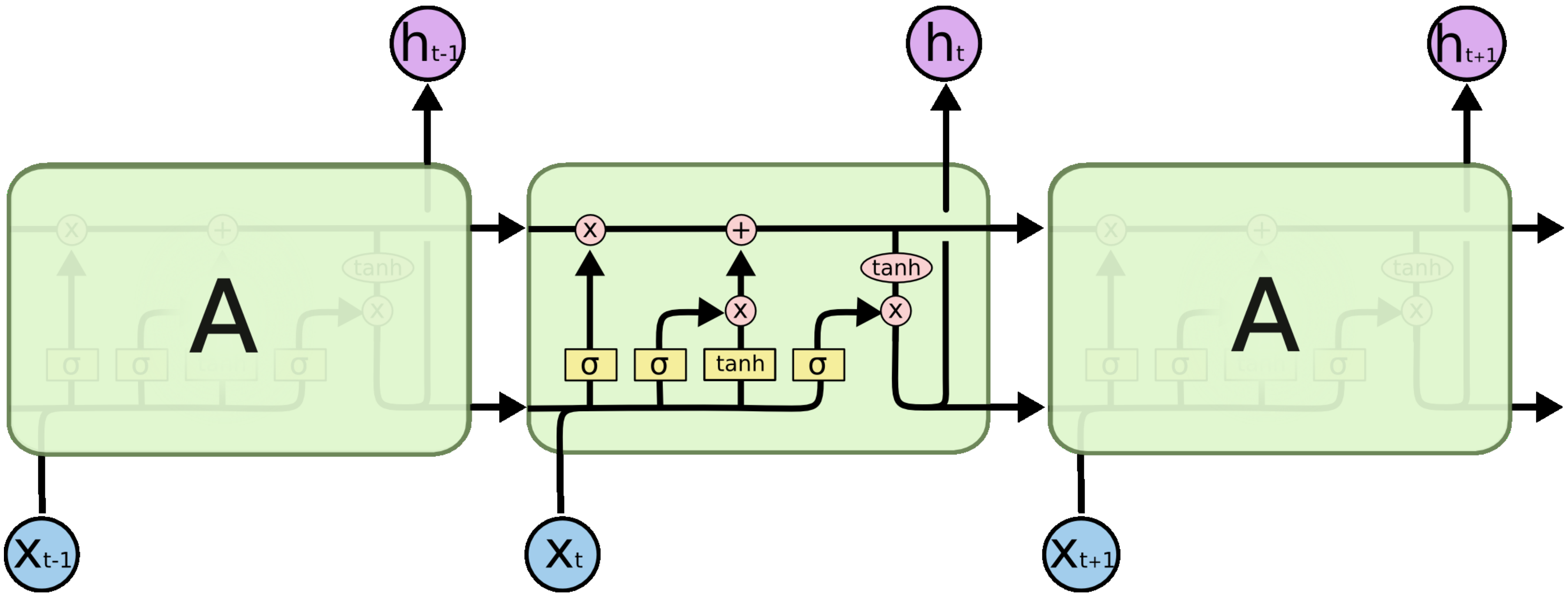
e.g. **Machine Translation**
seq of words -> seq of words

e.g. **Video classification on frame level**

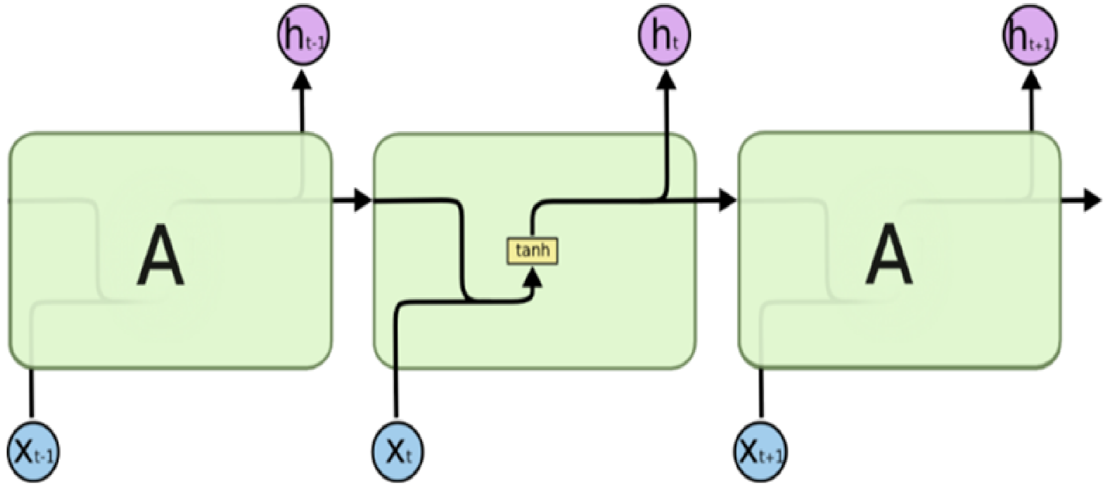
Fei-Fei Li & Andrej Karpathy

Understanding LSTM Networks

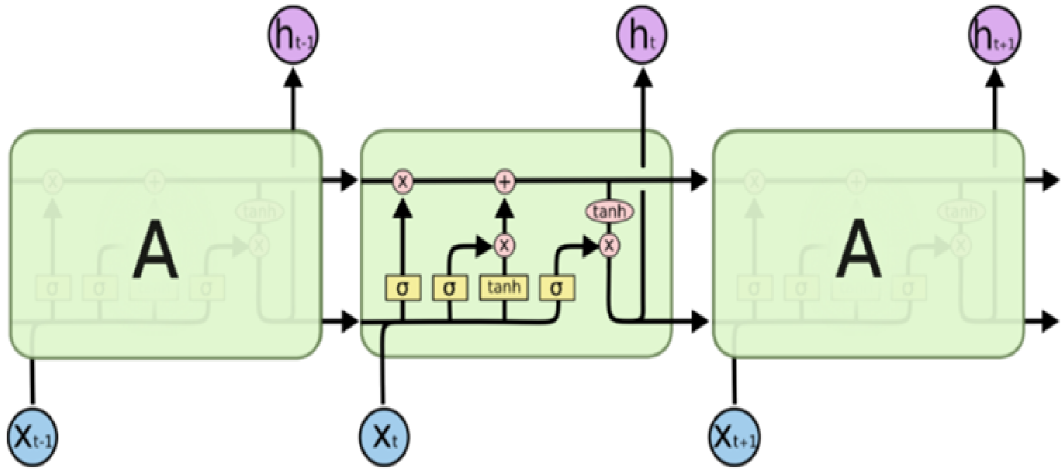
- https://colah.github.io/posts/2015-08-Understanding-LSTMs/



RNN vs LSTM



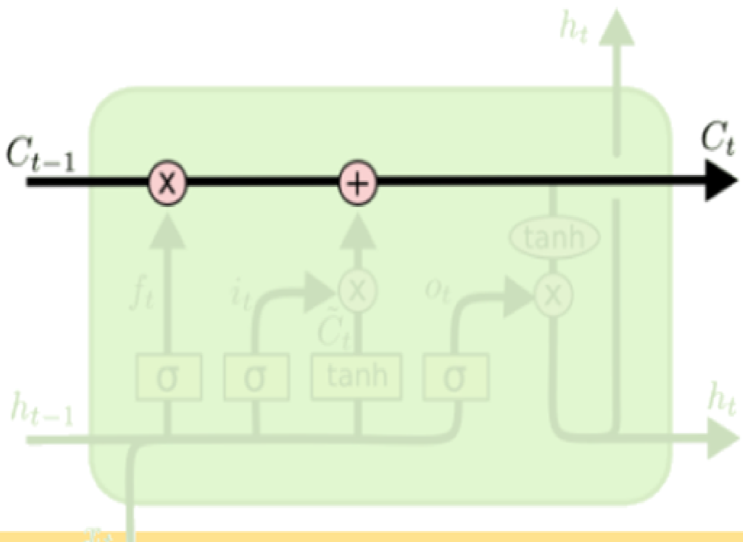
The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.

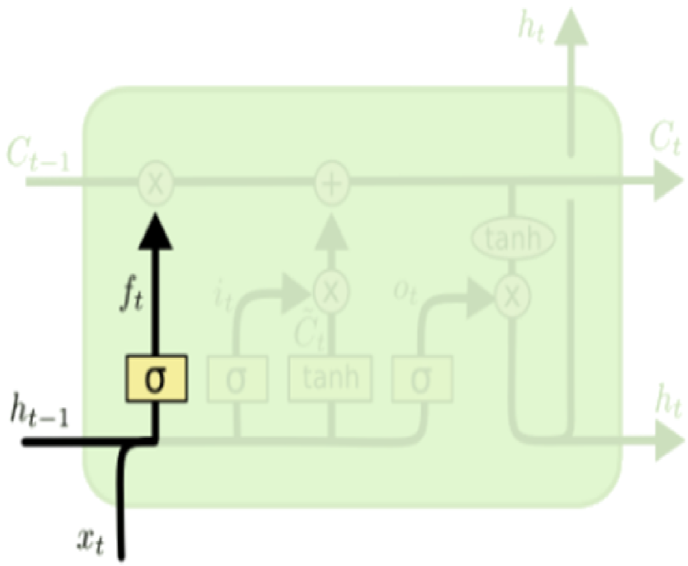
Core idea behind LSTM

- Key to LSTMs is the cell state
  - The horizontal line running through the top of the diagram
- LSTM can add or remove information to the cell state
- How? Through regulated structures called gates.
- LSTM has three gates to protect and control cell state



Steps

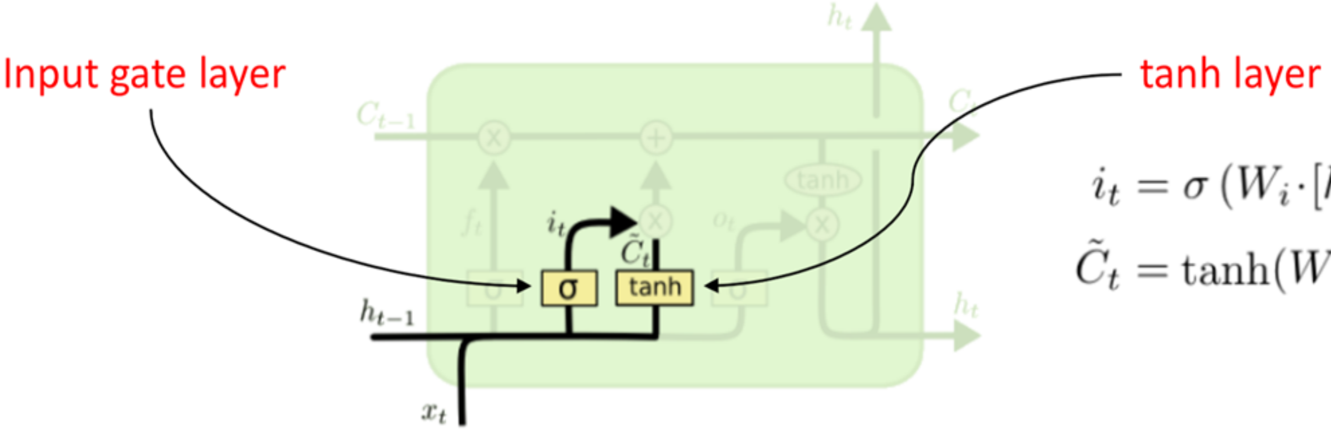
- Forget gate layer decides what information will be thrown away
- Looks at h_{t-1} and x_t and outputs a number between 0 and 1
- 1 represents completely keep this, 0 represents completely get rid of this
- Example: forget the gender of the old subject, when we see a new subject



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

Steps

- Next step: decides what new information will be stored in the cell state
- Two parts –
  - A sigmoid layer (input gate layer): decides what values we'll update
  - A tanh layer: creates a vector of new candidate values, $\widetilde{C}_t$
- Example: add the gender of the new subject to the cell state
  - Replace the old one we're forgetting

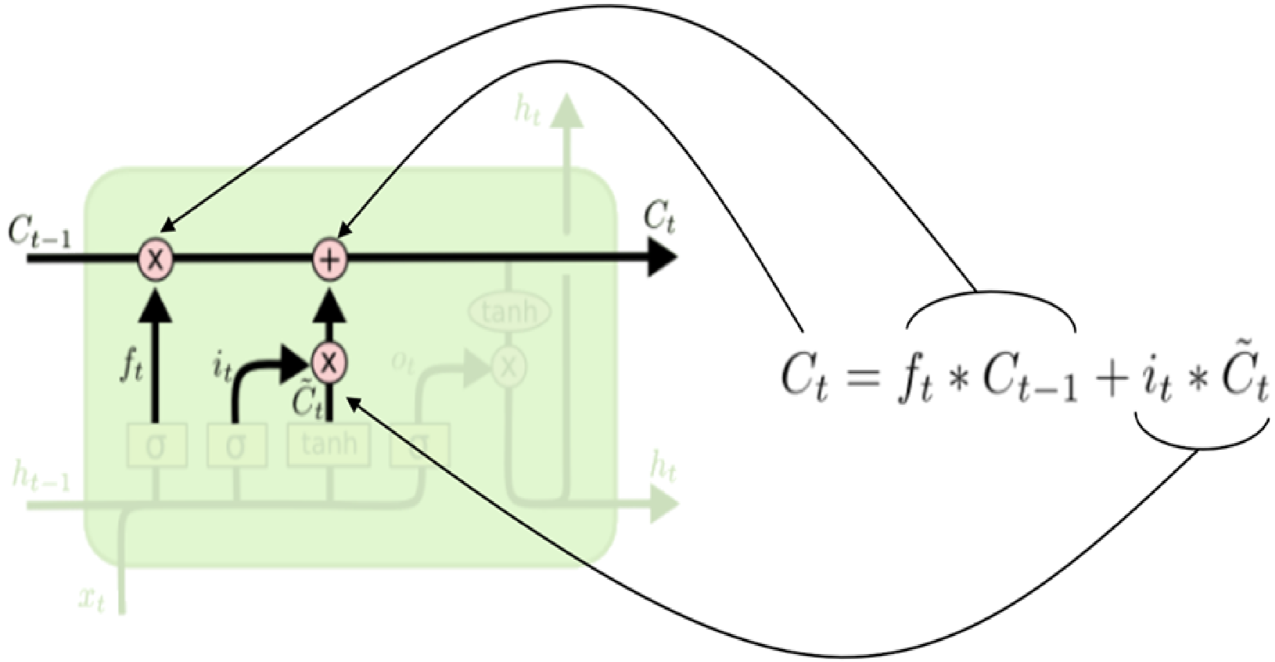


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

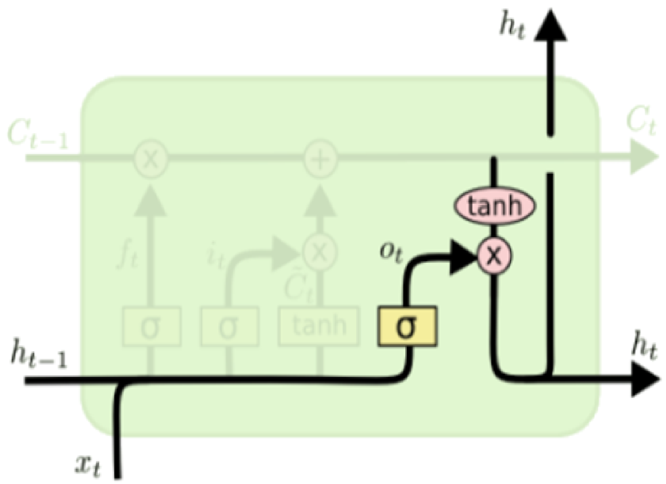
Steps

- Next step: update old state by C_{t-1} into the new cell state C_t
- Multiply old state by f_t
  - Forgetting the things we decided to forget earlier
- Then we add $i_t * \widetilde{C}_t$



Steps

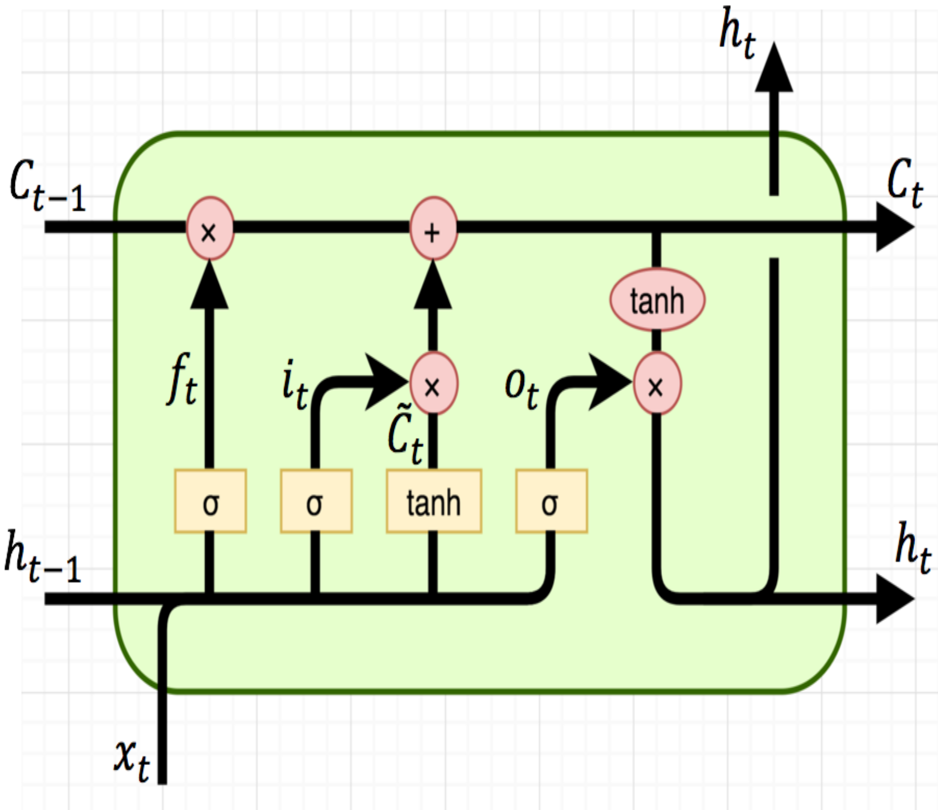
- Final step: decide what we're going to output
- First, we run a sigmoid layer
  - Which decides what parts of the cell state we're going to output
- Then, we put the cell state through tanh and multiply it by the output of the sigmoid gate



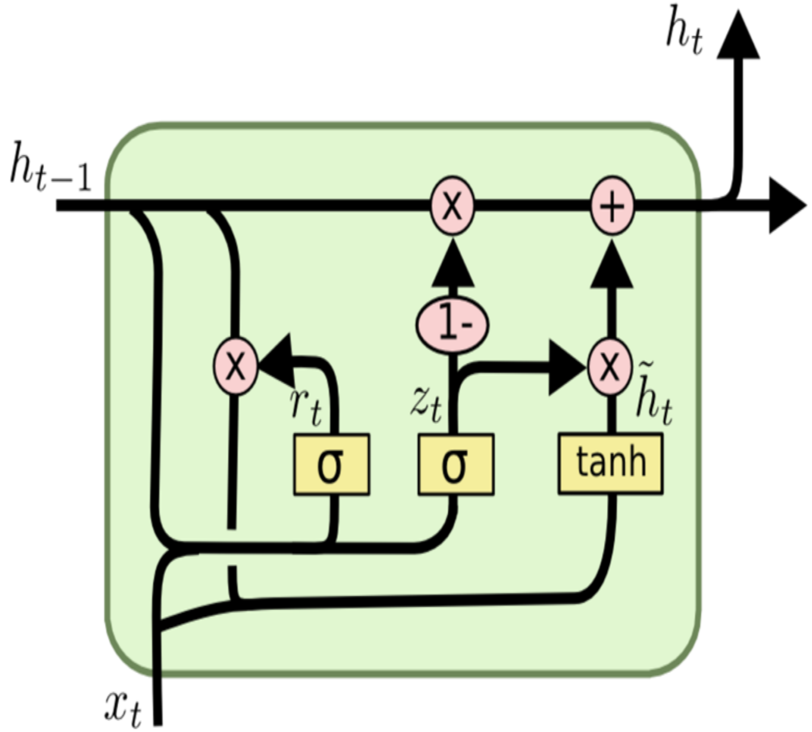
$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

LSTM vs GRU



(a) Long Short-Term Memory



(b) Gated Recurrent Unit

Conclusion

- Vanilla RNNs are simple but don’t work very well
- LSTM is a big step in what we can accomplish with RNNs
  - Backward flow of gradients in RNN can explode or vanish. Exploding is controlled by gradient clipping and vanishing is controlled with additive interactions (LSTM)
- GRU’s are faster to train and need fewer data to generalize.
- When there is enough data, an LSTM’s greater expressive power may lead to better results
- Better understanding (both theoretical and empirical) is needed