Natural Language Processing

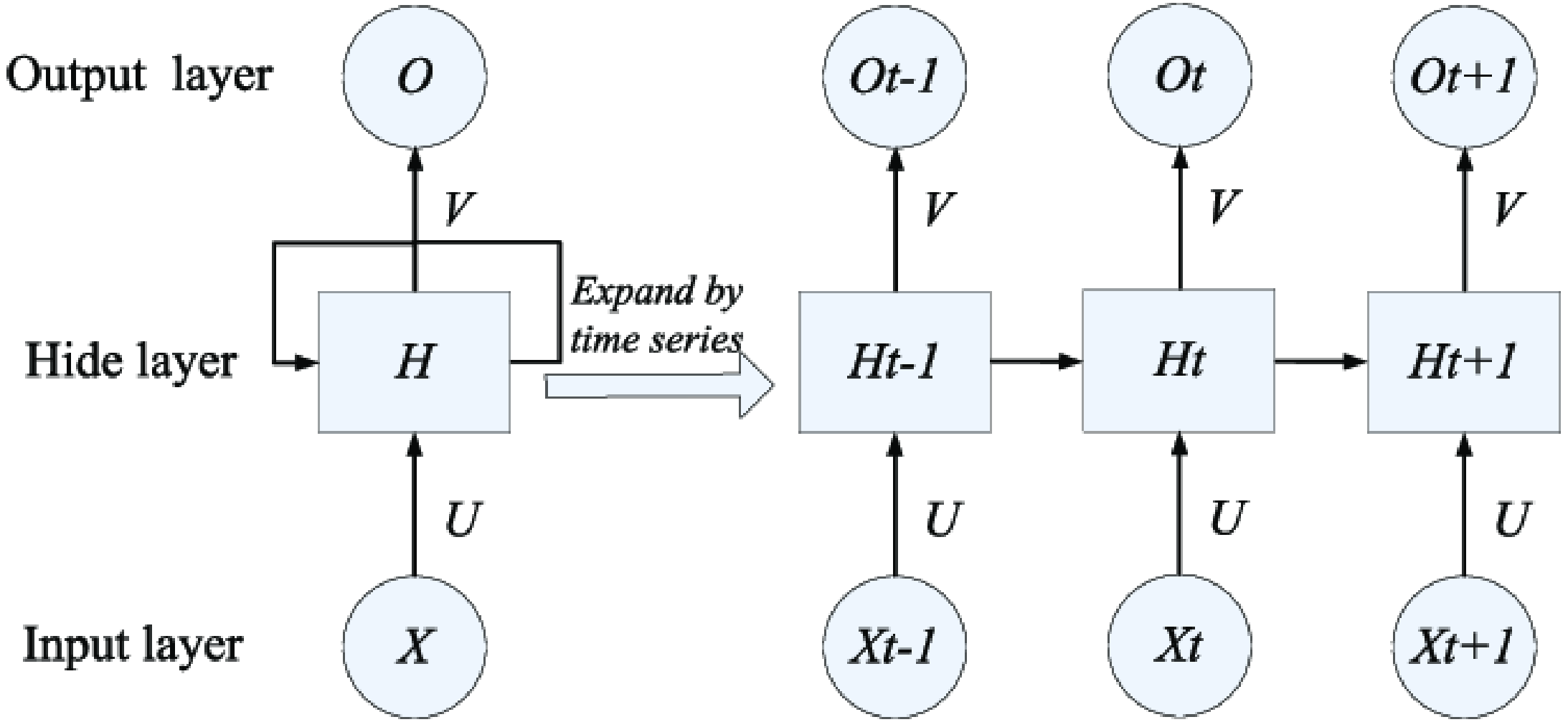
AIGC 5501

LSTMs

Instructor: Ritwick Dutta
Email: ritwick.dutta@humber.ca

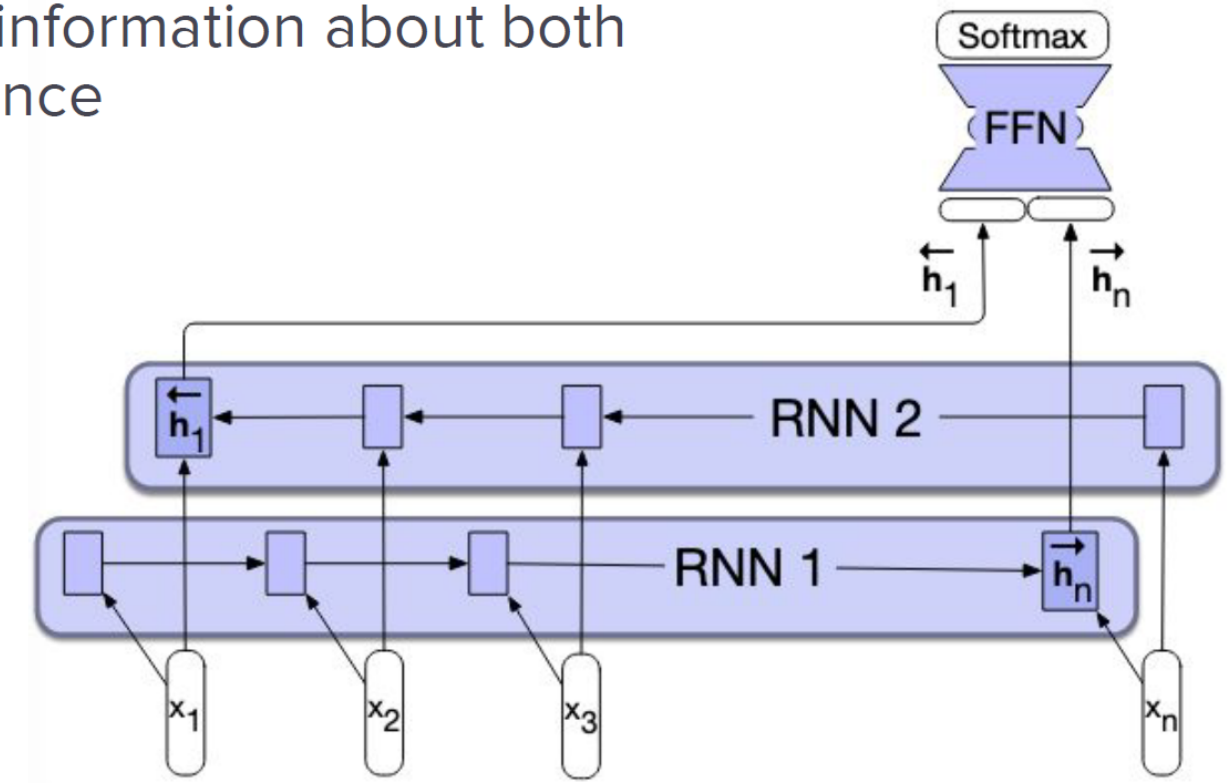
Last Class

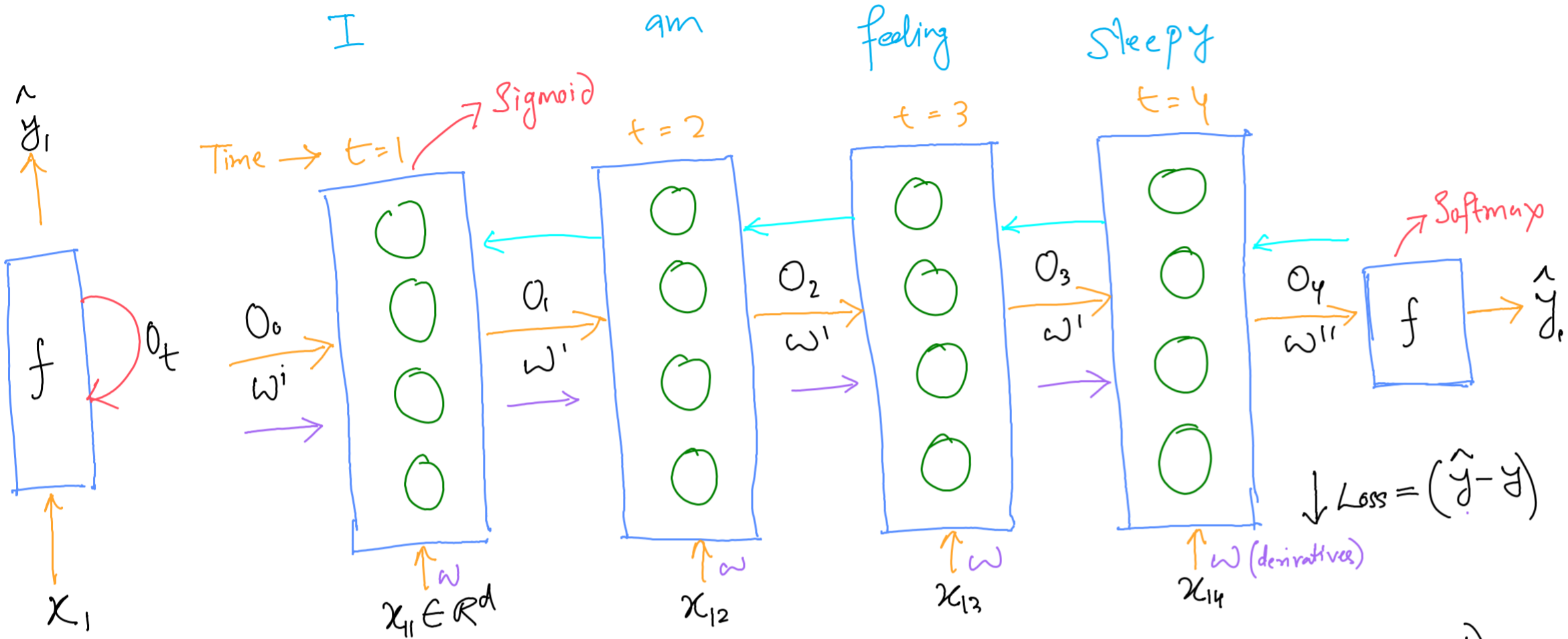
RNN



Bidirectional RNNs

- Especially helpful for sequence classification
  - h_n reflects more information about the **end** of the sentence, h_1, about the beginning
  - biRNNs/bidiRNNs allow capturing information about both **beginning** and **end** of input sequence
  - Normally use concatenation





Last night I was watching movie. I took a bath and ate dinner late and then reading book. Now, I am feeling _____.

$$O_1 = f(x_{11} w + O_0 w')$$
$$O_2 = f(x_{12} w + O_1 w')$$
$$O_3 = f(x_{13} w + O_2 w')$$
$$O_4 = f(x_{14} w + O_3 w')$$

Problems

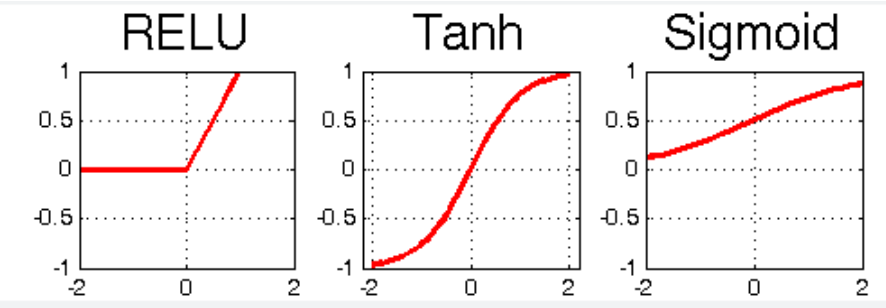
- RNNs (in theory) were supposed to allow us to make use of information from **arbitrarily long** sequences
- In practice, this doesn't usually happen 😢

Issue for RNNs: Long-distance information

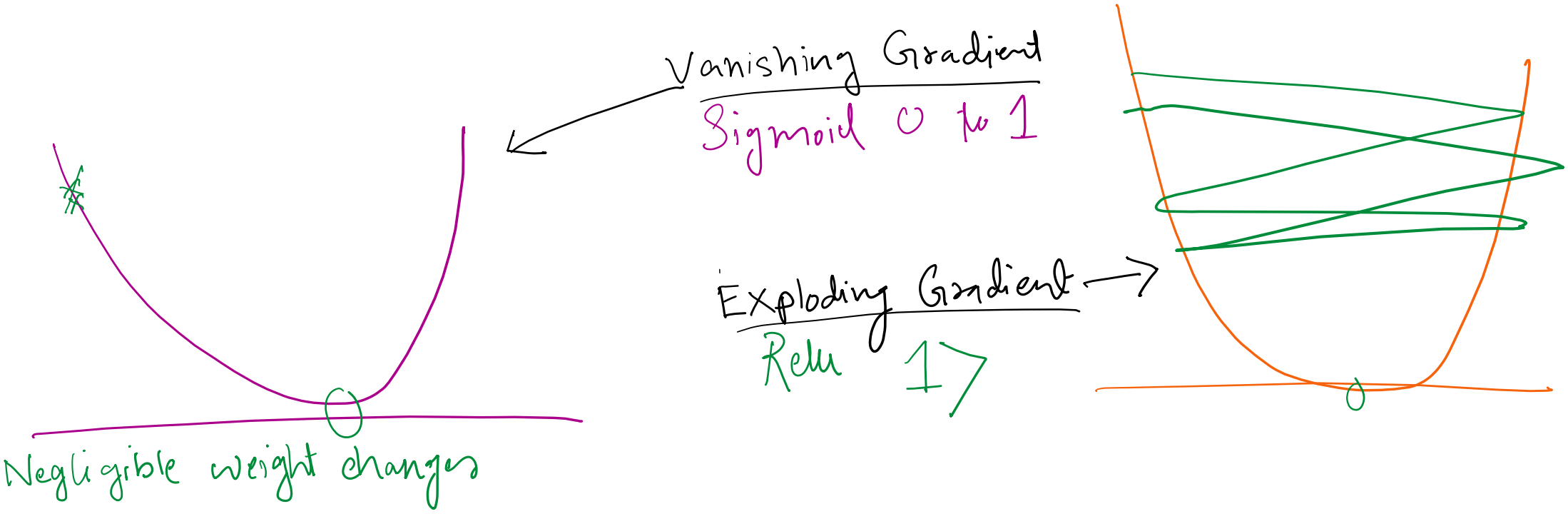
- Hard to encode in RNNs, **BUT** critical for many NLP tasks.
- E.g. Language Modeling

"When she tried to print her **tickets**, she found that the printer was out of toner. She went to the stationery store to buy more toner. It was very overpriced. After installing the toner into the printer, she finally printed her _____"

Problem of vanishing gradient



- Backprop for RNNs subjects hidden layers to repeated dot products
  - Dependent on length of sequence (recall Backpropagation through time)
- Can drive gradients to 0

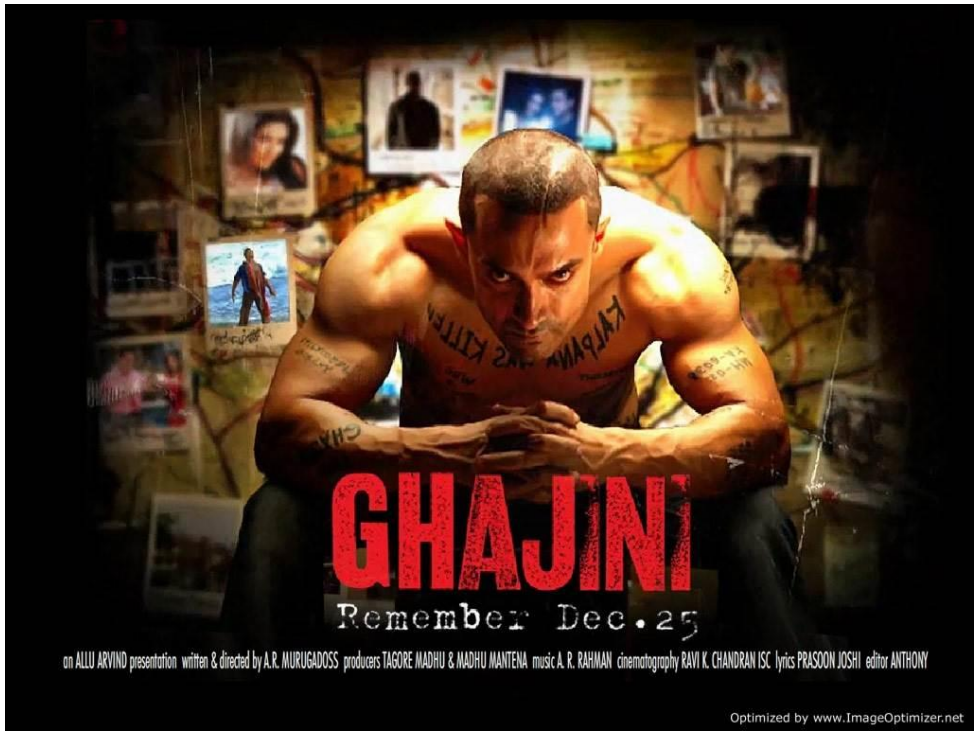


Impact on RNN

"When she tried to print her tickets, she found that the printer was out of toner. She went to the stationery store to buy more toner. It was very overpriced. After installing the toner into the printer, she finally printed her _____"

- RNN needs to **model the dependency** between "tickets" early on and target "tickets" at end
- But if gradient is small, the model **can't learn this dependency**
- Model can't predict similar long-distance dependencies at inference / test time!

RNN suffers from short-term memory problem



Problem of vanishing gradient

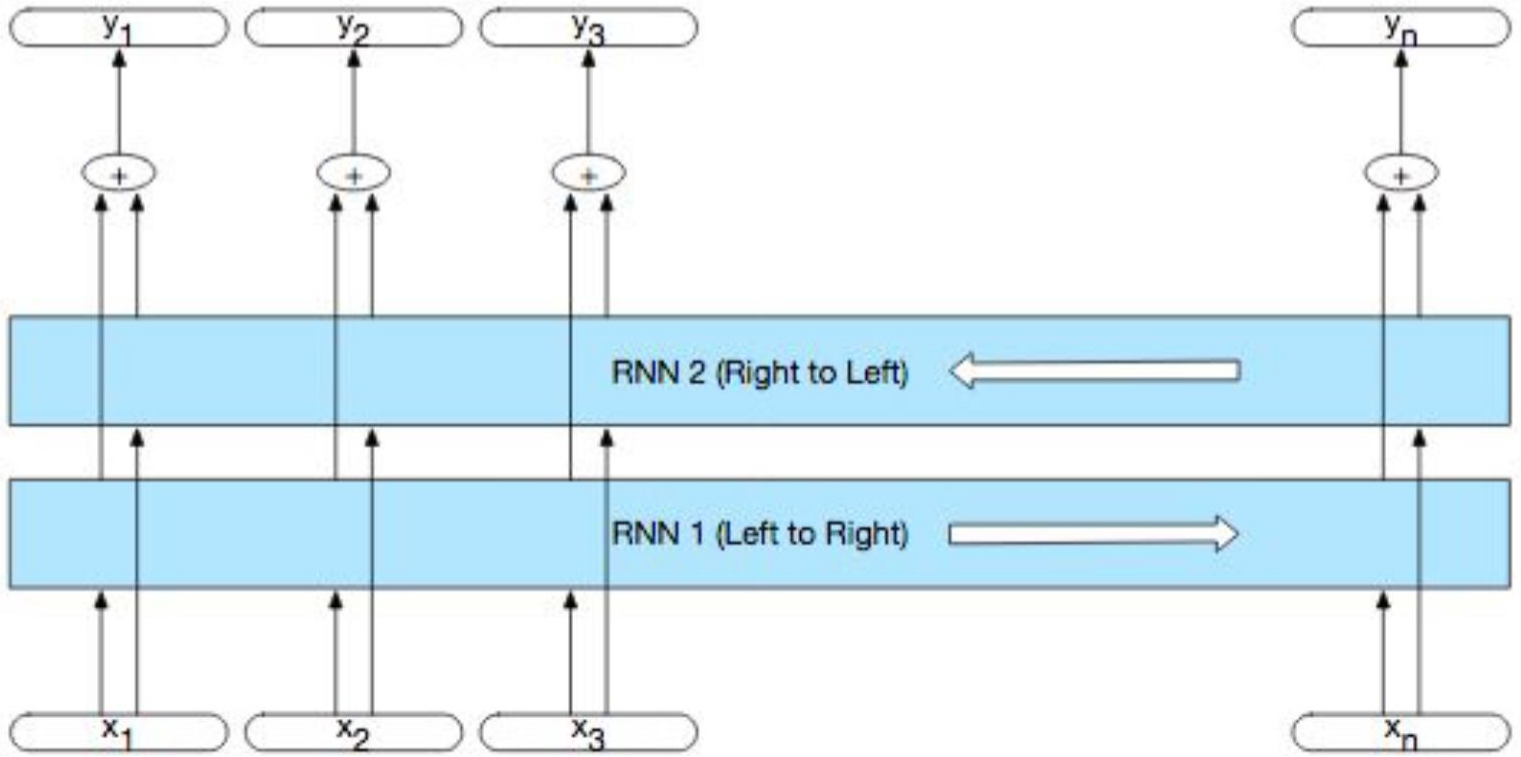
- Backprop for RNNs subjects hidden layers to repeated dot products
  - Dependent on length of sequence (recall Backpropagation through time)
- Can drive gradients to 0
- **Solution**:
  - more complex network architectures
  - explicitly manage the maintaining of context over time
  - treat context as a kind of memory unit

ELSE

Network needs to **forget** information that is no longer useful and **remember** information as needed for later decisions

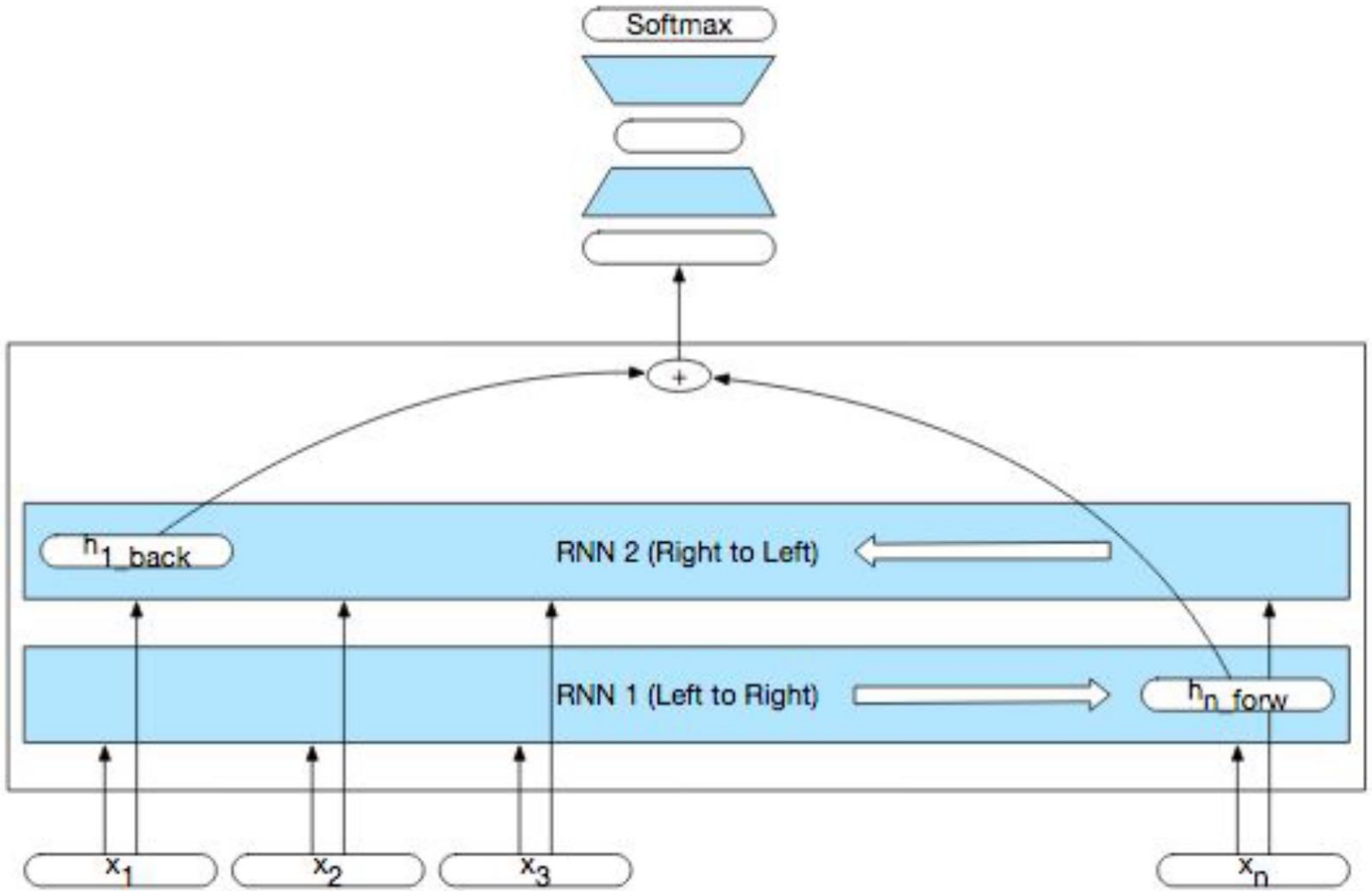
Managing the Context

- In practice, it is difficult for RNNs to use information distant from the current point of processing
- **Bi-directional RNNs** are one attempt



Managing the Context

- Bi-directional RNNs for sequence classification handle it a bit differently



Many many RNN architectures address this

- Gated Recurrent Network (GRU) [Cho et al. EMNLP/CoRR2014, Chung et al. CoRR 2014]
- Long Short-Term Memory [Hochreiter & Schmidhuber, NeurComp 1997]
- Memory Networks [Weston et al., ICLR 2015]
- Dynamic Memory Network (DRM) [Kumar et al., ICML 2016]
- RNNs with Attention
- .....

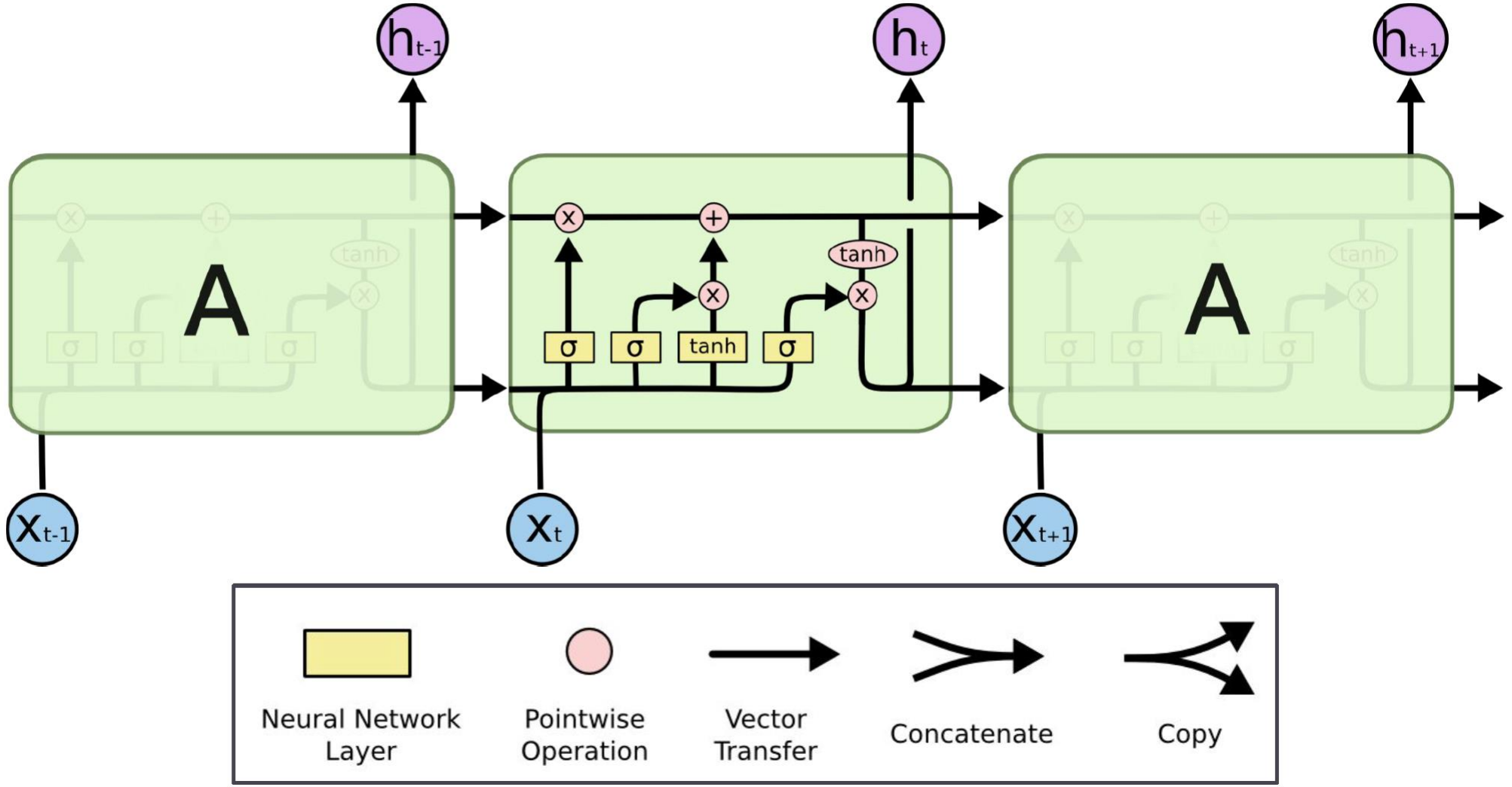
Long Short-Term Memory RNNs (LSTMs)

Key idea: Improve the context management within RNNs

- Removing information no longer needed from context
- Adding information likely to be needed for later decision making
- Learning how to balance the two

Long Short-Term Memory RNNs (LSTMs)

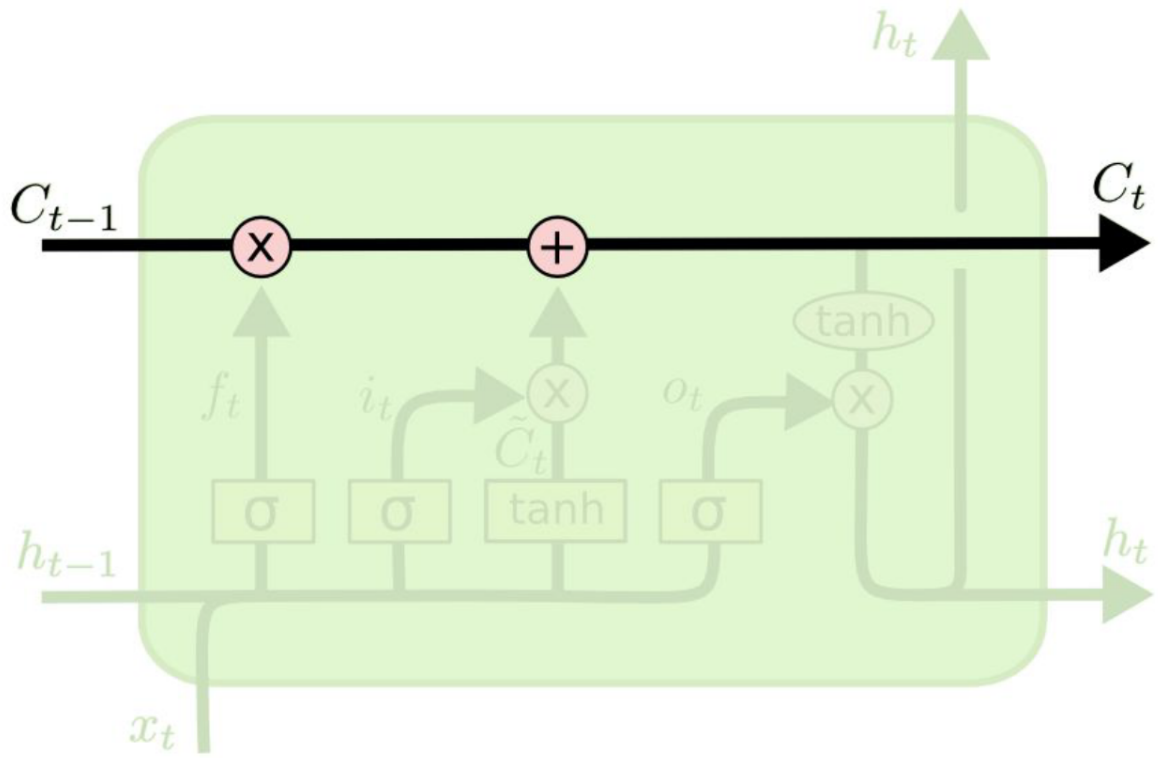
We'll look at the internals of this:

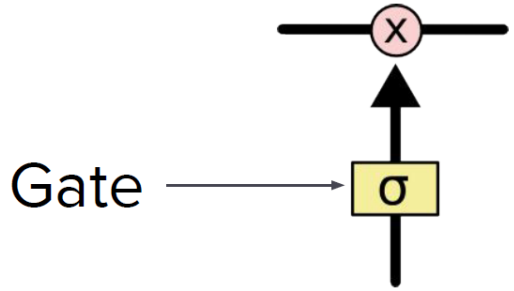


Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

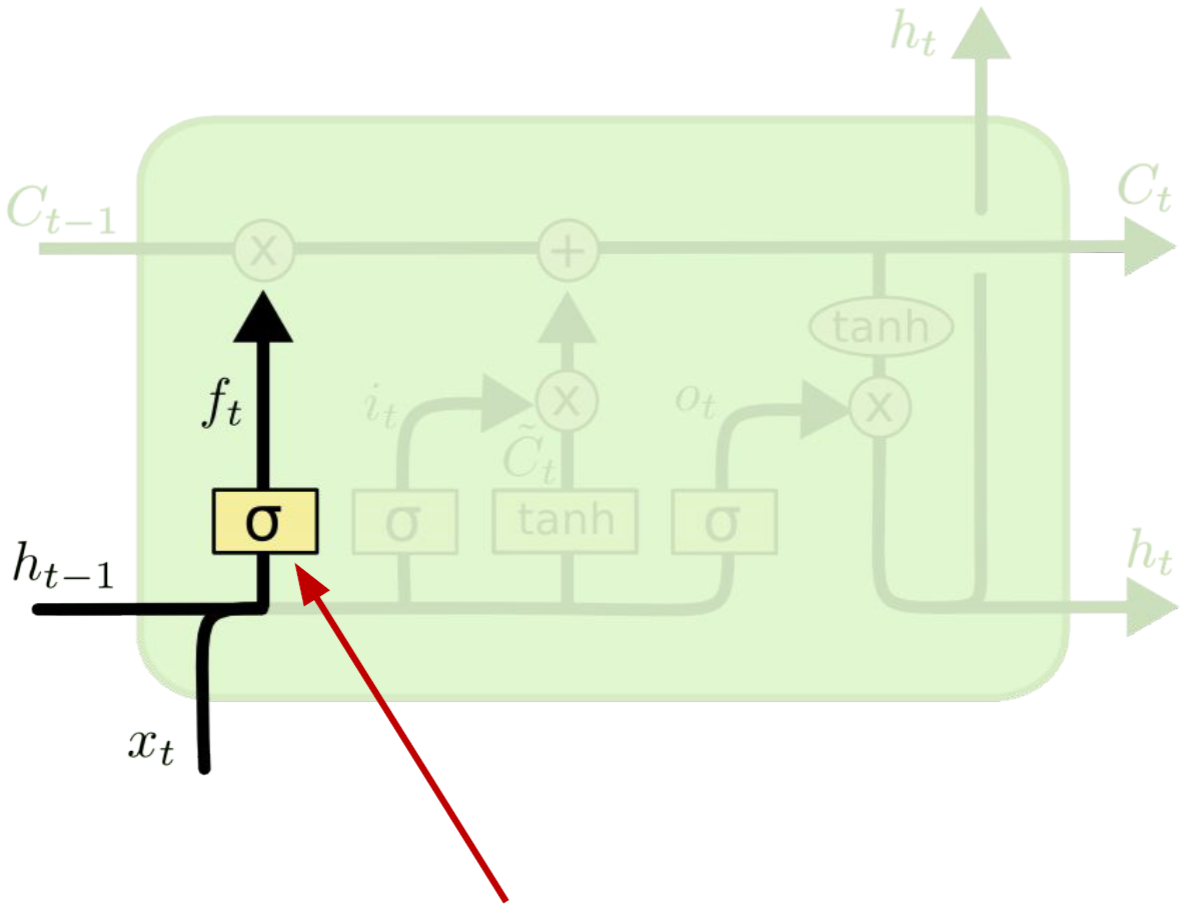
Key component: Cell state *C*

- Cell state acts like a conveyor belt
  - Easy to stay the same from t-1 to t
- Can *add* or *forget* information via 3 gates
  - How much old or new information do we pass on?
- Sigmoid acts as binary masking function





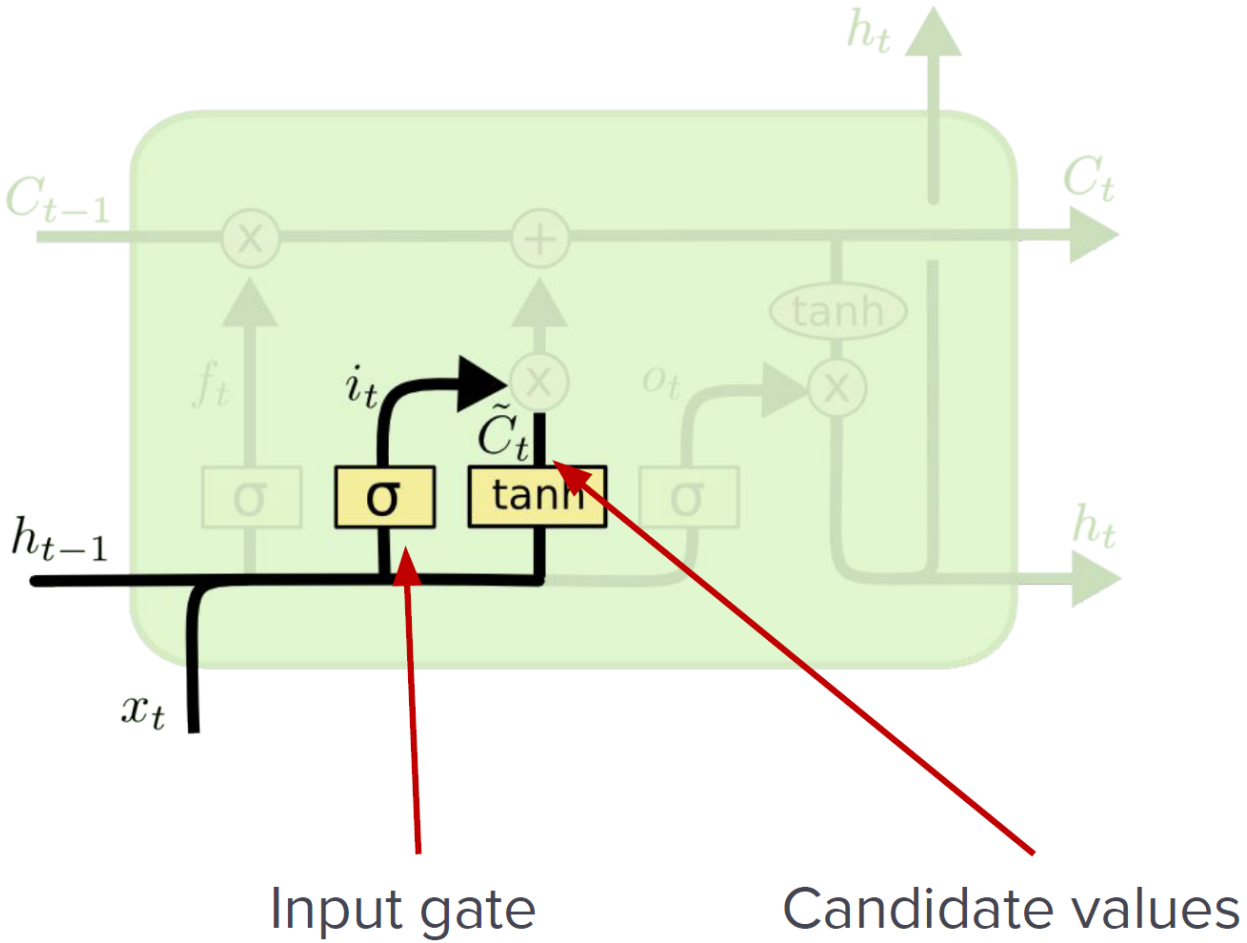
Step 1: Decide what to forget and remember



Forget gate

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

Step 2: What new information do we want to remember?



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

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

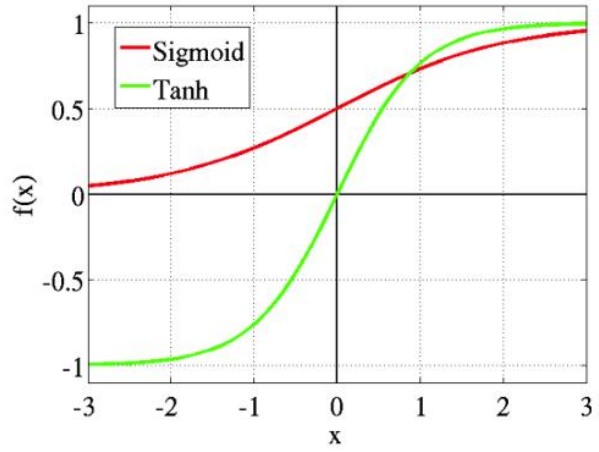
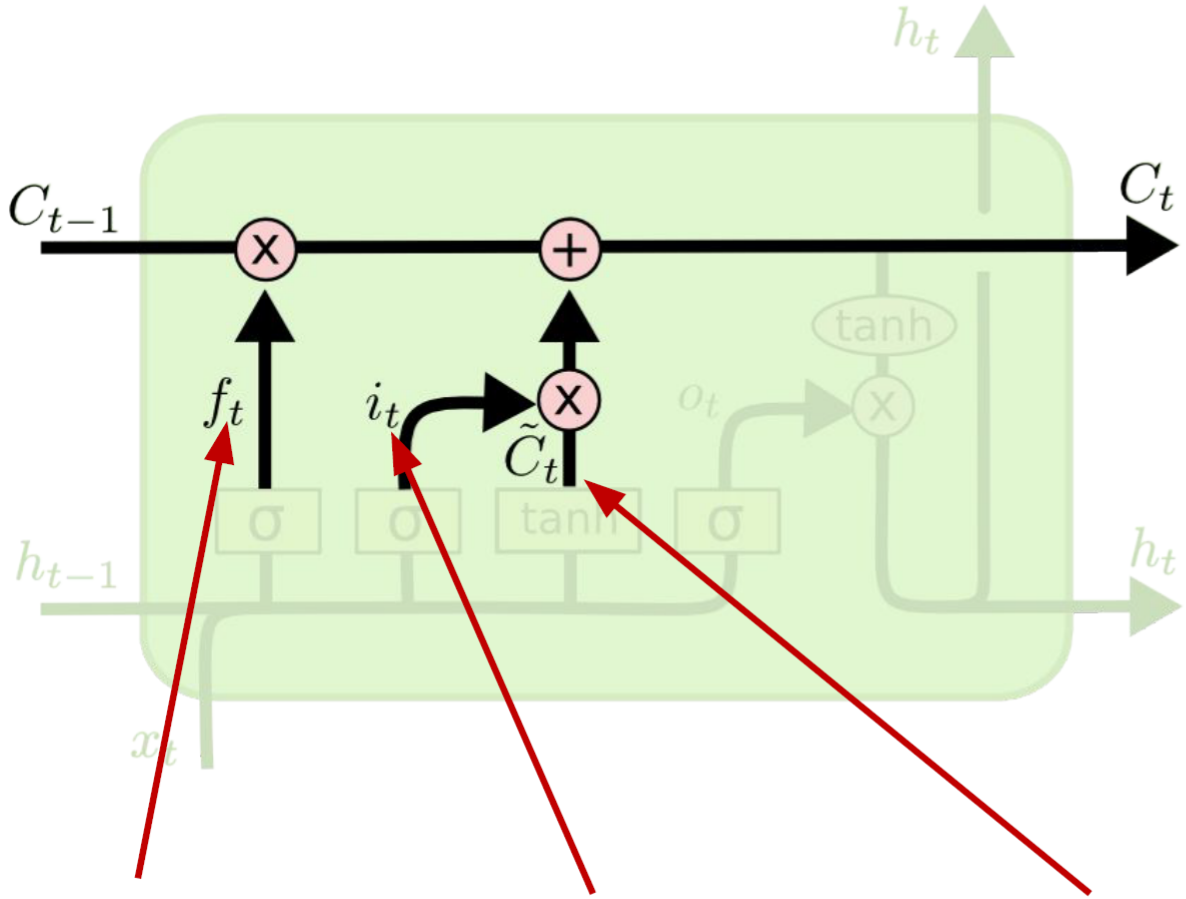


Fig: tanh v/s Logistic Sigmoid

Step 3: Update Cell state C



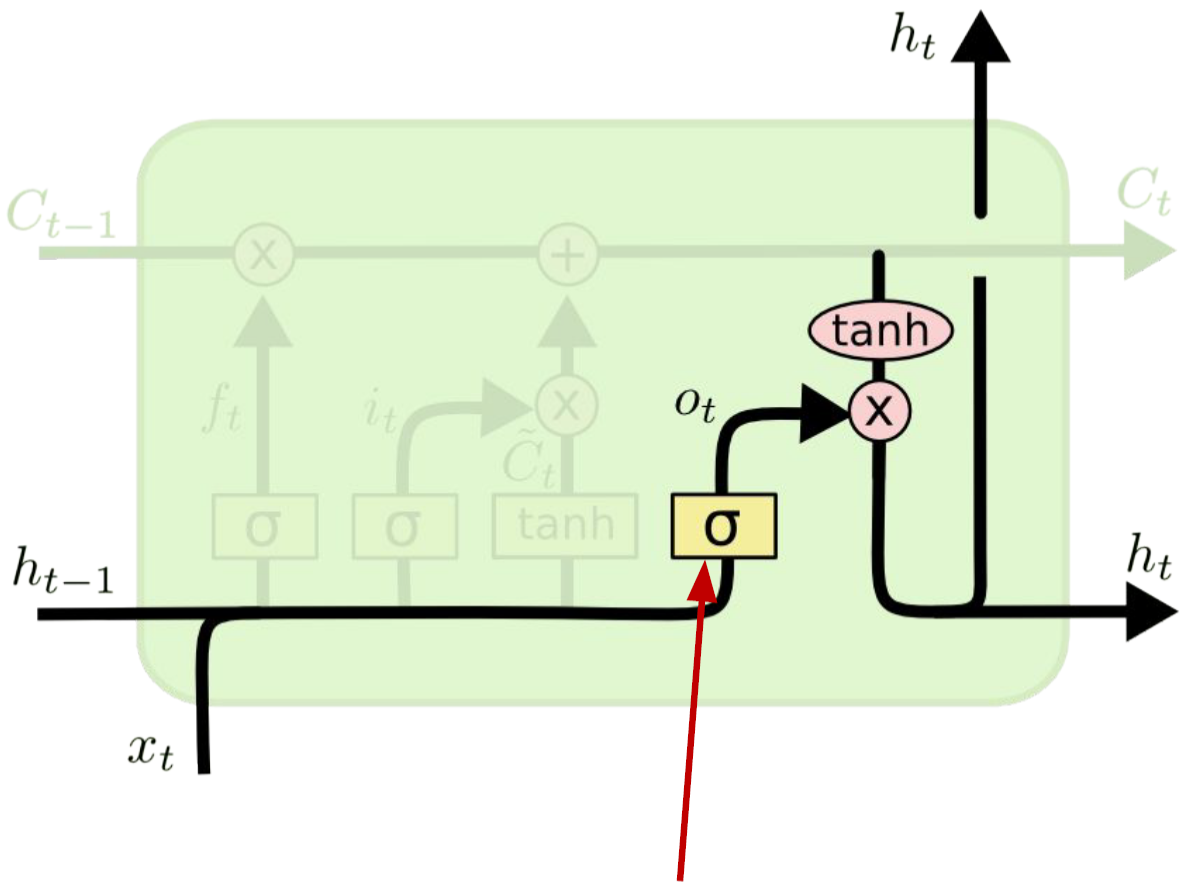
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Forget gate

Input gate

Candidate values

Step 4: Update hidden / What do we output?



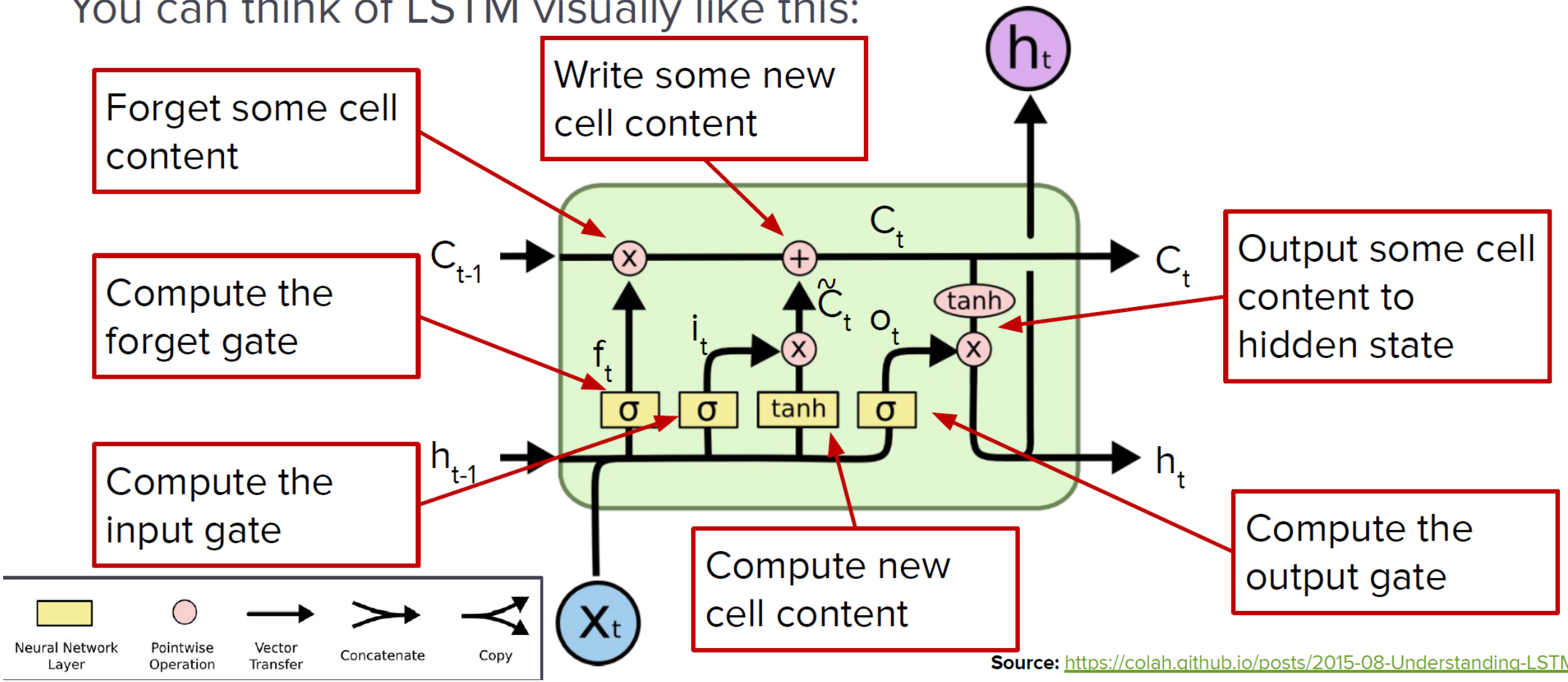
Output gate

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

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

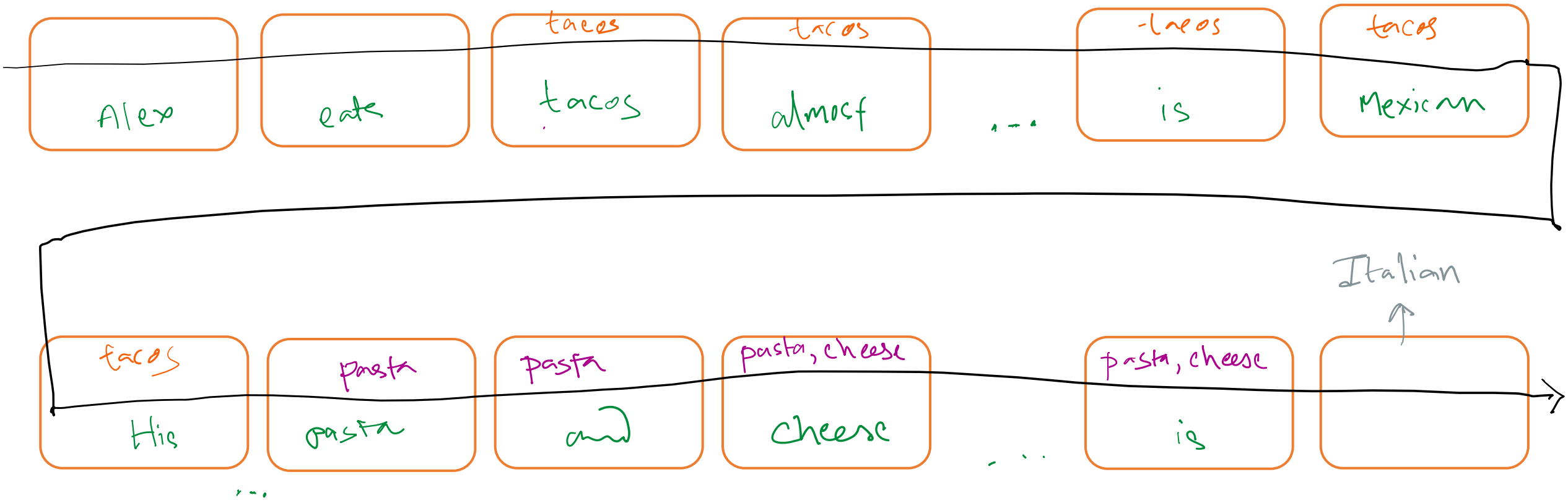
Long Short-Term Memory RNNs (LSTMs)

You can think of LSTM visually like this:



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

Alex eats **tacos** almost everyday, it shouldn't be hard to guess at his favorite cuisine is **Mexican**. His brother Fred however is a lover of **pastas** and **cheese** that means Fred's fabric cuisine is **Italian**



How does LSTM solve vanishing gradient?

- Makes it easier for RNN to preserve information
  - If forget gate is set to 1 for a cell dimension and input gate set to 0, then the information is **preserved indefinitely!**
  - In contrast, much harder for vanilla RNN to learn weight that preserves info in hidden state
- LSTM does not guarantee we have no vanishing gradient, but it makes it easier for the model to learn **long-distance dependencies**

LSTMs: Real-World Success

In 2013-2015, LSTMs started achieving state-of-the-art results

- Successful tasks include: handwriting recognition, speech recognition, machine translation, parsing, and image captioning as well as LM
- LSTMs became the **dominant approach** for most NLP tasks
  - Was used in Google Translate for a few years!
  - Even today, it is still used for some tasks (although a new approach is now dominant)

LSTM Explanation Additional Video

https://youtu.be/LfnrRPFhkuY?si=xhy9aURC2TevpFNR

Lab - 8

Exercise 1:

Training LSTM Model in PyTorch for Sentiment Analysis

https://www.datacamp.com/tutorial/nlp-with-pytorch-a-comprehensive-guide

Exercise 2:

Text generation via RNN and LSTMs (PyTorch)

https://www.kaggle.com/code/purvasingh/text-generation-via-rnn-and-lstms-pytorch

Final Projects - Group

a) Chatbot for Humber - students association assistant (Eng + French support)

b) Creative assistant for Humber marketing team (Eng + French support)

c) Document summarizer - PDF / word / reports (Eng + French support)

Confirm your group members over email by **March 25th**
Min members **5**, max **7**