RNN LSTM

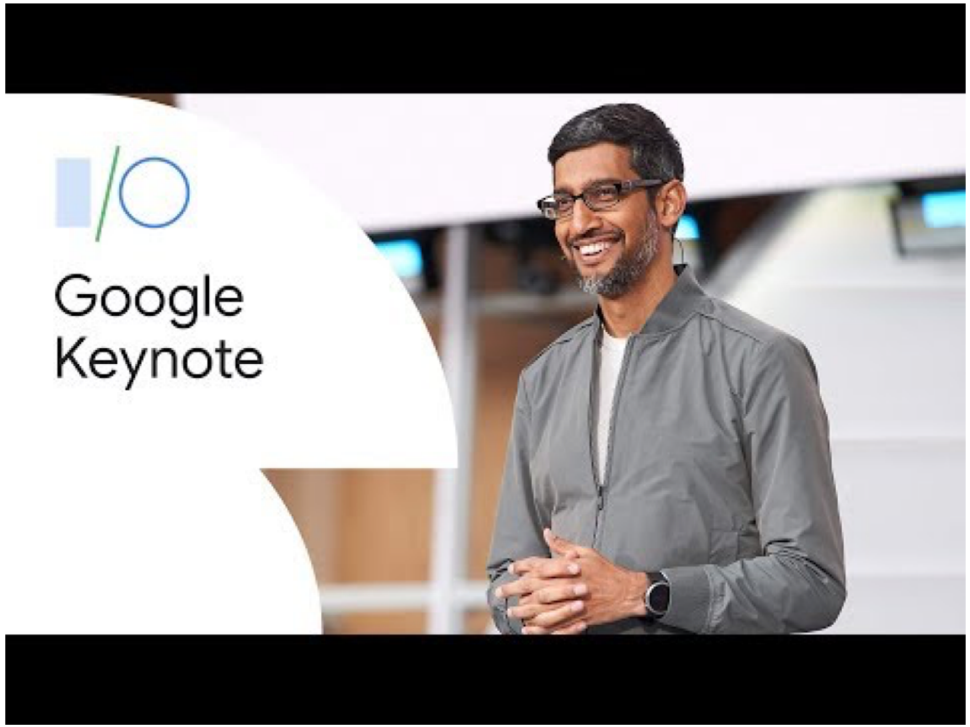
An Introduction

Objective of the session

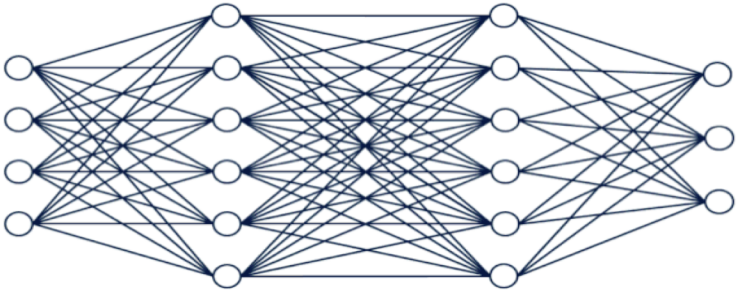
1. What are RNNs? What are the applications and issues with RNNs?
2. Why choose LSTMs?
3. Where to apply LSTMs?

We will also walk through an use case that shows how LSTMs can be used for a sequence problem

RNNs



Why not MLPs?



Simple forward neural networks (MLP) are great to:

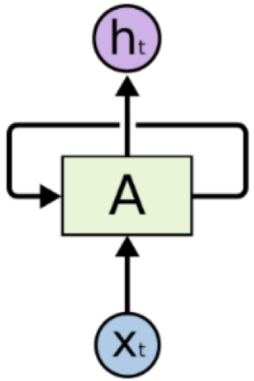
1. Capture complex non-linear relationships in the data
2. To do one to one predictions (Input: Email - Output: Spam/Not Spam)

Not so good:

1. Capturing context
2. Capturing temporal effects
3. Handling varying output

What are RNNs?

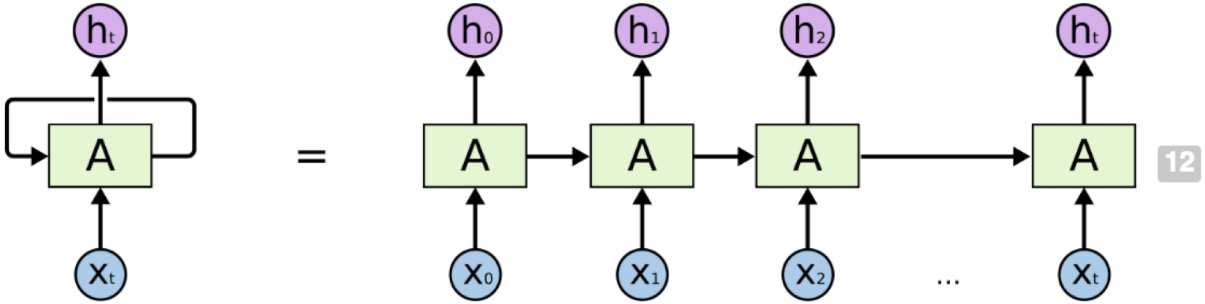
Recurrent Neural Nets are neural networks that are capable of holding an "**internal memory**". *The internal memory allows RNNs to the predict next value in **sequential/temporal** data based on inputs and past predictions (**old hidden state**)*



<u>Example:</u>

The color of the sky is *blue*

How do they really work?



An unrolled recurrent neural network.

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

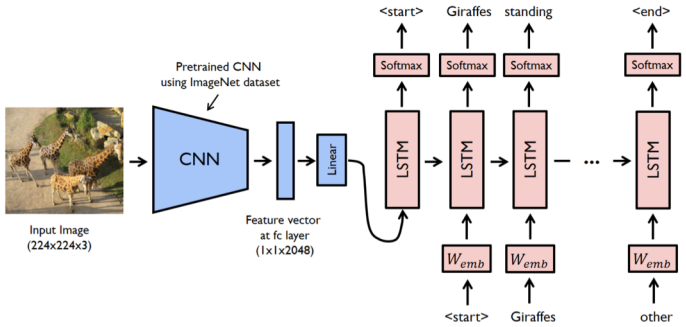
new state

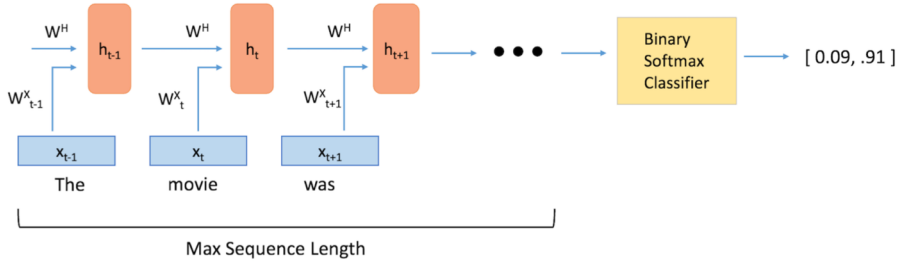
some function with parameters W

old state

input vector at some time step

Applications of RNN

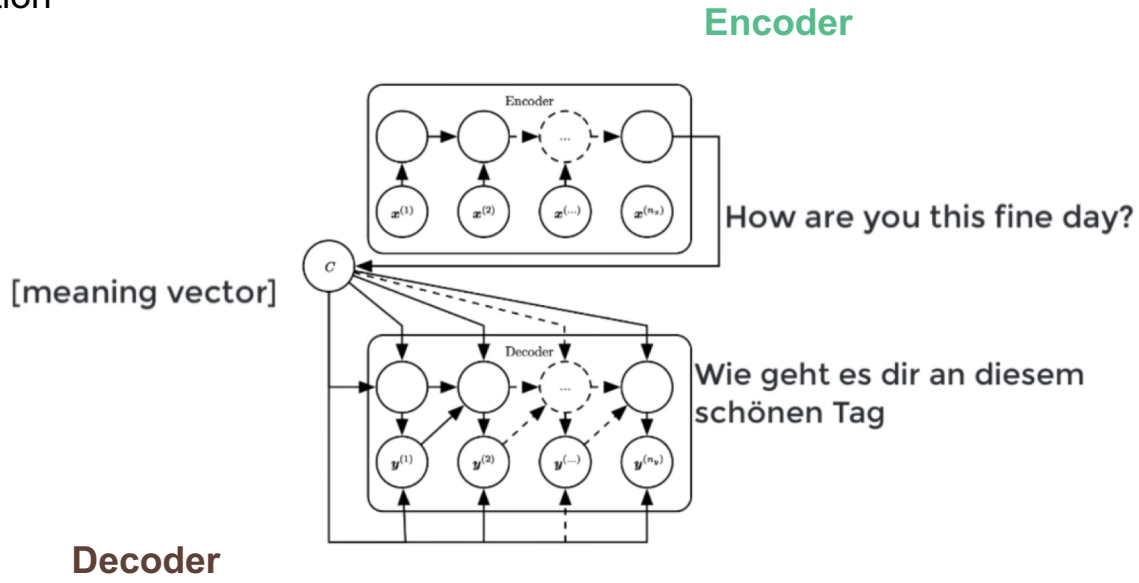




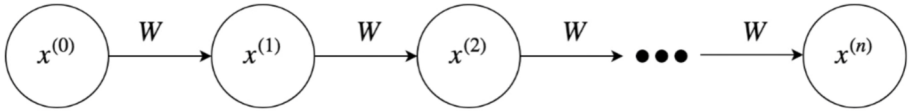
Vector to Sequence	Image Captioning, Music Generation, Text Generation
Sequence to Sequence	Text Summarization, Language Translation, Video Activity Recognition, Named Entity Recognition, Word Predictions
Sequence to Vector	Sentiment analysis, DNA sequence analysis, Time Series Classification, Word Predictions

Applications of RNN

Encoder - **Decoder** Architecture for Machine Translation



Issues with RNN



$$x^{(n)} = W^n x^{(0)} \qquad x^{(i)}, W \in \mathbb{R} \qquad x^{(i)} \in \mathbb{R}^D$$
$$i \in [0, n] \qquad W \in \mathbb{R}^{D \times D}$$

$$W^n x^{(0)} \rightarrow \begin{cases} \infty; & W > 1 \\ 0; & W < 1 \end{cases} \qquad \frac{\delta W^n x^{(0)}}{\delta W} \rightarrow \begin{cases} \infty; & W > 1 \\ 0; & W < 1 \end{cases}$$

1. Vanishing gradients
    a. When the weights in the matrix are small (<1), during backpropagation the gradients become very low that learning becomes very slow or stops learning
2. Exploding gradients
    a. When the weights in the matrix are high (>1), during backpropagation the gradients become very high leading to model not converging.

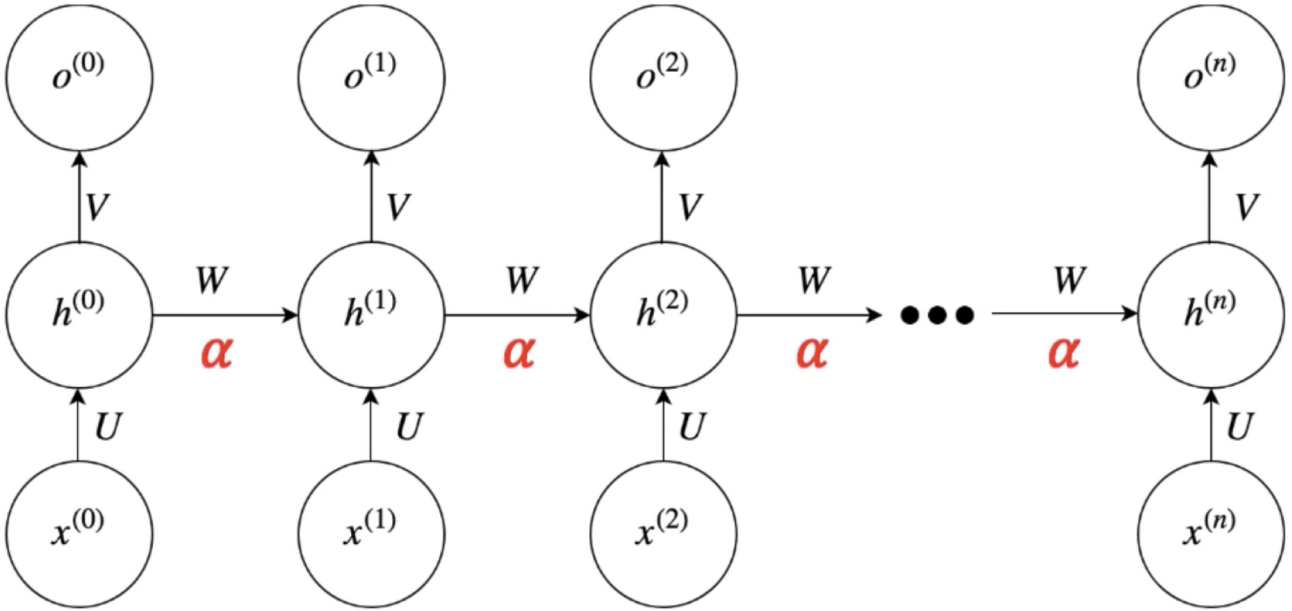
Issues with RNN

This renders RNNs ineffective for learning long term dependencies

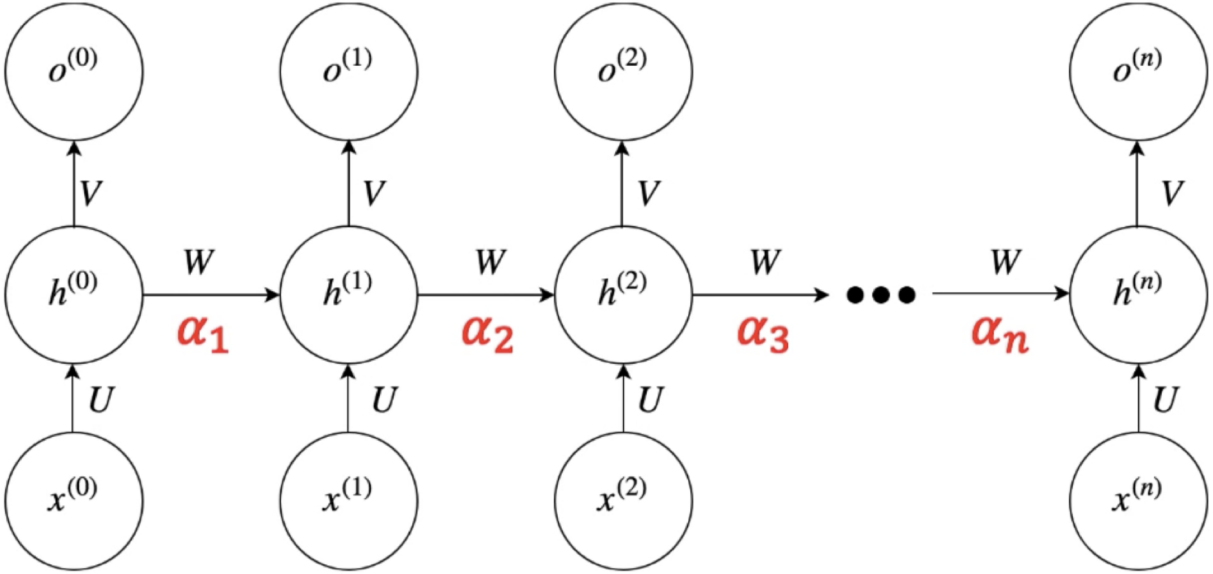
a. They are good for “The color of the sky is *blue*”, *ones where the context to learn is short term. But when we have long term dependencies to remember they don’t perform well enough: “I lived in France and I go to school there. ...[several sentences].. . I speak fluently in French”*

Solutions: Leaky Recurrent Units, GRUs (Gated Recurrent Units), LSTMs (Long Short Term Memory)

Leaky Recurrent Units

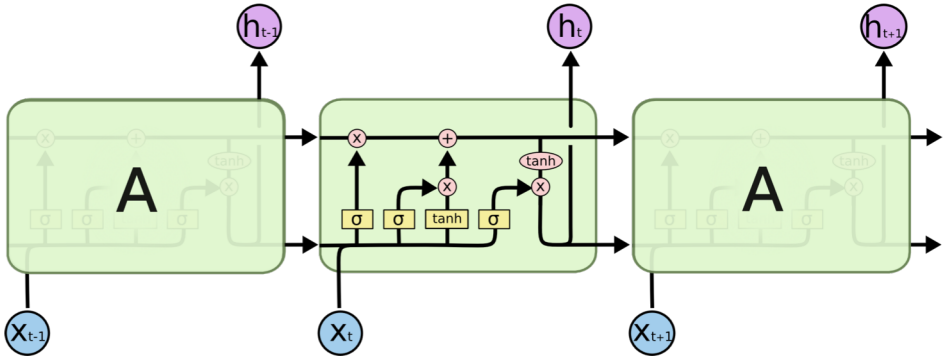


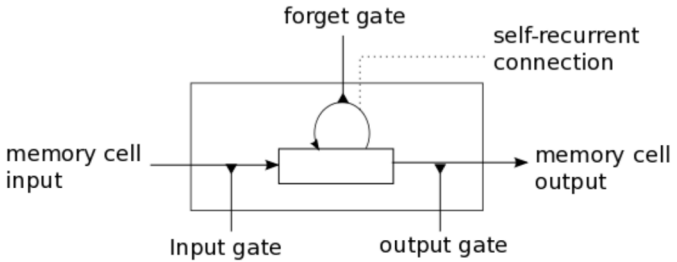
GRU


$o^{(0)}$
$o^{(1)}$
$o^{(2)}$
$o^{(n)}$
V
$h^{(0)}$
W
$h^{(1)}$
$h^{(2)}$
$h^{(n)}$
α_1
α_2
α_3
α_n
U
$x^{(0)}$
$x^{(1)}$
$x^{(2)}$
$x^{(n)}$

LSTMs at a high level

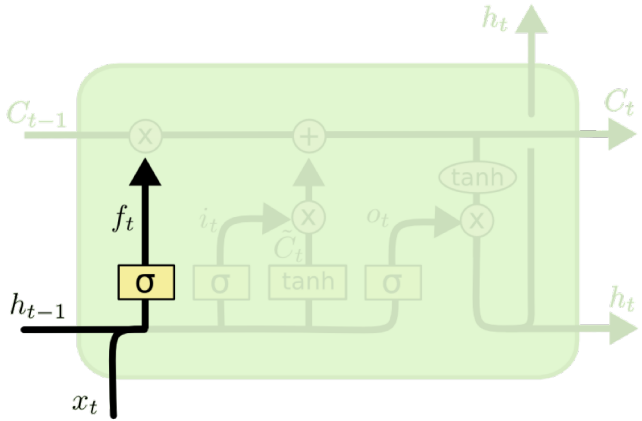
Imagine each LSTM cell to contain 3 different neurons. An input gate, forget gate and output gate. Each neuron has a specific purpose and is connected to serve this purpose





LSTMs teardown

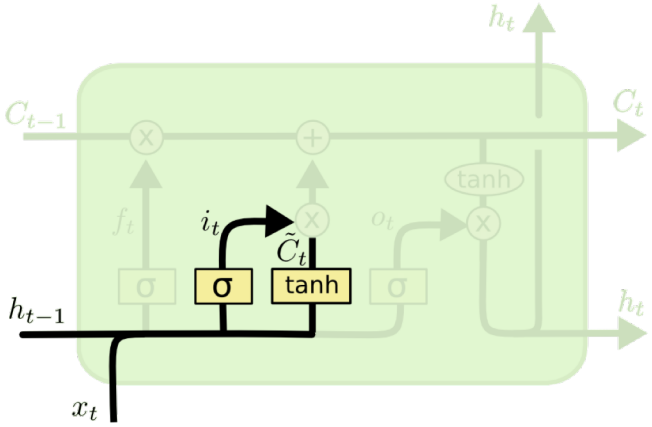
Forget Gate



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

LSTMs teardown

Input Gate

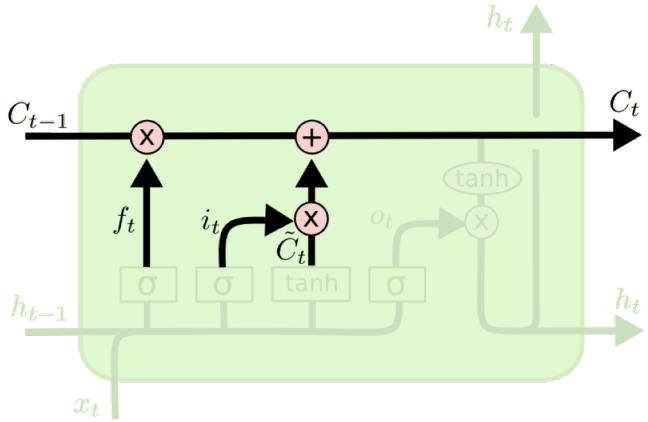


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

LSTMs teardown

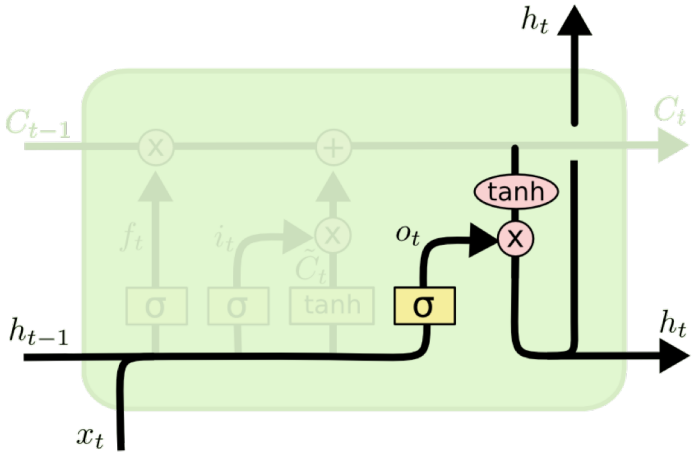
Combine for Cell State



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

LSTMs teardown

Output Gate



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

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

How I used LSTMs in my projects

Multiple Use Cases:

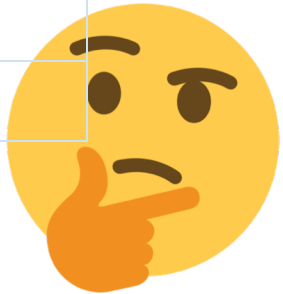
1. Patterns from sensors* (GPS) information feeding at different time steps
2. Used to identify type of sensor (sequence to vector)
3. Used to identify next action for the sensor (sequence to vector)
4. Used for anomaly detection (Sequence to Sequence) using AutoEncoder LSTM

Things to keep in mind:

1. Reshaping the data is critical prior to using the LSTM model [batch_size, seq_length, features] and the same to be done during predictions
2. Is your model stateful or stateless?
3. More LSTM units not necessarily better results
4. If you have a lot of static features with very few time series data, potentially a MLP might work better (complexity/performance tradeoff)

Choosing between LSTM vs Time Series Models

ARIMA (Gen. time series)	**LSTM**
Easy setup	Complex initial setup but greater reward*
Linear Relationships	Non-Linear Relationships
Requires making the series stationary	Non-stationary series can be handled
Performs better for long time series	Preferred for shorter time series but workarounds include attention
Complex for sequences of varying length	Handles sequence problems well
Scales okay with more data	Much better with more data



How to make LSTMs effective in your problems

Think of LSTMs when you want to do these:

- Short time series predictions, ex.
- Working with sensor devices with temporal data, ex.
- Anomaly detection in a time series
- Sentiment analysis
- Speech to text
- Translations

Remember LSTMs are more complex to set up initially and to tune, but probably more effective depending on the type of problem

How to use LSTMs to write your emails

Code